Effective In-Situ Chip Health Monitoring with Selective Monitor Insertion Along Timing Paths

Hadi Ahmadi Balef\textsuperscript{1}, Hamed Fatemi\textsuperscript{2}, Kees Goossens\textsuperscript{1,3}, José Pineda de Gyvez\textsuperscript{1,2}
\textsuperscript{1}Eindhoven University of Technology, Eindhoven, The Netherlands
\textsuperscript{2}NXP Semiconductors, Eindhoven, The Netherlands
\textsuperscript{3}Topic Embedded Systems, Best, The Netherlands
\{h.ahmadi.balef,k.g.w.goossens,j.pineda.de.gyvez\}@tue.nl,hamed.fatemi@nxp.com

ABSTRACT
In-situ delay monitoring is an advanced technique to monitor the robustness of digital circuits. Conventionally, in-situ delay monitors are inserted at the end-points of timing paths. To reduce the number of monitors and to increase their observability, intermediate points have been considered. In sharp contrast to these works, we propose a low overhead technique where the insertion points are selected along the timing paths such that timing violations can be predicted without false negative detections. With our approach, the number of required monitors is reduced by up to $\sim \times 11$ compared to end-point insertion techniques. The observability to delay degradation is $\sim \times 8$ better with our approach, compared to techniques with straight monitor placement at intermediate points.

KEYWORDS
Reliability, Digital Circuit, CMOS Variability, In-Situ Delay Monitoring

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:
Hadi Ahmadi Balef\textsuperscript{1}, Hamed Fatemi\textsuperscript{2}, Kees Goossens\textsuperscript{1,3}, José Pineda de Gyvez\textsuperscript{1,2}. 2018. Effective In-Situ Chip Health Monitoring with Selective Monitor Insertion Along Timing Paths. In GLSVLSI ’18: 2018 Great Lakes Symposium on VLSI, May 23–25, 2018, Chicago, IL, USA. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3194554.3194563

1 INTRODUCTION
CMOS technology scaling has been the driving force of the semiconductor industry for decades. In the Nano-scale era, however, the increased variability effects can nullify the benefits of the technology scaling by jeopardizing the robustness of designs. A variety of physical phenomena, such as aging effects (e.g. hot carrier injection and bias temperature instability), voltage droops, process variations, etc., are sources of variability. Variability effects on electrical parameters of a circuit are manifested as delay variation of logical gates. Delay variation may cause erroneous capturing of data at the flip-flops, i.e. a timing error happens. If the timing error is not masked properly, a timing failure occurs in the system. Traditionally, integrated circuits (ICs) are over-designed statically to improve their robustness, by designing at worst-case corners. However, the cost of worst-case design can be reduced by chip health monitoring at run-time, and compensating for the variability effects accordingly. Furthermore, chip health monitoring is essential for safety-critical applications in which the robustness of the system must be guaranteed. Chip health monitoring is performed based on physical sensors [1], replica paths [11], or in-situ delay monitors [3, 4, 7, 10]. The physical sensor and the replica path based techniques require calibration and thus have a diagnosis overhead. Besides, those techniques provide less correlation to the variation of the main circuit as they are not truly in the system. On the other hand, in-situ monitoring allows a truly in-system monitoring but the main challenges for this technique are the prohibitive design overheads, limited observability to the variation effect, and design intrusion.

