2011 Asian Test Symposium

Diagnosing Multiple Slow Gates for Performance Tuning in the face of Extreme Process Variations

Xi Qian, Adit D. Singh
ECE Department, Auburn University
Auburn, AL, USA
{xzq0002, adsingh}@auburn.edu

Abhijit Chatterjee
ECE Department, Georgia Institute of Technology
Atlanta, GA, USA
abhijit.chatterjee@ece.gatech.edu

***Abstract*—End-of-road map CMOS (<=10nm) technology is expected to display extreme random variability in device parameters, resulting in a very large spread in the speed of individual gates. Based on reasonable statistical estimates, virtually every large circuit in this environment can be expected to contain several extremely slow statistical outlier gates which will severely limit performance in synchronous designs. To address this challenge, gate level tuning techniques have recently been proposed [2] that can potentially speed up the slow gates to recover much of this lost performance. However, such tuning significantly increases power dissipation, and therefore must only be activated in the relative few performance limiting outlier gates. Consequently, application of such tuning techniques requires that the slow outlier gates be correctly diagnosed for proper tuning. This presents the challenging problem of diagnosing multiple delay faults in the circuit. In this paper we show how the performance tuning capability of the circuit can itself be exploited, in combination with scan delay tests, to address this problem. Our approach involves selectively tuning and speeding up subsets of suspect gates, and then uniquely identifying the slow outlier gates based on whether the tuning eliminates the slow path or not. We show that such an approach can correctly diagnose multiple slow gates in large circuits for successful performance tuning.**

Keywords- Multiple Delay Fault Diagnosis; Circuit Tuning; Outlier Parameter; Process Variation

I. INTRODUCTION AND BACKGROUND

Over the years, researchers have spent much effort in developing economical defect diagnostic methodologies. Until recently however, most of the effort targeted single stuck-at faults. Some progress has now also been reported on diagnosing single delay faults. However, multiple delay fault diagnosis, where the number of timing faults can be relatively large and unknown is a much more challenging problem.

Traditionally, the objective of fault diagnosis during high volume manufacturing test has been to identify systematic patterns of defects caused by design marginalities that are activated in some process windows. Once such defects are identified, the design can be modified to make it more robust and/or the process steps can be adjusted to minimize the likelihood of occurrence of such systematic defects. However, with diminishing feature sizes approaching atomic dimensions, random process variability is becoming a critical problem. Each device in the chip is subject to statistical effects such as random dopant fluctuation (RDF) and line edge roughness (LER) which can significantly impact its performance. A large chip with hundreds of million transistors can be expected to contain hundreds of extremely slow statistical outlier devices randomly scattered on the die, that are five or more standard deviations slower than nominal transistors. Such devices will dramatically limit the performance of synchronous designs. Unfortunately, because such effects result from random and not systematic effects in manufacturing, they cannot be eliminated through redesign or better process control.

One possible solution to this problem that is expected to become acute in end-of-road map CMOS (<=10nm) technologies is a circuit tuning technique that has been proposed [2] to speed a slow gate up to nearly nominal performance. Tunable gates are used in large circuits to substitute ordinary gates. Taking the structure of a tunable 2-input NAND gate for example, as shown in Figure 1, the programmable redundant tuning transistors Tp and Tn are disabled (turned off) if the gate demonstrates expected performance, i.e. all transistors are have acceptable device parameters. If a transistor in the gate has statistical extreme outlier parameters, which slows down either P pull-up or N pull-down network very significantly, the corresponding tuning pull-up or pull-down transistor can be enabled (programmed on) to provide additional current for charging or discharging output node capacitor. While this extra parallel path significantly speeds up the transition controlled by the excessively slow transistor as shown in [2], there is only minor switching time degeneration for the complementary network. To ensure effective "ratio logic" output voltage division for correct logic levels, Tp and Tn are carefully sized (as in pseudo NMOS logic). Furthermore, the two transistors clearly must not be enabled simultaneously. Also, while gates that have tuning transistors turned on will exhibit steady state power dissipation (like NMOS logic), if only a few dozen or even a few hundred gates in a multi-million gate circuit are so configured, the impact on overall power consumption is quite small. Nevertheless, *it is important to uniquely diagnose the critically slow gates for tuning so as to avoid turning on an excessive number of the tuning transistors.* It is envisioned that a configuration memory will hold the ON/OFF status of

all programmable tuning transistors. To minimize the area overhead of this memory, which would limit the silicon available for functional use, one proposal is to implement the memory using CVD (chemical vapor deposition) amorphous silicon transistors in a separate stacked layer among the metal interconnects. While such amorphous silicon transistors exhibit poor switching performance, which generally makes them unsuitable for use in logic, they can be quite acceptable for programming the tuning transistors because the inputs of the tuning transistors are set during test and configuration and never switched in functional operation.

