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

Hadi Ahmadi Balef[1], Hamed Fatemi[2], Kees Goossens[1,3], José Pineda de Gyvez[1,2]
[1]Eindhoven University of Technology, Eindhoven, The Netherlands
[2]NXP Semiconductors, Eindhoven, The Netherlands
[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 ~11X compared to end-point insertion techniques. The observability to delay degradation is ~8X better with our approach, compared to techniques with straight monitor placement at intermediate points.

KEYWORDS

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

ACM Reference Format:
Hadi Ahmadi Balef[1], Hamed Fatemi[2], Kees Goossens[1,3], José Pineda de Gyvez[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.

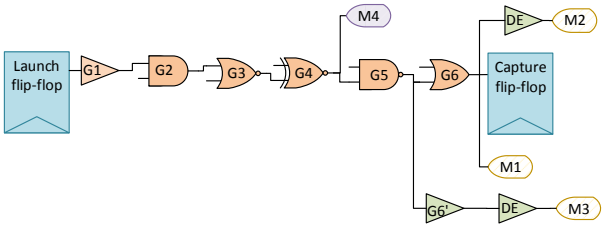

Figure 1: The insertion points of monitors along the timing paths of digital circuits for different in-situ delay monitoring techniques.

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

GLSVLSI '18, May 23–25, 2018, Chicago, IL, USA

ACM ISBN 978-1-4503-5724-1/18/05…$15.00
https://doi.org/10.1145/3194554.3194563

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 G6' 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 end-point 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

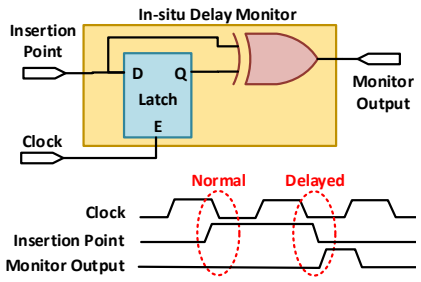

Figure 2: The design of in-situ delay monitor and the corresponding waveforms for the normal and delayed data.

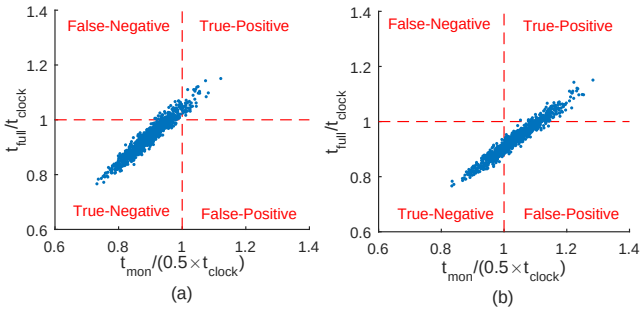

Figure 3: Scatter plot obtained from Monte Carlo simulations of a critical path delay of an industrial design showing t_{full} versus t_{mon} for the cases where (a) t_{mon}= 0.5×t_{clock}, and (b) t_{mon}= 0.55×t_{clock}.

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×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×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×t_{clock}. Therefore the monitor truly detects timing violation in the design.

- False-Negative region: $t_{full} > t_{clock}$ and $t_{mon} < 0.5 \times t_{clock}$. Therefore the monitor falsely detects no timing violation in the design.
- False-Positive region: $t_{full} < t_{clock}$ and $t_{mon} > 0.5 \times t_{clock}$. Therefore the monitor falsely detects timing violation in the design.

True-Negative and True-Positive detection is acceptable, while False-Negative and False-Positive detection should be avoided. False-Negative detection must be avoided, since in this case the delays of the gates after the insertion points increase, but the monitors are falsely showing that the system is robust. On the other hand, False-Positive detection can be acceptable for chip health monitoring since it reflects delay degradation of some gates in the circuit while the full path still has enough margin. In fact, during normal circuit operation, only True-Negative detections are acceptable, and as the circuit degrades, it is preferred to have False-Positive detections before True-Positive ones, to have a guard band between the detection of delay degradation with the monitors and the actual timing violation in the design. Note that this guard banding is implemented without additional design cost, by selecting the insertion points properly.