In Fig. 1, an example of a timing path and the possible insertion points of in-situ monitors are illustrated. A timing path, starts from the launch flip-flop, goes through combinational logic gates (see G1 to G6) and ends into a capture flip-flop. Conventionally, in-situ delay monitors are inserted at the end-point of timing paths [4] (see M1). A timing error is detected with end-point monitoring by sensing if the data arrival time is delayed to the next clock cycle. To avoid metastability in the main flip-flops and to avoid the complexity and overheads of error correction [5], guard banding is applied between the detection at the monitors and the actual timing error in the main flip-flops. The guard banding is performed by adding delay margin between the insertion point and the input of monitor (see DE before M2). Inserting the monitors at every flip-flop of the design adds a huge design cost due to the large number of added monitors. Moreover, the detection of timing degradation by the in-situ monitors depends on the excitation of the monitored paths, i.e. the paths which end into the monitors. The monitored paths are not always excited since path excitation is data dependent. Therefore, in-situ monitoring may have limited observability to the
variability effect. In [6], the monitors are inserted at intermediate points along timing paths to reduce the number of monitors. Since the first part of any timing path is excited more than the full path, the observability is increased by inserting the monitors within timing paths. In [6], the unmonitored part of the paths are accounted by adding a pessimistic delay margin between the insertion point and the input of the monitor (see C6 in Fig. 1). Note that the guard banding is still applied by adding another delay margin before the monitor (see DE before M3 in Fig. 1). To reduce the design overhead per monitor, the monitors are therefore preferably inserted almost at the end of the timing paths. In some works, the timing error is predicted within one clock cycle to perform correction at the end of clock cycle by manipulating the clock signal either locally (see e.g.[8, 9]), or globally (see e.g.[7]). However, the excitation of the path up to the monitor insertion point may not propagate to the end of the path due to logic masking, causing performance reduction due to unnecessary clock manipulations. In [8, 9], the temporal middle point of critical timing paths is monitored to check if the increased delay to the insertion point causes a transition after half of the clock signal. The insertion points are identified based on the timing report of all critical paths. To have the list of all critical paths, path-based static timing analysis is required which is not feasible for this purpose due to the complexity of the analysis [2] and the huge list of critical paths to be monitored. Furthermore, in [8, 9] no delay margin is added for guard banding assuming that the delay of monitored and unmonitored parts of the paths degrade in similar fashion.

In this paper, we propose a new in-situ chip health monitoring technique with reduced number of monitors compared to endpoint monitoring and improved observability to delay degradation through selective insertion at intermediate points along timing paths. We add implicit guard banding by inserting the monitors at particular points of the critical paths and reduce design cost per monitor by removing additional delay margins (see M4 in Fig. 1). Furthermore, we employ a novel technique which is based on graph-based static timing analysis to make sure that all critical paths are monitored while the drawbacks of path-based static timing analysis are avoided. To reduce design intrusion, we propose a new implementation flow that does not disturb the timing and layout of the main design during monitor insertion.

The remainder of this paper is organized as follows. In section 2, the proposed in-situ chip health monitoring scheme is discussed. The new technique to insert in-situ delay monitors in the design is explained in section 3. The experimental results of using the proposed technique are provided in section 4. Finally, section 5 concludes the paper.

2 IN-SITU DELAY MONITORING WITHIN TIMING PATHS

The design of in-situ delay monitor is shown in Fig. 2. The monitor detects delay degradation by checking if any late data transition occurs during the second half of the clock period. Therefore, the insertion points for this monitor are the points where the maximum delay of the timing paths up to the insertion point is less than half of clock period, during normal circuit operation. The monitor consists of a latch with active high clock and an XOR gate. The data value at the insertion point is captured by the latch at the negative clock edge. The XOR gate then flags any transition which occurs during the second half of the clock cycle. Delayed data arrival time to the insertion point implies delay degradation in the circuit and is detected by the monitor.

Notice that when the monitor is inserted at the temporal middle point of the timing paths, it does not capture the effect of timing spread on the remaining logical gates which are located after the insertion point in the timing path. To investigate this effect, we performed Monte Carlo simulations on the spice netlist of a critical path of an industrial design. The Monte Carlo simulation is performed to capture the effect of local and global process variations on the delays of gates in the path. In Fig. 3-(a), the scatter plot of delay samples is shown for full path delay $t_{full}$ versus the delay of the path up to the insertion point of the monitor $t_{mon}$. In this figure, $t_{full}$ and $t_{mon}$ are normalized to clock period $t_{clock}$ and $0.5 \times t_{clock}$, respectively. The dashed lines in the figure divide the space into four regions according to the timing checks performed on $t_{mon}$ and $t_{full}$. These checks are at half of the clock cycle for $t_{mon}$, and at the end of the clock cycle for $t_{full}$. Each sample then occurs in one of these regions