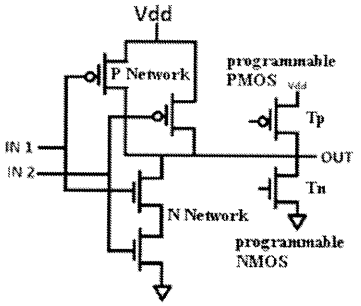

Figure 1. A Tunable NANDs Gate

It is important to recognize that this fine grained tuning approach is being proposed to address extreme variability expected in future end-of-road map technologies. A courser level of performance tuning is already employed in current microprocessors through body bias and clock skew control. *The multiple delay fault diagnosis approach presented in this paper can also be applied to such current designs.* However, such a methodology will be essential in an environment of extreme random variability where hundreds of gates in a large design will display delays 5-15X the nominal delay. Without proper diagnosis and tuning, the performance of such future nano-scale circuits will be greatly compromised.

The rest of this paper is organized as follow: In Section II we review prior work on multiple delay fault diagnosis research. Section III provides details of the proposed multiple delay fault diagnosis methodology. Section IV describes the experimental model on which simulations are carried out to evaluate the effectiveness of our proposed approach. Section V presents experimental results. Section VI concludes this paper.

II. Previous Work

Early research on delay fault diagnosis assumed the existence of a single fault. In [3]-[5], several algorithms were proposed to target only one gate causing observable timing violations. In [6], a flip-flop set partitioning methodology is introduced for finer diagnostic resolution and shorter diagnosis execution time. With the aggressive scaling of CMOS technology, however, the likelihood of more than one delay fault in a circuit cannot be ruled out. Therefore, a number of attempts at extending single delay fault diagnostic algorithms to multiple delay fault diagnosis have been published in recent years. However, in general these suffer from rapid degradation of diagnostic resolution, are as yet far from practical application in large circuits. For example, [7] proposed an early effort of "transplanting" the single delay fault diagnosis technique for the multiple fault case. The single-location-at-a-time (SLAT) diagnosis pattern prohibited its practical use because of excessive testing time and test pattern counts.

Other work on multiple delay fault diagnosis, such as [8]-[14], has concentrated on optimizing the diagnosis algorithm for efficient execution. For example, [8] focused on tracing of critical paths for fault site identification. Recently [14] provided a novel heuristic algorithm characterized with n-detection and timing-aware test pattern sets that not only help to locate fault sites, but also attempt to determine the upper and lower bounds of fault delays.

However, the published delay fault diagnosis research to date has two important aspects that limit its applicability to the performance tuning application which is the focus of this work.

First, it primarily targets manufacturing delay defects that can reasonably be expected to be somewhat limited in number in any well controlled process. Earlier work has not attempted delay fault diagnosis in the face of hundreds of exceptionally slow gates in an environment of extreme process variations.

More importantly, the diagnostic resolution targeted is limited to what can be achieved through logical deductions. For example, if an inverter drives a gate along the slow path, it may be impossible to determine if the gate delay fault is in the inverter or in the gate. However, this determination needs to be made for efficient gate level performance tuning; tuning two (or more) gates to address a single delay fault can lead to excessive power dissipation in the tuned circuit. Achieving this necessarily requires exploiting the performance tuning capability, for example by tuning one or the other gate until the delay fault is eliminated. *In this paper we develop and evaluate an efficient multiple delay fault diagnosis approach for such an environment.*

III. Proposed Methodology

A. Iterative Diagnose and Reconfigure Approach

The major challenge in diagnosing multiple delay faults turns out to be that, because of limited access to internal circuitry nodes, one fault is difficult to distinguish from others since many different delay faults can result in the same timing violation behavior at the observation points. Moreover, for a given delay test pattern, some faults may only be excited along short paths and may remain hidden in timing slacks and escape detection, thereby confusing the diagnosis. The scan chain structure further restricts delay test patterns (vector pairs) that can be applied for effective delay fault diagnosis.

In this paper we show how the performance tuning capability of the circuit can itself be exploited in combination with scan delay tests, to address the multiple delay fault diagnosis problem. Our approach involves

selectively tuning and speeding up subsets of suspect gates, and then uniquely identifying the slow outlier gates based on whether the tuning eliminates the slow path or not. We show that such an approach can correctly diagnose multiple slow gates in large circuits for successful performance tuning.

The ability to reconfigure the tuning transistors using the configuration memory allows us to observe the timing change on tested path after a different combination of potentially faulty gates is tuned. The impact of tuning on the delay of the gates is shown in Table I, where we use the term "semi-normal" to refer tuned gates capable of passing residing paths to given timing constrains, even if under slowest (worst) cases.

TABLE I. GATE DELAY BEFORE/AFTER TUNING

Gate Switching Speed	
Before Tuning	**After Tuning**
Normal	Normal
Slow	Semi-Normal
Unknown	**Semi-Normal**

Now if the observed logic failures for a given two vector timing test, for which initially M suspicious faulty gates are identified, disappear when a given set of N out of all M candidates are properly tuned, we can declare that the remaining M-N sites to be fault-free, and gates within the N-sized subset form a new smaller suspect list, of which some or all gates may contain a delay fault.

