Analysis and Mitigation of IR-Drop Induced Scan Shift-Errors

Holst, Stefan; Schneider, Eric; Kawagoe, Koshi; Kochte, Michael A.; Miyase, Kohei; Wunderlich, Hans-Joachim; Kajihara, Seiji; Wen, Xiaoqing

Proceedings of the IEEE International Test Conference (ITC’17) Fort Worth, Texas, USA, 31 October-2 November 2017
do: http://dx.doi.org/10.1109/TEST.2017.8242055

Abstract: Excessive IR-drop during scan shift can cause localized IR-drop around clock buffers and introduce dynamic clock skew. Excessive clock skew at neighboring scan flip-flops results in hold or setup timing violations corrupting test stimuli or test responses during shifting. We introduce a new method to assess the risk of such test data corruption at each scan cycle and flip-flop. The most likely cases of test data corruption are mitigated in a non-intrusive way by selective test data manipulation and masking of affected responses. Evaluation results show the computational feasibility of our method for large benchmark circuits, and demonstrate that a few targeted pattern changes provide large potential gains in shift safety and test time with negligible cost in fault coverage.
Analysis and Mitigation of IR-Drop Induced Scan Shift-Errors

Stefan Holst*, Eric Schneider‡, Koshi Kawagoe*, Michael A. Kochte‡, Kohei Miyase*, Hans-Joachim Wunderlich†, Seiji Kajihara* and Xiaoqing Wen*

*Kyushu Institute of Technology, Iizuka, Japan
Email: {holst@kawagoe@aries30.miayase@kajihara@wen}@cse.kyutech.ac.jp
‡University of Stuttgart, Stuttgart, Germany
Email: {schneiec@kochte@wu}@uni-stuttgart.de

Abstract—Excessive IR-drop during scan shift can cause localized IR-drop around clock buffers and introduce dynamic clock skew. Excessive clock skew at neighboring scan flip-flops results in hold or setup timing violations corrupting test stimuli or test responses during shifting. We introduce a new method to assess the risk of such test data corruption at each scan cycle and flip-flop. The most likely cases of test data corruption are mitigated in a non-intrusive way by selective test data manipulation and masking of affected responses. Evaluation results show the computational feasibility of our method for large benchmark circuits, and demonstrate that a few targeted pattern changes provide large potential gains in shift safety and test time with negligible cost in fault coverage.

I. INTRODUCTION

During scan-test, shift operations cause abnormally high switching activity in the logic circuit under test [1–4]. Excessive switching activity in localized areas can cause IR-drop that in turn affects clock distribution by increasing the delay of some clock buffers that constitute the clock tree [5]. The delay impact of IR-drop gets stronger as the nominal supply voltage scales down further in modern technologies [6].

Test data corruption arises due to excessive clock skew between neighboring scan flip-flops. If the clock arrives at a scan flip-flop too early or too late, the setup or hold time requirements may be violated resulting in undefined behavior or the wrong value being captured: If the clock arrives at a scan flip-flop $f_i$ much later than at its immediate predecessor in the scan chain $f_{i-1}$, metastability or a hold-time violation occurs where $f_i$ may actually capture the value of $f_{i-2}$. If the clock arrives too late at $f_{i-1}$, metastability or a setup-time violation occurs where $f_i$ may capture the previous value of $f_{i-1}$ a second time.

Figure 1 illustrates the chain of events that leads to test data corruption. The left-hand side shows a design with a scan chain containing three scan cells and the clock tree. The combinational logic gates placed near clock buffers are called aggressors. Two particular sets of aggressors are shown in the figure, which we call aggressor regions $A$ and $B$. In the beginning, the scan chain is loaded with the pattern 101. The first shift clock causes every flip-flop to change its value, resulting in the scan state 010. The scan state change 101 → 010 now propagates through the combinational circuit causing switching activity. In Case 1, the switching activity in aggressor region $B$ is assumed to be much higher than the activity in aggressor region $A$. This causes IR-drop and an additional delay on the clock path to flip-flop $f_i$ in the second shift cycle and a hold-time violation. In Case 2, there is excessive switching activity in aggressor region $A$ causing additional clock delay to $f_{i-2}$ and $f_{i-1}$ in the second shift cycle. This results in a setup-time violation and possible test data corruption in the third shift cycle. Hold-time violations are more likely to occur when the path between $f_{i-1}$ and $f_i$ is rather short, setup-time violations are more likely when the slack at $f_i$ is small.