By inserting the monitor at a point closer to the end-point more samples with $t_{mon} > 0.5 \times t_{clock}$ fall in the False-Positive region. The scatter plots shown in Fig. 3-(a) and (b) are for the cases in which the insertion point is selected so that the average value of t_{mon} is about $0.5 \times t_{clock}$ and $0.55 \times t_{clock}$, respectively. By inserting the monitors at the points which are after the temporal middle point of the timing path (*i.e.* $t_{mon} > 0.5 \times t_{clock}$), 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

$$t_{gb} = t_{mon} - 0.5 \times t_{clock}. \tag{1}$$

In the next section, a technique is introduced to identify the set of insertion points based on the selected t_{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, we define the maximum monitored slack , slk_{max}, 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_{clock}-slk_{max}). In one approach, the monitor insertion points are identified using the list of all critical timing paths. However, this approach is based on the path-based static timing analysis, which is extremely compute-intensive. Fast and feasible timing analysis is performed using the graph-based static timing analysis. In the graph-based static timing analysis, the maximum delay to each circuit node is calculated with summation and maximum operations. Hence, we employ the graph-based static timing analysis (see G_STA in Algorithm 1) to determine the maximum delay to each circuit node, from the nodes in a starting set V_{start}. The inputs of G_STA are therefore the circuit graph and V_{start}.

Algorithm 1 The procedure of graph-based timing analysis to identify the maximum delay to circuit points starting from a set of points V_{start}.

```
1:  procedure G_STA
2:  input: G, V_start
3:      D ← {d_i ∈ R_{-∞} | d_i = 0 if v_i ∈ V_start; else: d_i = -∞}
4:      V_f ← V_start
5:      while V_f ≠ ∅ do
6:          V_fn ← ∅
7:          for v_i ∈ V_f do
8:              V_fn ← V_fn ∪ Adj(G, v_i)
9:              for v_j ∈ Adj(G, v_i) do
10:                 d_j ← max{d_j, d_i + e_ij}
11:             end for
12:         end for
13:         V_f ← V_fn
14:     end while
15:     return D
16: end procedure
```

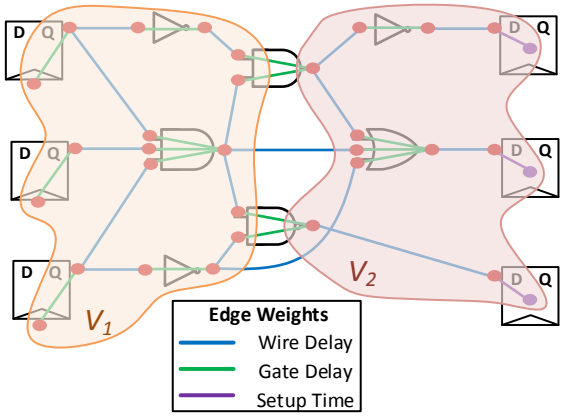

Figure 4: An example circuit graph and a cut $C = (V_1, V_2)$.

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

$$Adj(G, v_i) = \{v_j \in V(G) | e_{ij} \in E(G)\}. \tag{2}$$

G_STA returns the vector $D \in R_{-\infty}^{n \times 1}$, where $R_{-\infty}$=$R \cup \{-\infty\}$. The vector D=G_STA(G,V_{start}) contains the maximum delay from the vertices in V_{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_{start} to $v_i \in V(G)$, then d_i =-∞.