Thus, instead of directly identifying gates containing delay faults, it is more efficient to exclude fault free ones from an initial starting set of suspicious gates, which includes all possible gates that can lead to the timing failure under observation. We then apply the failing test pattern repeatedly while turning on different combinations of tuning transistors, until we zero in on the smallest set of gates that must all be tuned to allow the path to pass the timing test. This is the set of gates with delay faults detected by the given delay test pattern (vector pair). We repeat this process for all failing delay test patterns to identify all gates with delay defects.

To more fully describe the proposed approach, we first define three terms:

Observation point – any primary output, or data input of scan flip-flops whose contents can be scanned out for observation.

Timing violation – incorrect final logic value at an observation point at the active clock edge.

Diagnosis session – the process of identifying all faulty gates leading to timing violation at one observation point for one test pattern.

One delay fault in a gate can cause timing violations not only for different test patterns, but also at different observation points under the same test pattern, because of fan-outs at circuit nodes. During every diagnosis session, we keep every known faulty gate properly tuned once it is uniquely diagnosed, to speed up the slow MOSFET network and eliminate any faulty timing influence in the subsequent test and diagnosis process. By doing so, additional timing violations in subsequent test can have a chance to naturally disappear if all implicated faults are "fixed" in earlier diagnosis sessions.

As many diagnosis sessions as needed are conducted until no timing violation retains for the applied test pattern. A diagnostic process ends when all patterns pass the timing constrains. Our methodology works with the available scan delay test set, with the diagnosis algorithm working with all failing delay test patterns. No additional effort on diagnostic vector generation is required.

In evaluating the effectiveness of our methodology, we make the following three assumptions:

a) The timing violation at an observation point disappears when all involved faulty gates are properly tuned. In other words, we assume that the clock period is long enough that the tuned faulty gates, which will still be slower than nominal, will not cause timing violation even if cascaded together.

b) The timing violation at an observation point disappears ONLY when all involved faulty gates are properly tuned. In other words, accumulated speed-up of tuned fault-free gates cannot compensate the slow-down resulted from the faulty gate(s).

c) Tuned gates do not result in new timing violations. While speeding up one transition for a gate by enabling corresponding tuning transistor, the switching time of the complementary transition may degrade somewhat. However, such degradations, even if accumulated over multiple gates, are assumed not to cause new time violations.

In practice, appropriate design and test techniques can be adopted to satisfy these conditions. Balanced paths and timing slack margin, for example, are partially supporting b) and c). In our simulation experiments, we used reduced VDD supply [1] to fully support all three requirements, as discussed further in Section IV.

B. Acquiring Minimal Starting Candidate Set

As mentioned earlier, the objective of every diagnosis session is to identify all implicated faulty gates by gradually excluding good gates from a suspicious gate set. The size of this suspicious set should be made as small as possible at the start, to save diagnosis time and avoid the risk of damaging the device with excessive static power dissipation from tuning too many gates.

While the complete gate list from a backtrace from an observation point can be large (up to hundreds of gates in our experiments), only a small portion (around 15%~25%) of these actually switched under an applied test pattern (vector pair). Moreover, approximately half of these transitions are masked by the logic and will not contribute to switching at a path output (e.g. observation point). This small gate list is what we use as the starting set for a diagnosis session when a timing failure is flagged at an observation point for a given test pattern.

Clearly, this starting set of suspect gates is both test pattern dependent and observation point dependent. One way to quickly acquire starting sets for every diagnosis session is to create a fault dictionary represented as a table that can be

searched by the test pattern index and observation point; each table entry consisting of a list of possible faulty gates, along with the polarity of the failing transition at that gate. Figure 2 illustrates this idea. As an example, the RG1 entry for test pattern P1 and output O1 indicates the possibility of a slow-rising fault on gate G1 output, while FG6 refers to the slow-falling fault at output of gate G6, and so on for the other listed gates.

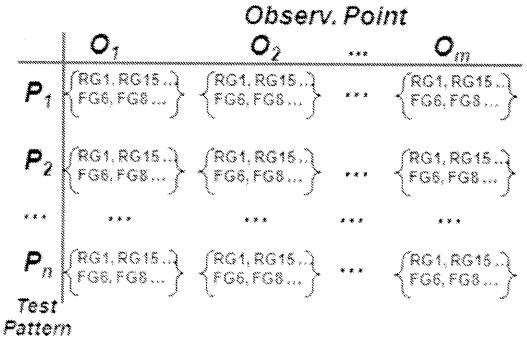

Figure 2. Delay Fault Dictionary Structure

One drawback of this dictionary based approach is the needed storage for this table in large circuits. However, it can provide quick access to a starting set of possible faulty gates for each diagnosis sessions, and while expensive, the size still remains viable. An alternative is real-time analysis of the circuit using ATPG (Automatic Test Pattern Generator) tools to identify the list of suspicious gates.