Clock skew during functional operation is usually tightly controlled during clock tree synthesis [7] and by integrating post-silicon tunable clock buffers [8, 9] in the design. In scan test, however, clock skew problems remain, because (1) excessive switching activity may occur in the circuit and (2) the scan chain may have to connect flip-flops that are driven...
by different branches of the clock tree.

IR-drop-induced timing problems are closely related to the shift clock speed. If the shift speed is slow enough, the voltage levels can recover in time before the next clock cycle eliminating the skew problems. To shorten test time and therefore cut test cost, however, shift speeds are usually chosen to apply the patterns as fast as possible and without test data corruption. A reduction of worst-case IR-drop induced clock skew easily translates into faster possible shift clocking and thus shorter test time.

Applicable previous works can be broadly classified into approaches that adjust the timing and into approaches that aim to reduce shift power. Whenever a scan cell is prone to a hold-time violation, hold-time fixing buffers can be added between the two affected scan flip-flops [10], or scan cells with relaxed hold-time requirements can be used [11, 12]. Probable setup-time violations can also be handled by changing the design slightly to counter the clock skew or speed up the scan path segment. Another way is to synthesize the clock tree after scan insertion and optimize the clock distribution for both functional mode and test mode together [13, 14]. All of these methods require intrusive changes in the design or design libraries which are either costly or even impractical especially in System-on-Chip designs that combine cores from various vendors. These methods usually also require additional area or increase power consumption during shift operations even more. Furthermore, our simulations have shown that compared to the huge amount of data that is shifted through the flip-flops, only very few cases are actually prone to timing violations due to clock skew. Therefore, we propose fixing just these few cases by changing the test pattern set and pinpointedly masking the affected response bits without any costly global adjustments. Our simulations have also shown that a single scan cell can be prone to both setup- and hold-time violations depending on the shift cycle. Therefore, fixing either of them by changing the timing in the design won’t help.

The other class of approaches aim to reduce power consumption during shift. They are applicable here as they may help to reduce excessive IR-drop and therefore power-related timing problems. Many low-power testing techniques like output gating logic [15], partial scan [16], or power-aware test de-compression [17] are available. The primary target of them is average power reduction to control heat and not the instantaneous effects that lead to excessive IR-drop [18]. Although they might help accidentally in reducing shift errors, they cannot guarantee good results. The scan segmentation technique in [19] explicitly handles the impact on clock trees but requires new scan chain design and more complex clock control.

The approaches in [20, 21] target peak shift power by identifying problematic test vectors and change them to reduce the number of transitions in the scan chain. Similar test pattern modification techniques have been shown to help with capture power as well [22]. Several techniques use multiple staggered clocks to reduce peak power consumption and IR-drop [23–25]. Such approaches require multiple clocks and are not applicable if flip-flops in the problematic area share a single clock tree. Selectively disabling scan chains as proposed in [26] or adding additional masking logic to each scan flip-flop such that shift switching activity does not propagate into the combinational circuit [15] can be applied to reduce overall power dissipation.

None of the approaches mentioned above target IR-drop induced clock-skew for scan shift. An application of them to the problem at hand would either lead to an over-designed solution or no solution at all when some skew issues remain.

In this work, we propose a simulation-based clock skew analysis method with sufficient resolution to precisely identify shift clock skew issues. Our approach performs a full timing simulation of all shift cycles in a matter of a few hours for the largest ITC’99 benchmark circuits by using an extremely efficient GPU-based simulation approach. Furthermore, we demonstrate a new approach to mitigate the potential test data corruption that would arise from these timing issues with minimal test pattern changes and minimal masking. Our approach does not require any design changes and can be readily combined with all other aforementioned low-power test approaches.

We will first discuss briefly the relationship between switching activity, IR-drop and clock skew that form the foundation of our model (Section II). We then describe our simulation based analysis approach to find the most likely candidates for timing violations during a test in Section III. In Section IV, we present our mitigation approach that guarantees to avoid test pattern corruption. The experimental evaluation of our approach in Section V shows performance results of the analysis method as well as the impact of our mitigation techniques on fault coverage and test time.

II. CLOCK SKEW MODELING

In general, high localized switching activity causes IR-drop in the affected region and the reduced effective supply voltage of the affected gates in turn increases their delay. The amount of IR-drop and other power supply noise at each point in the layout depends on the transient behavior of the individual standard cells, the power delivery network design, its decoupling capacities and parasitics. The delay of each cell at the various effective supply voltages again depends on its input slew, its load, the effective supply voltages of its driving and receiving cells and all involved parasitics. Finally, the delay changes in the circuit couples back to the switching activity and its distribution over time within a clock cycle. Clearly, electrical level SPICE simulations model all of these effects precisely. It is also obvious that such electrical level simulations are computationally too expensive to simulate a complete design for all shift cycles with reasonable resources.