- True-Negative region: $t_{full} < t_{clock}$ and $t_{mon} < 0.5 \times t_{clock}$: Therefore the monitor truly detects no timing violation in the design.
- True-Positive region: $t_{full} > t_{clock}$ and $t_{mon} > 0.5 \times t_{clock}$: Therefore the monitor truly detects timing violation in the design.
We define critical timing paths as those paths whose delay is longer without additional design cost, by selecting the insertion points as the constraint for selecting the critical timing paths to be monitored, i.e. the critical paths are all the paths whose delay is more than \( t\text{clock} - 0.5 \times t_{\text{clock}} \). In one approach, the monitor is inserted at the point that the average value of \( t_{\text{mon}} \) is about 0.5\( t\text{clock} \) and 0.55\( t\text{clock} \), respectively. By inserting the monitors at the points which are after the temporal middle point of the timing path (i.e. \( t_{\text{mon}} > 0.5 \times t\text{clock} \)), guard banding is implicitly applied without additional cost of delay margin. Also, note that False-Negative cases are avoided at the cost of more False-Positive ones which are actually acceptable for chip health monitoring. The achieved guard banding is therefore

\[
t_{\text{gb}} = t_{\text{mon}} - 0.5 \times t\text{clock}.
\]

In the next section, a technique is introduced to identify the set of insertion points based on the selected \( t_{\text{mon}} \), such that all critical paths of the circuit are monitored.

## 3 SELECTIVE POINTS FOR IN-SITU MONITORING

We define critical timing paths as those paths whose delay is longer than a specific value. Therefore, the monitors falsely detect no timing violation in the design. The inputs of \( \text{G_STA} \) are therefore the circuit graph and \( V_{\text{start}} \). The timing graph of a digital circuit is a weighted directed graph \( G \). The set of vertices in \( G \) representing the ports and instance pins is \( V(G) \), and the set of directed edges in \( G \) representing the timing arcs is \( E(G) \). The timing arc from vertex \( v_i \in V(G) \) to vertex \( v_j \in V(G) \) is represented by \( e_{ij} \in E(G) \). From an implementation perspective, we employ the Standard Delay Format (SDF) output of the circuit to generate \( G \). The set of adjacent vertices to the vertex \( v_i \in V(G) \) is

\[
\text{Adj}(G, v_i) = \{ v_j \in V(G) | e_{ij} \in E(G) \}.
\]

\( \text{G_STA} \) returns the vector \( D \in \mathbb{R}^{n \times 1} \), where \( n_{\text{mon}} \in \mathbb{R} \cup \{-\infty\} \). The vector \( D = \text{G_STA}(G, V_{\text{start}}) \) contains the maximum delay from the vertices in \( V_{\text{start}} \) to each vertex \( v_i \in V(G) \). The maximum delay to vertex \( v_i \in V(G) \) is represented as \( d_i \in D \). In Algorithm 1, the maximum delays are accumulated iteratively within each pipeline stage, up to the capturing flip-flops. If there is no timing path from any vertex in \( V_{\text{start}} \) to \( v_i \in V(G) \), then \( d_i = -\infty \).

Without loss of generality, we assume an ideal clock tree in this analysis. Note that during the clock tree synthesis, the design tool
In Design with Synthesis monitors

A cut \( C = (V_1, V_2) \) is created based on \( t_{mon} \) and \( D_{ck,pins} \), where
\[
V_1 = \{ v_i \in V(G) | d_i < t_{mon} \},
\]
and
\[
V_2 = \{ v_i \in V(G) | d_i \geq t_{mon} \}.
\]