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

attempts to make the delay from the clock ports to the clock pin of all flip-flops the same. Naturally, in a more accurate analysis, clock skew should also be included. To find the set of insertion points, first, the maximum delay vector from the clock pins is calculated as

$$D_{ck,pins} = \text{G\_STA}(G, \{\text{clock pin of all flip-flops}\}). \quad (3)$$

A cut $C = (V_1, V_2)$ is created based on t_{mon} and $D_{ck,pins}$, where

$$V_1 = \{v_i \in V(G) | d_i < t_{mon}\}, \quad (4)$$

and

$$V_2 = \{v_i \in V(G) | d_i \geq t_{mon}\}. \quad (5)$$

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, $V_{bnd} \subset V_1$, is defined as

$$V_{bnd} = \{v_i \in V(G) | e_{ij} \in E(G), v_i \in V_1, v_j \in V_2\}. \quad (6)$$

where V_1 and V_2 are obtained from (4) and (5), respectively. The vertices in V_{bnd} are candidates for placement of the monitors. A screening is performed on V_{bnd} to find the vertices which are in the critical paths, and insert the monitors at those vertices. Therefore, for each $v_i \in V_{bnd}$, G_STA is performed with $V_{start} = \{v_i\}$, and the corresponding maximum delay vector is obtained

$$D_{bnd,i} = \text{G\_STA}(G, \{v_i\}). \quad (7)$$

The set of insertion points is

$$V_{inp} = \{v_i \in V(G) | v_i \in V_{bnd}, t_i > (t_{clock} - slk_{max})\}. \quad (8)$$

where V_{bnd} is obtained from (6), and

$$t_i = d_i + \max\{D_{bnd,i}\}, \quad (9)$$

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.

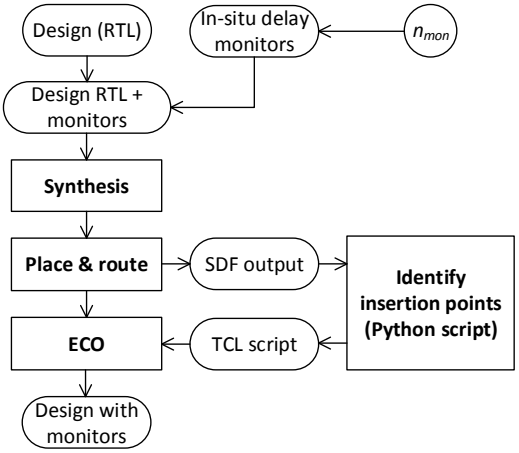

Figure 5: The proposed implementation flow.

Table 1: The implementation details of the benchmark circuit (ARM Cortex M0 processor implemented in 40nm technology).

Target frequency	200MHz
Number of gates	8441
Number of flip-flops	841
Number of IOs	136
Power (mW)	2.24
Area (μm^2)	18877

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}, for slk_{max} = 5%, 10%, 15%, and 20% of the clock period. 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×t_{clock}, independent of slk_{max}. Furthermore, it can be observed that by increasing 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} to 0.7× 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,

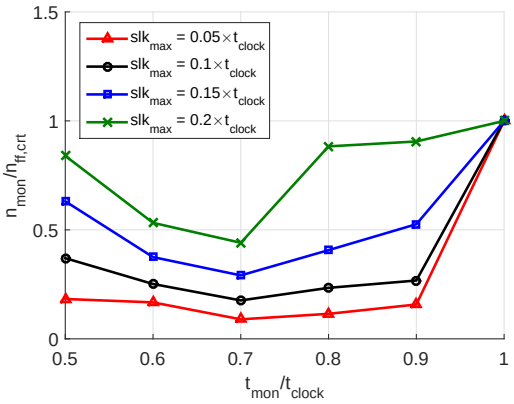

Figure 6: The number of the insertion points of monitors relative to the number of critical flip-flops versus t_{mon} ranging from $0.5\times t_{clock}$ to $0.9\times t_{clock}$, for slk_{max} from $0.05\times t_{clock}$ to $0.2\times t_{clock}$. The target speed of the design is 200MHz.

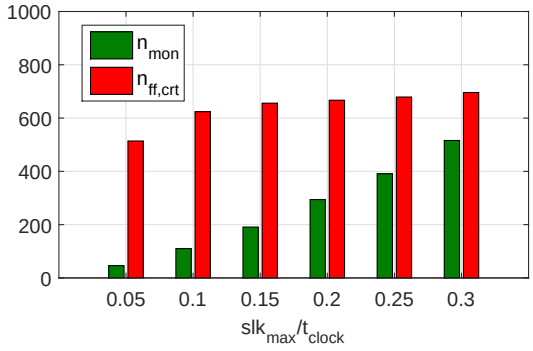

Figure 7: The effect of changing slk_{max} on the number of required monitors $n_{mon}$and the number of critical flip-flops $n_{ff,crt}$for the design with 200MHz target frequency, considering t_{mon}=$0.7\times t_{clock}$.

the number of required monitors is ~11X lower compared to the traditional end-point monitoring approach.