Various models have been proposed to estimate regional IR-drop from switching activity without expensive electrical level simulations [27, 28]. The relation between IR-drop and path delay increase has shown to be linear [29]. Introducing a new quantitative model is beyond the scope of this work and also unnecessary. Our aim is to generate a ranking of
the most likely timing violations, so a relative metric that can
determine this sorting of likely timing violations is sufficient.
The following model captures the basic relations between
regional switching activity, IR-drop and delay change without
necessarily giving accurate absolute delay values.

Let \( c \in C \) be a cell in the design and let \( b \) be a clock
buffer. Let \( \text{wsa}(c, j) \rightarrow \mathbb{R}^+ \) be a measure of the weighted
switching activity of the cell \( c \) in shift cycle \( j \). This number
is usually calculated as the number of toggles of the cell
\( \text{tog}(c, j) \) multiplied by a weight that corresponds to the power
demand of the cell per toggle: \( \text{wsa}(c, j) = \text{tog}(c, j) \cdot w(c) \).
For each pair of cell \( c \) and clock buffer \( b \), we define an influence
coefficient \( s(b, c) \rightarrow \mathbb{R}^+ \) that models the strength of the impact
of the activity at \( c \) on the delay of \( b \). This influence factor
can be calculated based on the proximity of \( b \) and \( c \) in the
layout, the proximity of their connections to the power delivery
network and the inner resistance of the power delivery network
at these points.

With \( B(f_i) \) being the set of clock buffers between the
common clock source and the clock input of the scan cell
\( f_i \), the impact of the surrounding switching activity on the
clock path in shift cycle \( j \) is calculated as:

\[
\text{imp}(i, j) = \sum_{c \in C} \sum_{b \in B(f_i)} s(b, c) \cdot \text{wsa}(c, j)
\]

The higher the value \( \text{imp}(i, j) \), the more additional delay is
expected at the clock input of \( f_i \) at shift cycle \( j \).

The relative amount of clock skew between the flip-flop \( f_i \)
and its immediate predecessor in the scan chain can now be
estimated with:

\[
\text{skew}(i, j) = \text{imp}(i, j) - \text{imp}(i - 1, j)
\]

If \( \text{skew}(i, j) \) is a large positive number, the probability of a
hold-time violation is high. If \( \text{skew}(i, j) \) is a large negative
number, the probability of a setup-time violation is high. If
\( \text{skew}(i, j) \) is near zero, the switching activities around both
clock paths are well balanced and the clock will arrive almost
at the same time at \( f_i \) and \( f_{i-1} \). Note that any impact on clock
buffers that are shared between the flip-flops \( f_i \) and \( f_{i-1} \) does
not change the value of \( \text{skew}(i, j) \).

III. Shift error analysis

Shift error analysis takes as input the design data and a
test pattern set. The design data consists of a gate-level
netlist, placement of all cells in the layout, gate-level timing
information after placement, and relevant design-for-test
information such as scan-chain organization. The test pattern
set can be generated by either a standard ATPG tool or an
on-chip pseudo-random pattern generator in case of a built-in
self-test application.

With each shift clock, each scan-cell updates its own value
with the value of its predecessor in the scan chain. The goal is
to find those scan-cell update events with the highest likelihood
of a hold-time or a setup-time violation. For a scan chain of
length \( l \), \( l \) update events occur for each shift cycle. If we
assume without loss of generality a single scan chain in the
circuit, we have in total \( t = l \cdot (p + 1) \) shift cycles for a
complete application of \( p \) test patterns. The number of update
events \( e = l \cdot t = l^2 \cdot (p + 1) \) grows quadratically with the
design size, so a very scalable analysis approach is necessary
to filter the few potential problematic cases out of this large
number of update events.

The overall flow of the shift error analysis is as follows.

1) \textbf{Aggressor Set Extraction}: Identify for each scan cell
the relevant aggressor cells whose switching activity
potentially affects the clock skew between the scan cell
and its predecessor in the chain.

2) \textbf{Scan State Expansion}: Expand the test pattern set to
a list of scan states for each individual scan clock cycle.

3) \textbf{Gate-Level Power Simulation}: Simulate each transition
from one scan state to the next and record the switching
activities of all aggressor cells.

4) \textbf{Simulation Result Aggregation}: Generate a list of
update cases ordered by the amount of switching activity
of their respective aggressing cells.

The details are given in the following.