C. *Multiple Fault Diagnosis Algorithm*

To review the expected behavior of a slow path after a subset of suspicious gates is tuned: the path passes timing if all faulty gates fall into the tuned set; it continues to show timing violation if at least one faulty gate is not tuned.

Now to compact the suspicious set over time, subsets of fault-free gates are identified by repeating the timing test with different tuning transistor selection memory reconfigurations. Diagnosis conclusions can be made as shown in Table II.

TABLE II. DIAGNOSIS CRITERIA 1.

All set A tuned	Only subset B tuned	Conclusions
Pass timing	Pass timing	No faults in B; all faults in A-B
Pass timing	Fail timing	At least one fault in B

Clearly, to identify an unknown number of faults implicated by a slow path, the starting/remaining set should be grouped into subsets and fault-free ones eliminated from the suspicious set. The challenge is how to create the set of gates for every test iteration, so as to achieve diagnosis most efficiently.

To illustrate how this can be done, let us first assume that there is only a single fault in the circuit. Recall that a diagnosis session involves repeating the failing timing test with different tuning transistors configurations. A binary search for this case can quickly locate the fault site, as shown in Figure 3. Upon reapplications of the same test pattern, half of all remaining suspect gates are tuned while the other half not. A recurrence or elimination of the path timing violation indicates faulty gate residing inside the untuned or tuned subset, respectively. The fault-free half is removed from the suspect set and the process is repeated until a single faulty gate is finally identified. Clearly this diagnosis can be achieved in a number of iterations that are logarithmic in the size of the starting suspicion set.

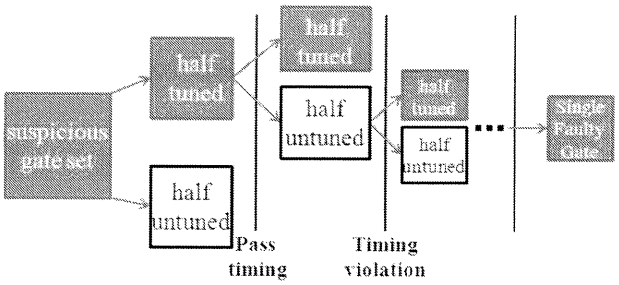

Figure 3. Binary Search of Single Fault

For diagnosing multiple fault (since the actual fault count is unknown, it is possible that there may be only one fault), the basic principle remains the same. The idea is to divide up the suspect fault list with the objective of always having at least one fault-free subset dropped from the suspect set during each test iteration, until all the faults are identified.

We first assume, because of the observed timing failure, that the number of faulty gates is at least one. We split up the set of suspicious gates into two equal subsets. The failing test is repeated with each subset tuned in turn. If the test passes when one of the subsets is tuned, then the other (unturned) subset cannot contain any failing gates and can be removed from the list of suspicious gates; the process is then repeated in another iteration. If both subsets are found to contain failing gates, then the number faulty gates is at least two. We must therefore create at least three subsets and tune two of them at a time in the hope of finding a delay fault free subset. If the delay test passes for some combination of two tuned subsets and one unturned, then clearly the unturned subset does not contain any faulty gates. Now if all three subsets are found to contain faulty gates then there are at least 3 faulty gates and we must create 4 subsets in the next iteration. If in any iteration we find a subset free of faulty gates, it is eliminated from the list of suspect gates. On occasion, by random chance, multiple faulty gates may cluster in a few of the subsets and multiple subsets may be found to be defect free and can be eliminated. Observe that criteria in Table III are used to determine if the UNTUNED subset is fault-free or not.

TABLE III. DIAGNOSIS CRITERIA 2.

N-1 out of N subsets tuned	
Circuit Timing	**Conclusions**
Pass	No fault in untuned subset
Fail	At least one fault in untuned subset

Selecting N-1 out of N gates can be done in N possible combinations. Thus the number of iterations (time) needed for one step in the diagnosis session is N.

The algorithm for multiple fault diagnosis can be described as follows:

1) Start with subset number N=2, and divide (remaining) implicated gates into N approximately equal subsets.

2) Tune all combinations N-1 subsets in turn to find and eliminate subsets not containing any slow gates.

3) Number of subsets N should be always one larger than the minimum known number of slow faulty gates in the circuit. Increment the number N by one if no subset can be identified fault-free durning a round of testing.

4) Continue until each subset has a single gate.

Figure 4 illustrates an example of a diagnosis session assuming 4 faults in the starting set of 16 suspicious gates. Here the suspect gates at each stage are always uniformly distributed into all subsets, and smallest-sized fault-free subset(s) dropped for every regrouping. In practice, the faulty gates could end up clustered in fewer subsets at some steps, allowing for the elimination of more fault free subsets and faster diagnosis.