In (4) and (5), \( d_i \in D_{ck,pins} \) is obtained from (3), and \( t_{mon} \) is the maximum delay to the monitor insertion point within timing path, as was discussed in the previous section. In Fig. 4, an example timing graph and the cut are illustrated. The set of boundary vertices, \( V_{bnd} \), is defined as
\[
V_{bnd} = \{ v_i \in V(G) | v_i \in V_1, v_j \in V_2 \}.
\]

The set of insertion points is
\[
V_{inp} = \{ v_i \in V(G) | v_i \in V_{bnd}, t_i > (t_{clock} - slk_{max}) \}.
\]

where \( d_i \in D_{ck,pins} \) is obtained from (3), and \( D_{bnd,i} \) is obtained from (7). Once the set of insertion points is identified from (8), the monitors are inserted at the corresponding gate pins of all vertices in \( V_{inp} \).

The monitors are conventionally added to the design after the place and route step (see e.g. [6, 7]). However, such an implementation flow perturbs the layout and the timing of the main design [6]. To minimize the perturbation, we propose the implementation flow illustrated in Fig. 5. In the proposed flow, the monitors are added at the synthesis stage with dangling insertion point inputs. Yet, the clock input of the monitors is connected, such that their loading effect is taken into account during clock tree synthesis. The design with unconnected monitors, then goes through synthesis and back-end design steps. The insertion points of the monitors are identified based on the SDF output, when timing closure is achieved after the place and route step. The monitors are then truly inserted into the design by connecting them to the identified insertion points (i.e. \( V_{inp} \)). The insertion is done by generating a TCL script which contains Engineering Change Order (ECO) commands. Then, optimization is performed to compensate for the loading effects of the monitors on the timing closure of the main circuit. The target slack of the latch in monitors is set to \( -t_{gb} \) to preserve guard banding and avoid over-optimization of the circuit during the final optimization.

Hence, in the proposed implementation flow, the number of monitors \( n_{mon} \) is decided at the front end. Since the monitored paths are the critical paths identified by \( slk_{max} \), with a given \( n_{mon} \), we obtain a path coverage in terms of \( slk_{max} \). Intuitively, by increasing \( n_{mon} \), a higher \( slk_{max} \) is possible since more paths are monitored. To compare our results against the end-point monitoring approach, we define \( n_{ff,crt} \) as the number of critical end-points for a given \( slk_{max} \). Therefore, the number of monitors relative to the number of critical end-points, \( n_{mon}/n_{ff,crt} \), represents the effectiveness of the proposed technique compared to end-point monitoring in terms of number of monitors.

4 EXPERIMENTAL RESULTS

The proposed technique is employed for in-situ monitoring in an ARM Cortex M0 processor, implemented in an industrial 40nm technology. Cadence Genus and Cadence Innovus are used for synthesis and back-end design, respectively. The design specifications are provided in Table 1. The benchmark application for netlist simulation and power calculation is an infinite impulse response filter.

In Fig. 6, \( n_{mon}/n_{ff,crt} \) is plotted versus \( t_{mon} \) in the range of 0.5×
\( t_{clock} \) to \( t_{clock} \). As it can be observed, the maximum effectiveness of the technique (i.e. minimum \( n_{mon}/n_{ff,crt} \)) is achieved when \( t_{mon} = 0.7 \times t_{clock} \), independent of \( slk_{max} \). Furthermore, it can be observed that the number of monitors increases with \( slk_{max} \). \( n_{mon}/n_{ff,crt} \) increases. To illustrate the dependency of the number of insertion points on \( slk_{max} \), we fixed \( t_{mon} = 0.7 \times t_{clock} \) and swept \( slk_{max} \) from 5% to 30% of the clock period and show \( n_{mon} \) and \( n_{ff,crt} \) in Fig. 7. Based on these results we can see that, to monitor all paths whose slack is less than 5% of the clock period,
Inserting more monitors in the design allows for a higher slk corresponding to 250MHz.

The effectiveness of the proposed technique in terms of the number of required monitors is ~11X lower compared to the traditional end-point monitoring approach.