A. \textbf{Aggressor Set Extraction}

In this step, the circuit structure and placement information
is analyzed to calculate all necessary weights \( w(c) \) and influence
coefficients \( s(b, c) \). The scan chains are traced to obtain
the position of each scan flip-flop in the chain, then the clock
tree is traced to obtain the set of clock buffers for each flip-flop.
The placement information is used to calculate \( s(b, c) \) for
each encountered clock buffer \( b \) and potential aggressor cell
\( c \in C \), and for each \( c \) with some non-zero \( s(b, c) \), the weight
\( w(c) \) is obtained. Again, the actual values of the coefficients
\( w(c) \) and \( s(b, c) \) are determined by the IR-drop models used.

In a final step, a sparse matrix \( \text{weight}(i, c) \) is generated that
contains the coefficients to each cell \( c \in C \) for a flip-flop pair
at each scan chain position \( i \). As the linearity of the model
allows us to write:

\[
\text{skew}(i, j) = \sum_{c \in C} \text{tog}(c, j) \cdot \text{weight}(i, c),
\]

this matrix of static weights is just

\[
\text{weight}(i, c) = w(c) \cdot \left( \sum_{b \in B(f_i)} s(b, c) - \sum_{b \in B(f_{i-1})} s(b, c) \right)
\]

The matrix of static weights only depends on the design
structure and the used IR-drop model and is independent
from the actual patterns and shift states. It is calculated once
during simulation setup and then used throughout the actual
simulation for the skew estimations. The aggressor set \( A \cup B \)
shown in Figure 1 are all cells \( c \) with \( \text{weight}(i, c) \neq 0 \).

B. \textbf{Scan State Expansion}

This step determines the state of the scan chains for each
shift cycle \( j \) from a given test pattern set and the scan structure
extracted from the design. This is a rather simple calculation
that involves shifting the test patterns and the responses to
the appropriate places in the chains. Each pair of consecutive scan states is then combined into a set of input waveforms that contain for each scan cell a static value (if the scan cell does not change during that shift cycle) or a transition (if the scan cell changes its value). For each test a large amount of scan states is generated. The calculation of the scan states is rather easy and can be performed on demand in parallel to the actual simulation to eliminate the need for storing all the states in memory.

C. Gate-Level Power Simulation

The waveforms from the scan-state expanded test set is simulated with a modified version of the massively parallel GPU-based timing simulator published in [30, 31].

The simulator operates on combinational gate-level netlists. The timing information for each cell is loaded from the SDF (Standard Delay Format) file generated by physical synthesis or parasitics extraction tools. The simulator supports an industry-standard pin-to-pin delay model including short pulse filtering.

The combinational gate-level netlist is first topologically ordered and then uploaded to GPU memory. Figure 2 shows how a set of waveforms is propagated through one topological level of the circuit with multiple dimensions of data-parallelism. The high degree of parallelism allows timing simulation with extremely high throughput.

![Waveform Propagation](image)

**Fig. 2.** Waveform propagation through one topological level of a combinational circuit with two dimensions of parallelism.

The needed modifications to the simulator are similar to the modifications previously done in [32] for IR-drop estimation for launch switching activity. In addition, after each simulation, the relevant toggle data is retrieved from the GPU and skew\((i, j)\) is calculated as the weighted sum of these toggles. While the previous work needed to calculate regional IR-drop for all circuit regions, here we only consider the regions near certain clock buffers \(s(b, c) > 0\).

D. Simulation Result Aggregation

As mentioned before, the number of scan-cell update events \(e(i, j)\) grows quadratically with the circuit size. Even for rather small circuits like b17, the number of update events is more than two billion. Just storing the shift cycle \(i\), the scan chain position \(j\) and the skew measure itself for all update events would take several gigabytes of memory. However, given that in general DFT insertion is performed with reasonable timing margins, the vast majority of these update events will not be problematic. The update events are sorted by \(\text{skew}(i, j)\) to get a ranked list of events most prone to shift-errors due to hold-time violations \((\text{skew}(i, j) > 0)\) or setup-time violations \((\text{skew}(i, j) < 0)\). The update events with a near-zero \(\text{skew}(i, j)\) can easily be removed from this list without affecting the final result. Our experiments show that compared to the total number of update events only very few relevant cases actually need to be stored.