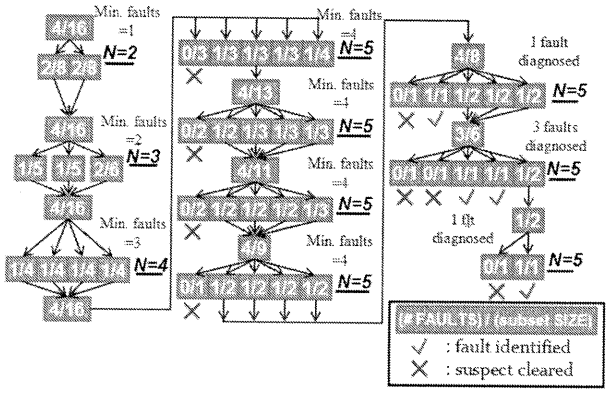

Figure 4. Illustration of the multiple fault diagnosis algorithm for the case of 4 faulty gates out of the 16 initially suspected

In this example, the suspicious set is regrouped 10 times before having all faults identified. For the first 3, no subset can be dropped as at least one fault exists in every subset. The test pattern is applied the same times of N for every regrouping. Note that the last regrouping is not a binary search for a single fault since the number of faults is unknown before all are identified. Thus a total of 2+3+4+5+5+5+5+5+2=41 reapplications of test pattern and reconfigurations of tuning transistor selection memory are required to diagnose and identify all implicated faults. This is the worst case.

It's important to point out that since P and N networks of a CMOS gate are not to be tuned simultaneously, a priority complementary check for identified faults is necessary. For example, if Gate K is already demonstrated slow in P-network, and is potentially N-network slow for current diagnosis session, the formerly enabled pull-up tuning should be disabled and the N-network should be firstly tested by having all other suspicious gates properly tuned. Once a gate is determined handicapping the overall device performance on both P and N networks, no further diagnosis for this circuit is necessary.

IV. Experimenal Modeling and Simulation

To evaluate the effectiveness of our proposed methodology, we conducted simulation experiments using the 3 largest ISCAS'89 benchmark circuits, i.e. s38584, s38417 and s35932. Our experiments, involving timing simulations, aim to verify if several extremely slow outlier gates in a large design can indeed be correctly diagnosed using the approach described in this paper, for the purposes of performance tuning.

A. Experimental Model

The experimental model that we adopt is shown in Figure 5.

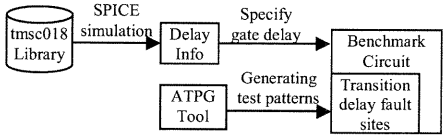

Figure 5. Experiment Model

Accurate gate level delay information is first acquired from SPICE simulation using gate layouts in tsmc018 technology and the Virginia Tech open source cell library. We then rewrite the Verilog HDL netlist of all tested benchmark circuits with parameterized descriptions of all primitive gates. Every gate can be individually designated with delay values with respect to gate type, input combination, number of output fan-out stems, VDD supply voltage, threshold voltage, etc, and these parameters are reassigned for every test/diagnosis case.

TABLE IV. Circuit Information

Circuit	s38584	s38417	s35932
# FFs	1426	1636	1728
# Gates	19253	22179	16065
# LOC Patterns	612	455	203
Delay FC (TDF)	96.96%	93.34%	87.65%

For each circuit, we use ATPG Tool DFTAdvisor to develop a set of LOC (launch on capture) test patterns aiming for as high delay fault coverage as possible (see in Table IV). These test patterns are then applied onto these modified Verilog netlists containing gate delay information to help observing timing violations.

B. Extreme Outlier Delay Injection

Recall that random process variability is expected to significantly limit gate performance. This is because the largest expected contributors to variability in future highly scaled devices, random dopant fluctuations (RDF) and line edge roughness (LER), both cause significant random variations in transistor threshold voltages. And critically,

gate delays at low VDD are nearly exponentially impacted by threshold voltage variations.

At current feature sizes, the standard deviation (sigma) observed in device threshold voltages in some process is already in the 50mV range, and this will likely increase significantly over the next few technology generations. Since 3 out of 10 million parts drawn from a normal distribution are 5 sigma or more beyond the mean on either side, several dozen, even a few hundred, devices in a large chip may see a 0.25V or greater increase in their threshold voltages. Since gate delay is a near exponential function of the threshold voltage, at low VDD, the few random gates with such a large increase in threshold voltage can be 5-15X slower than a nominal gate.

To study the magnitude of this delay increase with variation in the threshold voltage, delay simulations for the output falling transition an inverter are presented in Table V. For this simulation, an inverter chain was laid out using the 0.18micron cell library, and electrical parameters were extracted for spice simulation. The inverter, observed for the output falling delay, was embedded in the middle of the chain and therefore subject to the load of another similar inverter. SPICE simulations were run with various threshold voltages as shown. Note that for this simulation, and for all the delay tests conducted for the experiments reported in this paper, the power supply voltage VDD is the minimum supported by the circuit, here assumed to be 0.9 volts. This is to maximize the extra "faulty" delay due to the slow outlier gates, thereby making delay detection easier.