The target speed of the design, \( f_{\text{clock}} \), affects the topology of the circuit and its timing graph. To investigate the effect of \( f_{\text{clock}} \) on the effectiveness of the proposed technique, \( f_{\text{clock}} \) is varied from 50MHz to 250MHz. \( n_{\text{mon}} \) and \( n_{\text{ff,crt}} \) are calculated for each new design considering slk\(_{\text{max}}\)=0.1\( \times \)t\(_{\text{clock}}\). Fig. 8 shows \( n_{\text{mon}} \) and \( n_{\text{ff,crt}} \) plotted versus \( f_{\text{clock}} \). The design with \( f_{\text{clock}}=50\text{MHz} \) is so relaxed that no critical path exists for slk\(_{\text{max}}\)=0.1\( \times \)t\(_{\text{clock}}\). Therefore, both \( n_{\text{mon}} \) and \( n_{\text{ff,crt}} \) are zero. The effectiveness of the proposed technique in terms of the number of monitors, is ~8X higher than the traditional end-point monitoring, when the speed target is \( f_{\text{clock}}=100\text{MHz} \) and it reduces at higher frequencies.

Following the implementation flow which was discussed in the previous section, we added the monitors to the design with \( f_{\text{clock}}=200\text{MHz} \). Table 2 shows the overheads for \( n_{\text{mon}}=64 \) and 128 considering \( t_{\text{mon}}=0.7\times t_{\text{clock}} \). Furthermore, slk\(_{\text{max}} \) and the corresponding \( n_{\text{ff,crt}} \) are provided in the Table to indicate the path coverage and the higher overhead of end-point monitoring techniques. Inserting more monitors in the design allows for a higher slk\(_{\text{max}} \) at the expense of more power, area, and speed overhead. In Fig. 9, the histogram of worst case monitor slacks, and of the main design, are shown for the slow corner and for \( t_{\text{mon}}=0.7\times t_{\text{clock}} \). Based on the histogram shown in Fig. 9, and considering (1), \( t_{gb} \) is distributed from ~0.5ns to ~1ns, (i.e. in the range of 10% to 20% of clock period).

Next, a netlist simulation was performed with the annotated timing at the typical corner. The simulation testbench runs the benchmark application on the processor. Delay degradation is modelled by using a scaling factor during the generation of the SDF output used for timing annotation. The number of monitors which are excited (i.e. detection of a late data arrival) is obtained by counting the timing violations happening in the latch of the monitors. In Fig. 10-(a) the number of excited monitors is plotted versus the scaling factor used for SDF generation. No monitor excitation happens up to 1.3X delay degradation. By scaling more the delays, the monitor

---

**Figure 6**: The number of the insertion points of monitors relative to the number of critical flip-flops versus \( t_{\text{mon}} \) ranging from 0.5\( \times t_{\text{clock}} \) to 0.9\( \times t_{\text{clock}} \), for slk\(_{\text{max}} \) from 0.05\( \times t_{\text{clock}} \) to 2.2\( \times t_{\text{clock}} \). The target speed of the design is 200MHz.

**Figure 7**: The effect of changing slk\(_{\text{max}} \) on the number of required monitors \( n_{\text{mon}} \) and the number of critical flip-flops \( n_{\text{ff,crt}} \) for the design with 200MHz target frequency, considering \( t_{\text{mon}}=0.7\times t_{\text{clock}} \).

**Figure 8**: The effect of changing \( f_{\text{clock}} \) on \( n_{\text{mon}}/n_{\text{ff,crt}} \), considering \( t_{\text{mon}}=0.7\times t_{\text{clock}} \) and slk\(_{\text{max}}\)=0.1\( \times \)t\(_{\text{clock}}\).

**Figure 9**: The histogram of the monitor slacks and the slack of the flip-flops with \( t_{\text{mon}}=0.7\times t_{\text{clock}} \).

**Table 2**: The power, area, and speed overheads of adding monitors to the design with \( f_{\text{clock}}=200\text{MHz} \), with the corresponding slk\(_{\text{max}} \) and \( n_{\text{ff,crt}} \), considering 64 and 128 monitors, and \( t_{\text{mon}}=0.7\times t_{\text{clock}} \).