A final analysis determines for all worst-skew update events whether an actual timing violation would lead to a test data corruption. If there is a high chance of a hold-time violation \((\text{skew}(i, j) > 0)\), flip-flops \(f_{i-2}\) and \(f_{i-1}\) must have different values before shift cycle \(j\) for a test data corruption to occur. If \(f_{i-2}\) and \(f_{i-1}\) hold the same value, the shift-in signal of \(f_i\) remains stable over the whole shift cycle and the test data cannot be corrupted in such a case despite the excessive clock skew. Similarly, if there is a high chance of a setup-time violation \((\text{skew}(i, j) < 0)\), flip-flops \(f_{i-2}\) and \(f_{i-1}\) must have different values before shift cycle \(j - 1\) for a test data corruption to occur. If these conditions are not met, the update event is flagged as inert, because it does not lead to test data corruption in case of a timing violation. Consequently, these cases do not have to be mitigated.

IV. Shift-Error Mitigation

The goal of shift-error mitigation is to avoid all test data corruption that would arise from update events with a skew outside a given margin. Unlike more general low-power testing approaches discussed earlier, this approach is much less intrusive and guarantees that all IR-drop related test data corruption is either avoided or masked.

As shown in the previous section, test data corruptions arise only when the scan-in pin of a flip-flop \(f_i\) changes. The first mitigation approach taken here is to change the original test patterns in order to keep the scan-in of a flip-flop \(f_i\) stable in all scan cycles \(j\) where \(\text{skew}(i, j)\) is outside the given margin. For mitigating a potential hold-time violation, the contents of flip-flop \(f_{i-2}\) or \(f_{i-1}\) at scan cycle \(j\) have to be equal. For mitigating a potential setup-time violation, the contents of flip-flop \(f_{i-2}\) or \(f_{i-1}\) at scan cycle \(j - 1\) have to be equal. Whenever at least \(f_{i-2}\) at the appropriate shift cycle contains a bit of a test pattern and not a bit of a test response, this is easily accomplished by flipping the appropriate bit in the original test set. Note that at all times during shifting, test stimulus data always precedes the test responses, so all potential test data corruptions in the first part of the scan chain can be handled in this way. If both \(f_{i-2}\) and \(f_{i-1}\) contain response data, this kind of low-impact mitigation is not possible. In this case, our approach falls back to masking the appropriate response bit. Every potential shift-error is mitigated by either flipping a single bit in the test set or by masking a single bit in the test response.

Whenever the test pattern is changed, the switching activity in the circuit may change as well and there is some potential
of different shift-errors to occur. Therefore, repeated analysis and mitigation steps are necessary as shown in Figure 3. First, shift-error analysis is performed on the complete test set. Then, all potential shift-errors that can be fixed by changing the test set itself are handled. If some patterns were changed, all shift cycles relevant to a changed test pattern or a response to a changed test pattern are re-analyzed. This loop continues until no more potential shift-errors remain that can be handled without response masking. The resulting test pattern set will only show shift-errors in test responses. In a final step, masking information is generated that prevents the effects of these shift errors to lead to a false test and the resulting new test pattern set is fault simulated to evaluate the change in fault coverage.

The transition fault ATPG patterns were expanded and clock skew analysis was performed. First, we show the necessity of a full timing simulation based analysis for reasonable regional switching activity estimates. Figure 4 shows on the x-axis the estimated clock skew calculated with full timing (including static and dynamic hazards) and on the y-axis the estimated clock skew calculated using zero-delay logic simulation (without hazard information). The color at each point shows the number of scan-cell update cases. To improve the color scale the point (0, 0) has been omitted from the graph. As can be seen, the cases with extreme clock skew are not well correlated. In fact, some of the cases that show highest skew in timing simulation have almost no skew in zero-delay

![Fig. 3. Overall flow of shift-error mitigation.](image-url)

If scan-compression or on-chip pattern generation is used, it becomes harder to change single bits in patterns. Several compression schemes such as bit-flipping BIST [33] or programmable deterministic BIST [34, 35] allow more fine-grained control over pattern generation and could easily handle the additional bit-flip information generated by our method. Masking logic is usually already present in scan-compression to avoid contamination of signatures by non-deterministic scan-out data from multi-cycle paths or uninitialized memory. The original mask can easily be augmented by the additional masking information generated by our method.

V. EXPERIMENTAL EVALUATION

We conducted a series of experiments on the largest ITC’99 benchmark circuits. Using a standard commercial tool flow, the circuits were synthesized and a single scan chain was inserted. Physical synthesis was performed on these circuits to generate layouts with the Synopsys’ Open EDK 90nm [36] standard cell library. The post-synthesis delay information was saved in the standard delay format (SDF) for timing simulation.