TABLE V. FALL-TIME OUTPUT INVERTER DELAYS DRIVING SINGLE INVERTER LOAD

TSMC 180nm, nmos nominal Vth=0.4725V, VDD=0.9V			
ΔVth	**Delay(ps)**	**ΔVth**	**Delay(ps)**
0	163.84	0	163.83
-0.05V	128.81	+0.05V	219.62
-0.10V	105.22	+0.10V	317.50
-0.15V	88.63	+0.15V	511.91
-0.20V	76.37	+0.20V	959.99
-0.25V	67.59	+0.25V	2167.80
-0.30V	59.85	+0.30V	5868.30

Observe the dramatic increase in gate delay once the increase in the threshold voltage increases beyond about 0.15volts. Note also that the speed-up for lower transistor thresholds is much less than the slow-down from equivalent increases in threshold, which is why the impact from variability on long path delays does not average out. Extreme variability increases path delays, and results in significant loss of performance.

Because that the increase in gate delay falls off very rapidly for smaller increases in threshold voltages, and less than 0.2% of the transistor threshold voltages are beyond 3 sigma, in our simulation experiments, for simplicity, we classify gates into only two groups. A few extremely slow "faulty" gates whose delays are estimated using SPICE simulation of the 0.18 micron process, assuming a 0.2 volt increase in the nominal threshold voltage available from the technology files. (From the Table 5, the "faulty" transition in these gates is approximately 5X slower.) The remaining are nominal gates with gate delays again estimated using SPICE simulations. These extremely slow gates represent the slow gate "fault" injection in our simulation.

C. *The timing simulation*

To further ensure realism in our timing simulations, we use these following simulation conditions and timing approximations:

1) All the test patterns in the test set are first applied to a "fault-free" copy of the target circuit to find the timing delay of the longest path length. Then an additional 10% slack is added to determine the operational clock cycle.

2) An timing violation during test application is recognized if the length of any path exceeds the operational clock period (which includes the above 10% timing margin).

3) Gate delay information from SPICE simulations, using a 0.2V elevation in the threshold voltage is used to randomly inject faulty gate delays in the circuit.

4) A delay fault is only inserted to one MOS network of every chosen faulty gate. (While it is possible that both the pull-up and pull down network in a gate may contain extremely slow transitors, such accurance is likely to be extremely rare. Moreover, such a gate cannot be tuned and may need to be discarded.)

5) For tuned fault-free gates, we use: 0.9X nominal delay values for the speeded up network and 1.1X nominal dealy values for the complementay network. These numbers were estimated from SPICE simulations.

6) For tuned faulty gates, we use: 1.2X nominal delay values for the speeded up network and 1.1X nominal dealy values for the complementay network. Again, these numbers were estimated from SPICE simulations.

7) All faults implicated by a test along any slow path are fully diagnosed when and only when both of the following conditions are satisfied: (1) the timing violation during test remains even when all gates tuned to speed up the path transition except the fastest faulty gate; (2) timing violation disappears even when all faulty gates properly tuned and all fault-free gates are tuned in the complementary manner to slow down the transitions along the path. These two contitions enforce worst case diagnosis.

8) All diagnosed faulty gates remain properly tuned for the remainder of the test once identified.

9) The complete delay test set, all test patterns, are reaplied after complete diagnosis and gate tuning to varify that all timing violations resulting from slow gates are eliminated.

V. EXPERIMENTAL RESULTS

In the simulation experiments, diagnosis was attempted under three different fault injection scenarios: single fault, multiple faults detectable by the same test vector, and a large number of random faults detected over multiple vectors.

A. *Single Fault Insertion Experiment*

For single fault insertion, we randomly inserted a single detectable delay fault (note in Table IV that fault coverage for every circuit is less than 100%), into each circuit, and applied all the test patterns to see how many of them display timing violations. We then randomly picked 3 of these patterns, to check if the inserted fault can be correctly diagnosed. If the fault caused timing violations at more than one observation point because of fan-outs at circuit nodes, we randomly chose one for the diagnosis session.

The sample results summarized in Table VI demonstrate the capability of proposed methodology to diagnose the inserted single delay faults. The results were seen to hold for a large number of similar experiments.

TABLE VI. SINGLE FAULT DIAGNOSIS

Circuit	s38584	s38417	s35932
Faulty Gate	AND2_3930	NAND3_83	OR2_777
# Pattern w/ Timing violation	216	175	69
Diagnosed by pattern 1	Y	Y	Y
Diagnosed by pattern 2	Y	Y	Y
Diagnosed by pattern 3	Y	Y	Y
# Failing pattern w/ fault properly tuned	0	0	0

B. *Multiple Fault Insertion Experiment*

For multiple faults insertion, we randomly picked one test pattern and used ATPG to find faults detected by the test pattern at a selected test observation point. From these we randomly chose three faults. Then we used ATPG to generate additional vectors targeting these faults. We then randomly picked three from generated test patterns capable of detecting all three faults. These were used to check if the inserted faults can be correctly diagnosed using each of the test vectors and our diagnosis algorithm.