<table>
<thead>
<tr>
<th>( n_{\text{mon}} )</th>
<th>64</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Overhead</td>
<td>7.6%</td>
<td>10.5%</td>
</tr>
<tr>
<td>Area Overhead</td>
<td>9.7%</td>
<td>11.25%</td>
</tr>
<tr>
<td>Speed Overhead</td>
<td>0.5%</td>
<td>1.2%</td>
</tr>
<tr>
<td>slk(_{\text{max}} )</td>
<td>0.07( \times t_{\text{clock}} )</td>
<td>0.12( \times t_{\text{clock}} )</td>
</tr>
<tr>
<td>( n_{\text{ff,crt}} )</td>
<td>518</td>
<td>651</td>
</tr>
</tbody>
</table>
Figure 10: Netlist simulation results with timing annotated at the typical corner. (a) The number of excited monitors versus the delay scaling factor considering $t_{mon} = 0.7 \times t_{clock}$. (b) The effect of $t_{mon}$ selection on the actual guard band, and the total number of excitations.

excitation starts and reaches its maximum value of 33 when the scaling factor is 1.7X. The monitor excitation decreases afterwards due to the fact that the monitor does not detect the late arrival if it crosses the positive clock edge and enters into the next clock cycle (i.e., false negative detection). The actual timing error happens in the main design when the scaling factor of 1.9X is used. The actual guard band based on the simulation is the difference between the scaling factor which causes timing error in the main design, and the lowest scaling factor which causes monitor excitation. The actual guard band is therefore 0.5 based on the results shown in Fig. 10-(a). Furthermore, the excitation rate of the monitors reflects their observability to delay degradation since with more monitor excitation rate, delay degradation is detected faster. Hence, the maximum number of excitations during the simulation indicates the monitor observability and depends on $t_{mon}$ since shorter part of a path is excited more frequently compared to longer part of the path. Therefore, we considered three $t_{mon}$ values of $0.6 \times t_{clock}$, $0.7 \times t_{clock}$, and $0.8 \times t_{clock}$ to investigate more the effect of $t_{mon}$ on the actual guard banding and the observability of monitors. The effect of $t_{mon}$ on the guard band, and the total number of excitations are shown in the bar plots illustrated in Fig. 10-(b). It can be observed that the actual guard band increases when the insertion points of monitors are adjusted with higher $t_{mon}$. On the other hand, more observability to delay degradation is possible by adjusting the monitor insertion points with lower $t_{mon}$. When the monitors are inserted with $t_{mon} = 0.6 \times t_{clock}$, more than ~8X higher observability to delay degradation is obtained compared to the case where $t_{mon} \geq 0.8 \times t_{clock}$. Hence, the insertion points can be adjusted to find the optimal set of insertion points in terms of observability to delay degradation, guard banding, and the number of monitors.

5 CONCLUSIONS

A new in-situ chip health monitoring technique is proposed in which the insertion points are selected at intermediate points along the critical timing paths. It is shown that with effective guard banding, the unmonitored part of the paths are protected such that false negative detections are avoided. The guard banding is implemented without additional overhead per monitor by properly selecting the insertion points. The number of required monitors depends on the number of monitored paths which are selected based on their minimum slacks. With the proposed technique, the number of monitors is reduced by up to ~11X compared to the traditional end-point monitoring techniques, to monitor all timing paths which their minimum slacks are less than 5% of clock period. Furthermore, ~8X more observability to delay degradation is obtained compared to other techniques which insert the monitors at intermediate, but close to end, point of the critical paths. Finally the design intrusion is minimized by adding the monitors at the front-end of design, and connecting the insertion points after the design is placed and routed.

ACKNOWLEDGMENT

This research has been partially supported by NL grant STW ZERO (2016/STW/00157176) and EU grant EECSEL SILENSE (2017/EECS/E0737487).

REFERENCES