The design data is loaded into our simulation system, which starts by calculating the static weights \( w(i, c) \) for each scan chain position \( i \) and each cell \( c \). Since our primary concern is the validation of our simulation and mitigation method, we used the following simple model for calculating the coefficients \( w(c) \) and \( s(b, c) \) from the given circuit and layout data. The weight \( w(c) \) of a cell was set to the number of fanouts of the cell, i.e. if a cell \( c \) drives \( z \) other cells, then \( w(c) = z \). The coefficients \( s(b, c) \) depend on the standard cell layout. Let \( d \) be the width of the NAND2X1 cell in the library. Let \( y \) be a row number and \( x \) be a location of a cell in a row in units of \( d \). For instance, the locations of two neighboring NAND2X1 cells on a row \( y \) are \((x, y)\) and \((x+1, y)\). Let the location of a clock buffer \( b \) be \((x_b, y_b)\) and the location of a cell \( c \) be \((x_c, y_c)\). We set \( s(b, c) = 1 \) (full strength), if \( x_b - 2 \leq x_c \leq x_b + 2 \) and \( y_b - 1 \leq y_c \leq y_b + 1 \), and \( s(b, c) = 0 \) (no influence) otherwise.

Table I shows basic design statistics of the considered benchmarks. The first four columns show the benchmark name, the number of combinational gates \(|C|\), the number of transition delay fault test patterns obtained from ATPG and the number of flip-flops in the design. The column Clock Tree – Depth shows the maximum number of clock buffers between the clock input and any flip-flop, and column Clock Tree – Buffers shows the overall number of clock buffers \(|B|\) in the tree. Column Aggressors shows the number of logic gates that influence at least one clock buffer: \(|\{c \in C : \exists b \in B \text{ with } s(b, c) > 0\}|\).

<table>
<thead>
<tr>
<th>Design</th>
<th>Gates</th>
<th>Patterns</th>
<th>Flip-Flops</th>
<th>Clock Tree Depth</th>
<th>Clock Tree Buffers</th>
<th>Aggressors</th>
</tr>
</thead>
<tbody>
<tr>
<td>b14</td>
<td>3609</td>
<td>406</td>
<td>215</td>
<td>2</td>
<td>20</td>
<td>184</td>
</tr>
<tr>
<td>b15</td>
<td>6788</td>
<td>978</td>
<td>416</td>
<td>2</td>
<td>41</td>
<td>390</td>
</tr>
<tr>
<td>b17</td>
<td>21028</td>
<td>1198</td>
<td>1314</td>
<td>3</td>
<td>138</td>
<td>1234</td>
</tr>
<tr>
<td>b18</td>
<td>51977</td>
<td>1563</td>
<td>2762</td>
<td>5</td>
<td>291</td>
<td>2695</td>
</tr>
<tr>
<td>b19</td>
<td>75525</td>
<td>1968</td>
<td>4375</td>
<td>5</td>
<td>454</td>
<td>3847</td>
</tr>
<tr>
<td>b20</td>
<td>8427</td>
<td>476</td>
<td>429</td>
<td>2</td>
<td>44</td>
<td>358</td>
</tr>
<tr>
<td>b21</td>
<td>8184</td>
<td>509</td>
<td>429</td>
<td>2</td>
<td>45</td>
<td>374</td>
</tr>
<tr>
<td>b22</td>
<td>12972</td>
<td>893</td>
<td>611</td>
<td>3</td>
<td>65</td>
<td>634</td>
</tr>
</tbody>
</table>

The transition fault ATPG patterns were expanded and clock skew analysis was performed. First, we show the necessity of a full timing simulation based analysis for reasonable regional switching activity estimates. Figure 4 shows on the x-axis the estimated clock skew calculated with full timing (including static and dynamic hazards) and on the y-axis the estimated clock skew calculated using zero-delay logic simulation (without hazard information). The color at each point shows the number of scan-cell update cases. To improve the color scale the point (0, 0) has been omitted from the graph. As can be seen, the cases with extreme clock skew are not well correlated. In fact, some of the cases that show highest skew in timing simulation have almost no skew in zero-delay.
logic simulation. While zero-delay logic simulation data may give some good indications for average weighted switching activity estimations over many clock cycles, our data shows that for instantaneous IR-drop and clock skew estimations, simulation of all static and dynamic hazards in the circuit is absolutely necessary. The same was observed for the other benchmarks as well and from now on, we only report results based on full timing simulation.

FIG. 4. Correlation between timing simulation and logic simulation in clock skew estimation for the circuit b17.