TABLE VII. MULTIPLE FAULT DIAGNOSIS

Circuit	s38584	s38417	s35932
Faulty Gates (# fan-out)	AND3_53(1) OR2_1427(1) AND2_2517	NOT_1970(1) NAND2_94(1) NAND2_95(1)	OR2_242(1) AND2_111(1) NOT_815(1)
# Patterns Detecting all 3 faults	42	29	8
# Faults diagnosed by chosen pattern (# sessions)	3(1)	3(1)	3(1)
# Faults diagnosed by pattern 2 (# sessions)	3(1)	3(1)	3(2)
# Faults diagnosed by pattern 3 (# sessions)	3(2)	3(2)	3(2)
# Failing pattern w/ faults properly tuned	0	0	0

We can observe from the results that multiple faults can separately cause timing failure along different paths for different test patterns. Moreover, because extreme outlier delay faults are relatively rare, each fault influences the circuit almost independently from others, largely behaving like isolated single faults. This greatly increases the probability of every fault being identified, by all detecting patterns.

C. *Large Number of Faults Insertion Experiment*

In this experiment 1% of the circuit gates, e.g. 193 out of all the gates in s38584, are randomly injected with delay faults. Such a situation is highly unlikely, in fact virtually impossible, in practice, but we use the scenario to evaluate the overall diagnosis resolution of our algorithm. All test patterns are applied and as many diagnosis sessions as needed are conducted whenever timing violations occur. The whole process is repeated two more time with reshuffled test pattern orders.

The results in Table VIII show the effect of "delay fault repair" in our diagnosis. While this collection of 193 faults cause timing violations for 603 test patterns, only around 80 patterns are needed for diagnosis, if faulty gates are kept properly tuned once identified. Also, the order of test pattern application influences the number of required diagnosis sessions.

TABLE VIII. LARGE NUMBER FAULTS DIAGNOSIS

Circuit	s38584
# inserted faults	193
# diagnosed faults	188
# fault escapes	5
# potentially diagnosed faults	4
# patterns applied	612
# failing patterns	603
# patterns involving diagnosis sessions	79/80/79
# diagnosis sessions	145/148/146
# max. faults diagnosed in one session	3/3/2
# failing pattern w/ all identified faults properly tuned	0

One fault is identified as not covered by given test patterns, because of the structural limitations of LOC patterns. Another four faults escaped detection because they were located on very short paths. These faults are labeled potentially diagnosable in Table VIII if their size is sufficiently large. Recall that we use a fixed minimum outlier delay fault size for all injected faults. In practice, the statistical slow gates can have much larger delays. To check the potential diagnosis of these faults, we manually increased the injected delay to cause a timing violation at the rated clock. Subsequently they were all successfully diagnosed by our algorithm.

VI. CONCLUSION

In this paper we have proposed a new diagnosis methodology aimed at locating multiple large gate level delay faults in a circuit. This work is motivated by the observation that "End-of-Road-map" CMOS (<=10nm) technology is expected to display extreme random variability in device parameters, resulting in a very large spread in the speed of individual gates. Based on reasonable statistical estimates, virtually every large circuit in the near future can be expected to contain several dozen, or even several

hundred, extremely slow outlier gates which will severely limit performance in synchronous designs. The individual chip level randomness of this phenomenon prevents it being addressed at the design stage; researchers are proposing to address it with post manufacture circuit tuning for enhanced performance. Such a strategy requires the many extremely slow gates in a large chip to be correctly diagnosed.

In this paper we show how the performance tuning capability of the circuit can itself be exploited, in combination with scan delay tests, to address this problem. Our approach involves selectively tuning and speeding up subsets of suspect gates, and then uniquely identifying the slow outlier gates based on whether the tuning eliminates the slow path or not. We show that such an approach can correctly diagnose multiple slow gates in large circuits for successful performance tuning.

The proposed diagnosis scheme is specifically designed to work with a recently published circuit tuning technique [2], but can be also adapted for other performance tuning approaches, including currently employed coarse granularity performance. It is likely that the need for such many delay fault diagnosis strategies will become increasingly important to recover performance in the face of extreme variability in highly scaled technologies over the next decade.

REFERENCES

[1] X. Qian and A. D. Singh, "Distinguishing Resistive Small Delay Defects from Random Parameter Variations", Asian Testing Symposium, 2010.

[2] K. Mishra, A. Faraz, A. D. Singh, Path Delay Tuning for Performance Gain in the face of Random Manufacturing Variations, in Proc. International Conference on VLSI Design 2011.

[3] V. J. Mehta, Z. Wang, M. Marek-Sadowska, K. H. Tsai, and J. Rajski, "Delay fault diagnosis for non-robust test," in Proc. 7th Int. Symp. Quality Electron. Des., Mar. 2006, pp. 463–472.

[4] Z. Wang, M. Marek-Sadowska, K. H. Tsai, and J. Rajski, "Delay-fault diagnosis using timing information," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 24, no. 9, pp. 1315–1325, Sep. 2005.