The target speed of the design, f_{clock}, affects the topology of the circuit and its timing graph. To investigate the effect of f_{clock} on the effectiveness of the proposed technique, f_{clock} is varied from 50MHz to 250MHz. n_{mon} and $n_{ff,crt}$ are calculated for each new design considering slk_{max}=$0.1\times t_{clock}$. Fig. 8 shows n_{mon} and $n_{ff,crt}$ plotted versus f_{clock}. The design with f_{clock}=50MHz is so relaxed that no critical path exists for slk_{max}=$0.1\times t_{clock}$. Therefore, both n_{mon} and $n_{ff,crt}$ 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_{clock}=100MHz 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_{clock}=200MHz. Table 2 shows the overheads for n_{mon}=64 and 128 considering t_{mon}=$0.7\times t_{clock}$. Furthermore, slk_{max} and the corresponding $n_{ff,crt}$ 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_{max} at

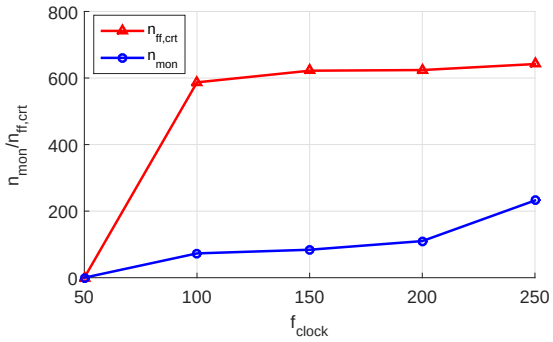

Figure 8: The effect of changing f_{clock}, on $n_{mon}/n_{ff,crt}$, considering t_{mon}=$0.7\times t_{clock}$ and slk_{max}=$0.1\times t_{clock}$.

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

n_{mon}	64	128
Power Overhead	7.6%	10.5%
Area Overhead	9.7%	11.25%
Speed Overhead	0.5%	1.2%
slk_{max}	$0.07\times t_{clock}$	$0.12\times t_{clock}$
$n_{ff,crt}$	518	651

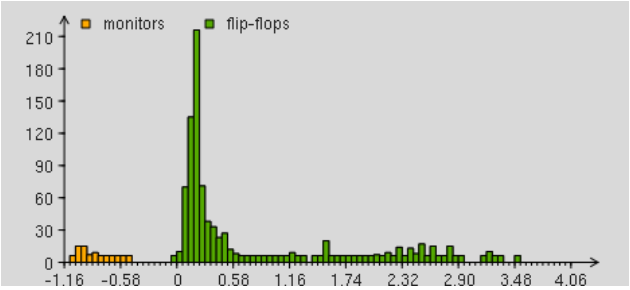

Figure 9: The histogram of the monitor slacks and the slack of the flip-flops with t_{mon}=$0.7\times t_{clock}$.

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_{mon}=$0.7\times t_{clock}$. 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