Table II shows the runtime of the simulation and some general results. The second column shows the number of shift cycles for applying all test patterns. The runtime of the simulation is shown in Column 3. It can be seen that full timing simulation for all shift cycles only takes a couple of seconds up to a few hours for the biggest benchmark circuits on a single GPU. This level of performance can only be achieved with a GPU-accelerated timing simulator that is 100X...1000X faster than traditional event-based timing simulators [30]. Column max. |skew| shows the maximum |skew(i, j)| encountered in the complete test. Column Update Events with |skew| ≥ 0 shows all the scan-cell update cases where the flip-flop changes its value. Subcolumn |skew| ≥ 50% shows the number of update cases among them with at least some skew, and subcolumn |skew| ≥ 50% shows the update cases among them with a skew equal or larger than half of the maximum skew in the test. It can be seen that even though the total number of cases is extremely large, the number of updates with high clock skew is fairly low. As the maximum observed clock skew is a major factor in choosing the fastest possible shift clock speed, fixing just these few extreme clock skew cases can translate to a major improvements in test time.

Now, we take b17 as an example to explore the relation between allowable skew and the amount of potential shift errors in more detail. Figure 5 shows on the x-axis the maximum allowable skew in both positive (tolerance against hold-time violations) and negative (tolerance against setup-time violations) direction. The y-axis shows the percentage of update cases that reach the allowable skew or exceed it. Note that the y-axis is in logarithmic scale. Less than 20% of all update cases show some amount of clock skew (less than 10% in positive direction plus less than 10% in negative direction). As the allowable skew increases, the number of potential shift errors decreases roughly exponentially. With an allowable skew of half of the maximum observed skew, only about 0.01% of all scan-cell update cases are prone to shift errors.

FIG. 5. Relation between allowable clock skew and amount of potential shift errors (100% = 1.64B cases) for the circuit b17.

Figure 6 shows for each location in the scan chain of b17 the amount of potential shift errors. Again, we assume a maximum allowable clock skew of 50% of the maximum observed skew. A mark in the lower part of this figure shows the presence of at
least one timing violation at that position. The bars above the marks show the number of timing violations during the test. As mentioned before we observe that only very few locations are prone to shift errors at all. Furthermore, the vast majority of timing violations tend again to cluster at a very few locations among them while most shift error locations show only few cases of excessive clock skew. The same observation holds true in all the other benchmark circuits.

![Fig. 6. Number of potential shift errors (|skew| ≥ 50%) at each scan chain position classified by error type for the circuit b17.](image)

We then eliminated the potential shift errors by manipulating the test set and by masking. After that we performed fault simulation on the new pattern set and masking information and measured the cost in terms of fault coverage. Table III shows in the first three columns the name of the design, the number of faults and the fault coverage of the base test set generated by the ATPG tool, respectively. We chose maximum allowable skews from 50% to 90% of the maximum in 10% increments. The respective skew target is shown in Column 4. Column 5 shows the number of potential shift errors fixed by flipping bits in the test pattern set. The number of bit flips required to fix all of these potential errors are reported in Column 6. The number of bit flips is often less than the number of fixed errors, because sometimes the same test pattern bit is affected multiple times in different shift cycles. Columns 7 and 8 show the number of potential shift errors in the test responses and the required number of bit masks to hide them, respectively. As expected, the number of masked bits is slightly less than the number of shift errors fixed.

The last column shows the resulting fault coverage with the new test sets. Comparable fault coverages can be observed for skew targets of 90% and 80%, and a slight decline of 0.6% in average for a skew target of 70%. This slight 0.6% reduction in coverage is, while expected, still surprisingly small compared to the substantial potential gain of 30% in shift speed. Since shift speed dominates the overall test time, about 30% more patterns could be applied without increasing test time. This gives plenty of room to restore the original fault coverage with a few additional ATPG patterns, if desired.

### VI. Conclusions

We have proposed a new simulation-based method to assess the risk of IR-drop induced test data corruption at each scan cycle and flip-flop. Our highly parallel GPU-based timing simulator allows all shift cycles to be evaluated with full timing accuracy in a few hours for the largest benchmark design. The most likely cases of test data corruption can be mitigated in a non-intrusive way by selective test data manipulation and masking of affected responses. Evaluation results have demonstrated that a few targeted test data changes provide potential gains in shift safety and test time of about 30% with negligible cost in fault coverage of below 1%.

### ACKNOWLEDGMENT

This work was jointly supported by JSPS Grant-in-Aid for Scientific Research (B) #25280016 and #17H01716, JSPS Grant-in-Aid for Scientific Research on Innovative Areas #15K12003, DFG under grant WU 245/16-1 (PARSIVAL), JSPS Grant-in-Aid for the Promotions of Bilateral Joint Research Projects (Japan-Germany), and the German Academic Exchange Service DAAD (supported by the BMBF).