[5] K. Yang and K.-T. Cheng, "Timing-reasoning-based delay fault diagnosis," in Proc. Des. Autom. Test Eur., Mar. 2006, vol. 1, pp. 418–423.

[6] Ahmed, N.; Ravikumar, C.P.; Tehranipoor, M.; Plusquellic, J.; "At-speed transition fault testing with low speed scan enable", VLSI Test Symposium, 2005.

[7] T. Bartenstein, D. Heaberlin, L. Huisman, and D. Sliwinski, "Diagnosing combinational logic designs using the single location at-a-time (SLAT) paradigm," in Proc. Int. Test Conf., Oct./Nov. 2001, pp. 287–296.

[8] J. Ghosh-Dastidar and N. A. Touba, "A systematic approach for diagnosing multiple delay faults," in Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst., Nov. 1998, pp. 211–216.

[9] Y. C. Lin, F. Lu, and K.-T. Cheng, "Multiple-fault diagnosis based on single-fault activation and single-output observation," in Proc. Des. Autom. Test Eur., Mar. 2006, pp. 1–6.

[10] J. B. Liu, A. Veneris, and S. Safarpour, "Diagnosing multiple transition faults in the absence of timing information," in Proc. 15th ACM Great Lakes Symp. VLSI, 2005, pp. 193–196.

[11] Z. Wang, K. H. Tsai, M. Marek-Sadowska, and J. Rajski, "An efficient and effective methodology on the multiple fault diagnosis," in Proc. Int. Test Conf., Sep./Oct. 2003, pp. 329–338.

[12] Z. Wang, M. Marek-Sadowska, K. H. Tsai, and J. Rajski, "Multiple fault diagnosis using n-detection tests," in Proc. 21st Int. Conf. Comput. Des., Oct. 2003, pp. 198–201.

[13] Dastidar, J.G.; Touba, N.A., "A systematic approach for diagnosing multiple delay faults", Defect and Fault Tolerance in VLSI Systems. IEEE International Symposium, 1998

[14] Mehta, V.J.; Marek-Sadowska, M.; Kun-Han Tsai; Rajski, J.; "Timing-Aware Multiple-Delay-Fault Diagnosis" Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions, 2009

[15] Tekumalla, R.C., "On test set generation for efficient path delay fault diagnosis" VLSI Test Symposium, 2000.

[16] Ying-Yen Chen; Jing-Jia Liou; "Diagnosis Framework for Locating Failed Segments of Path Delay Faults" Very Large Scale Integration (VLSI) Systems, IEEE Transactions, 2008

[17] Ghosh-Dastidar, J.; Touba, N.A.; "Adaptive techniques for improving delay fault diagnosis" VLSI Test Symposium, 1999

[18] Cox, H.; Rajski, J.; "A method of fault analysis for test generation and fault diagnosis" Computer-Aided Design of Integrated Circuits and Systems, IEEE Transaction, 1988

[19] A. D. Singh, "Scan Based Testing of Dual/Multi Core Processors for Small Delay Defects", International Test Conference, 2008.

[20] A. D. Singh, "A self-timed structural test methodology for timing anomalies due to defects and process variations", International Test Conference, 2005.

[21] H. Yan and A. D. Singh, "A New Delay Test Based on Delay Defect Detection Within Slack Intervals (DDSI)" IEEE Transactions on Very Large Scale Integration Systems, vol. 14, 2006, pp. 1216-1226

[22] H. Yan, A. D. Singh; "Experiments at Detecting Delay Faults using Multiple Higher Frequency Clocks and Results from Neighboring Die", Proceedings of the International Test Conference, 2003.

[23] C. Barnhart, "Delay Testing for Nanometer Chips", in Chip Design, August/September, 2004, pp. 8-14.

[24] B. D. Cory, R. Kapur and B. Underwood, "Speed binning with path delay test in 150-nm technology", IEEE Design & Test of Computers, vol. 20, 2003, pp. 41-45.

[25] R. David, S. Rahal and J. L. Rainard, "Some relationships between delay testing and stuck-open testing in CMOS circuits", European Design Automation Conference, 1990, pp. 339-343.

[26] B. Dervisoglu and G. Strong, "Design For Testability: Using Scan path Techniques for Path Delay Test and Measurement", International Test Conference, 1991, pp. 365-374.

[27] D. Dumas, P. Girard, C. Landrault and S. Pravossoudovitch, "Effectiveness of a variable sampling time strategy for delay fault diagnosis", European Design and Test Conference, 1994, pp. 518-523.

[28] W. B. Jone and Y. P. Ho, "Delay Fault Coverage Enhancement Using Variable Observation Times", Journal of Electronic Testing: Theory and Applications, October 1997, pp. 131-146.

[29] K. S. Kim, S. Mitra and P. G. Ryan, "Delay defect characteristics and testing strategies", IEEE Design & Test of Computers, vol. 20, 2003, pp. 8-16.