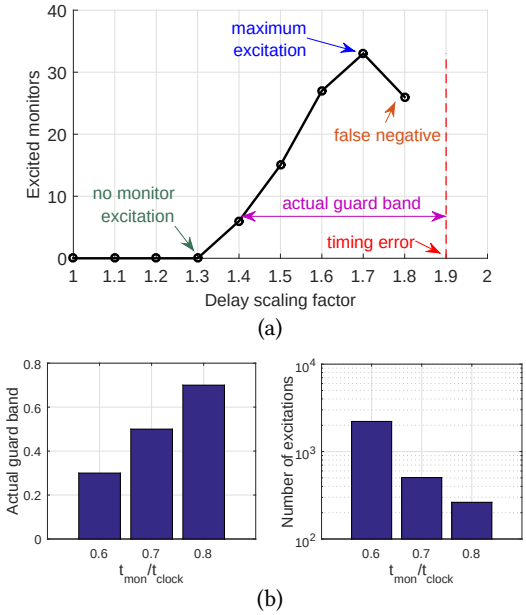

Figure 10: Netlist simulation results with timing annotated at the typical corner. (a) The number of excited monitors versus the delay scaling factor considering t_{mon}=0.7×t_{clock}. (b) The effect of t_{mon} selection on the actual guard band, and the total number of excitations.

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×t_{clock}, 0.7×t_{clock}, and 0.8×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×t_{clock}, more than ~8X higher observability to delay degradation is obtained compared to the case where t_{mon}≥ 0.8×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 ECSEL SILENSE (2017/ECSEL/00737487).

REFERENCES

[1] M. Agarwal, B. C. Paul, M. Zhang, and S. Mitra. 2007. Circuit Failure Prediction and Its Application to Transistor Aging. In *Proceedings of IEEE VLSI Test Symposium (VTS'07)*. 277–286.

[2] H. Cherupalli and J. Sartori. 2015. Graph-based Dynamic Analysis: Efficient Characterization of Dynamic Timing and Activity Distributions. In *Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD'15)*. 729–735.

[3] M. Choudhury, V. Chandra, K. Mohanram, and R. Aitken. 2010. TIMBER: Time Borrowing and Error Relaying for Online Timing Error Resilience. In *Proceedings of Design Automation and Test in Europe Conference and Exhibition (DATE'10)*. 1554–1559.

[4] S. Das, C. Tokunaga, S. Pant, W. H. Ma, S. Kalaiselvan, K. Lai, D. M. Bull, and D. T. Blaauw. 2009. RazorII: In Situ Error Detection and Correction for PVT and SER Tolerance. *IEEE Journal of Solid-State Circuits (JSSC)* 44, 1 (January 2009), 32–48.

[5] D. Fick, N. Liu, Z. Foo, M. Fojtik, J. s. Seo, D. Sylvester, and D. Blaauw. 2010. In Situ Delay-Slack Monitor for High-Performance Processors Using An All-Digital Self-Calibrating 5ps Resolution Time-to-Digital Converter. In *Proceedings of IEEE International Solid-State Circuits Conference - (ISSCC)*. 188–189.

[6] L. Lai, V. Chandra, R. C. Aitken, and P. Gupta. 2014. SlackProbe: A Flexible and Efficient In Situ Timing Slack Monitoring Methodology. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 33, 8 (August 2014), 1168–1179.

[7] S. M. Londono and J. P. de Gyvez. 2012. A Better-Than-Worst-Case Circuit Design Methodology Using Timing-Error Speculation and Frequency Adaptation. In *Proceedings of IEEE International SOC Conference*. 15–20.

[8] V. Mahalingam, N. Ranganathan, and R. Hayman. 2012. Dynamic Clock Stretching for Variation Compensation in VLSI Circuit Design. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 8, 3, Article 16 (Aug. 2012), 16:1–16:13 pages.

[9] M. Nejat, B. Alizadeh, and A. Afzali-Kusha. 2014. Dynamic Flip-Flop Conversion: A Time-Borrowing Method for Performance Improvement of Low-Power Digital Circuits Prone to Variations. *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)* 23, 11 (November 2014), 2724 – 2727.

[10] M. Nicolaidis. 1999. Time Redundancy Based Soft-Error Tolerance to Rescue Nanometer Technologies. In *Proceedings of IEEE VLSI Test Symposium (VTS'99)*. 86–94.

[11] J. Tschanz, K. Bowman, S. Walstra, M. Agostinelli, T. Karnik, and V. De. 2009. Tunable Replica Circuits and Adaptive Voltage-Frequency Techniques for Dynamic Voltage, Temperature, and Aging Variation Tolerance. In *Proceedings of IEEE Symposium on VLSI Circuits (VLSIC'09)*. 112–113.