### TABLE III

<table>
<thead>
<tr>
<th>Design(1)</th>
<th>Faults(2)</th>
<th>Skew(3)</th>
<th>Shift-In Errors(4)</th>
<th>Shift-Out Errors(5)</th>
<th>New FC(9)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base FC(6)</td>
<td>Target(7)</td>
<td>Fixed(8)</td>
<td>Flips(9)</td>
<td>Fixed(8)</td>
</tr>
<tr>
<td>b14</td>
<td>17336</td>
<td>89.18%</td>
<td>50%</td>
<td>959</td>
<td>933</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>280</td>
<td>279</td>
<td>591</td>
<td>583</td>
</tr>
<tr>
<td></td>
<td>70%</td>
<td>80</td>
<td>80</td>
<td>151</td>
<td>151</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>12</td>
<td>12</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>90%</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>b15</td>
<td>32120</td>
<td>80.84%</td>
<td>50%</td>
<td>14571</td>
<td>13407</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>6122</td>
<td>5608</td>
<td>4049</td>
<td>3650</td>
</tr>
<tr>
<td></td>
<td>70%</td>
<td>4296</td>
<td>3961</td>
<td>2820</td>
<td>2595</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>1239</td>
<td>1133</td>
<td>778</td>
<td>713</td>
</tr>
<tr>
<td></td>
<td>90%</td>
<td>666</td>
<td>620</td>
<td>399</td>
<td>379</td>
</tr>
<tr>
<td>b17</td>
<td>99102</td>
<td>84.11%</td>
<td>50%</td>
<td>66159</td>
<td>62541</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>15899</td>
<td>15704</td>
<td>12665</td>
<td>12491</td>
</tr>
<tr>
<td></td>
<td>70%</td>
<td>5444</td>
<td>5425</td>
<td>3446</td>
<td>3434</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>1280</td>
<td>1280</td>
<td>673</td>
<td>672</td>
</tr>
<tr>
<td></td>
<td>90%</td>
<td>79</td>
<td>79</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>b18</td>
<td>242932</td>
<td>79.20%</td>
<td>50%</td>
<td>51980</td>
<td>49970</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>1561</td>
<td>1472</td>
<td>2878</td>
<td>2801</td>
</tr>
<tr>
<td></td>
<td>70%</td>
<td>223</td>
<td>221</td>
<td>753</td>
<td>749</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>46</td>
<td>46</td>
<td>172</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td>90%</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>b19</td>
<td>354040</td>
<td>82.46%</td>
<td>50%</td>
<td>3719595</td>
<td>6151933</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>1748006</td>
<td>1050101</td>
<td>3122016</td>
<td>1838790</td>
</tr>
<tr>
<td></td>
<td>70%</td>
<td>90107</td>
<td>88198</td>
<td>209229</td>
<td>204066</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>861</td>
<td>860</td>
<td>2246</td>
<td>2245</td>
</tr>
<tr>
<td></td>
<td>90%</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>b20</td>
<td>39456</td>
<td>88.85%</td>
<td>50%</td>
<td>74206</td>
<td>45978</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>6995</td>
<td>6602</td>
<td>6589</td>
<td>6242</td>
</tr>
<tr>
<td></td>
<td>70%</td>
<td>328</td>
<td>318</td>
<td>358</td>
<td>358</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>27</td>
<td>26</td>
<td>41</td>
<td>41</td>
</tr>
<tr>
<td></td>
<td>90%</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>b21</td>
<td>38310</td>
<td>88.23%</td>
<td>50%</td>
<td>3344</td>
<td>3211</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>824</td>
<td>810</td>
<td>3154</td>
<td>3154</td>
</tr>
<tr>
<td></td>
<td>70%</td>
<td>69</td>
<td>69</td>
<td>173</td>
<td>173</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>5</td>
<td>5</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>90%</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>b22</td>
<td>60028</td>
<td>87.92%</td>
<td>50%</td>
<td>11162</td>
<td>10136</td>
</tr>
<tr>
<td></td>
<td>60%</td>
<td>2138</td>
<td>2027</td>
<td>3893</td>
<td>3893</td>
</tr>
<tr>
<td></td>
<td>70%</td>
<td>420</td>
<td>402</td>
<td>487</td>
<td>487</td>
</tr>
<tr>
<td></td>
<td>80%</td>
<td>75</td>
<td>71</td>
<td>56</td>
<td>56</td>
</tr>
<tr>
<td></td>
<td>90%</td>
<td>9</td>
<td>9</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>
REFERENCES


