# Clock Tree Design Considerations to Combat Asymmetric Transistor Aging

1st Freddy Gabbay
*Faculty of Engineering*
*Ruppin Academic Center*
Emek Hefer, Israel
freddyg@ruppin.ac.il

2nd Firas Ramadan
*Faculty of Electrical and Computer Engineering*
*Technion - Israel Institute of Technology*
Haifa, Israel
firasramadan@campus.technion.ac.il

3rd Majd Ganaiem
*Faculty of Electrical and Computer Engineering*
*Technion - Israel Institute of Technology*
Haifa, Israel
majd.ga@campus.technion.ac.il

***Abstract*—Reliability is critical for integrated circuits (ICs) to ensure accurate operation over their lifetime. With the rise of mission-critical systems, the demands imposed by reliability concerns continue to grow. However, recent advancements in semiconductors have revealed that ICs are vulnerable to reliability issues, particularly those stemming from transistor aging. Transistor aging refers to the gradual deterioration of a transistor's performance over time and depends mainly on the bias-temperature instability (BTI). The BTI severely affects IC reliability, degrading performance and causing critical circuit failures due to timing violations. Additionally, asymmetric aging occurs when the degradation is unevenly distributed, intensifying timing violations and reliability concerns. This paper examines how asymmetric transistor aging affects clock tree design and highlights the role of useful skew, clock gates, and asymmetry between clock buffer delays and net delays in amplifying reliability concerns. Furthermore, we propose new design flow guidelines to address asymmetric-aging-related violations in clock trees.**

***Index Terms*—Transistor aging, BTI, Reliability, Clock-tree**

## I. Introduction

In recent decades, VLSI technologies have remarkably advanced in several characteristic ways. First, the continuous development of new process nodes has ensured the consistent miniaturization of transistors to nanometric dimensions, in line with the principles of Moore's law. Second, revolutionary devices and materials have played a pivotal role in driving advancements, resulting in improved performance and reduced energy consumption. However, these advances have also brought exposed the vulnerability of integrated circuits (ICs) to reliability issues, particularly those caused by transistor aging. Transistor aging refers to the decline in a transistor's performance over time and is primarily attributed to the bias-temperature instability (BTI), which will be described in detail in Section 2. The impact of the BTI on IC reliability is significant: it degrades performance and causes critical circuit failures due to timing violations. Moreover, asymmetric aging exacerbates timing violations and amplifies reliability concerns because it results from unevenly distributed degradation.

Given the ongoing advances in VLSI technologies, semiconductors are increasingly being integrated into mission-critical systems, including autonomous vehicles, medical appliances, finance, and security systems [14], [15]. These emerging applications have raised the standards for ICs in terms of resiliency, reliability, and safety, as enforced by regulatory agencies and industry standards [13]. As a result, reliability-aware IC design is now obligatory.

This paper investigates how asymmetric aging affects clock tree design. Clock trees are crucial circuit resources responsible for distributing a balanced clock signal across the chip die. The reliability of the clock tree is vital to IC reliability because even a single clock tree failure point can lead to complete failure of the entire clock distribution network. This paper shows that clock trees are highly vulnerable to reliability issues caused by asymmetric aging. Prior studies [9], [10] mainly focused on how asymmetric aging affects gated clocks, whereas the present study extends the scope of previous work and shows that factors such as useful skews and the asymmetry between net and cell delays can contribute to significant timing violations in the clock tree, ultimately resulting in overall IC failure. Our experimental analysis of a case study examines the clock tree susceptibility to asymmetric aging in General Purpose Graphic Units (GPGPUs) and demonstrates the timing violations that arise when considering asymmetric transistor aging. Additionally, we introduce timing constraints combined with design flow guidelines to mitigate the impact of asymmetric aging on clock trees. The proposed mitigation guidelines are examined by applying an aging-aware timing analysis.

The remainder of this paper is structured as follows: Section 2 provides background and discusses prior works. Section 3 examines the vulnerability of clock trees to asymmetric transistor aging and introduces extended timing constraints. Section 4 presents the experimental results, and Section 5 concludes.

## II. Background and Prior Work

This section gives background information on transistor aging and the BTI and reviews previous studies in the field of transistor aging.

### A. Transistor aging

Transistor aging refers to the deterioration over time of transistors in digital circuits [8], [9] and is caused by the trapping of charge carriers from the transistor inversion channel at the dielectric insulator of the transistor gate. The BTI is recognized as the primary mechanism governing transistor aging. The BTI activates when a constant voltage is applied to the transistor gate, elevating the transistor's threshold voltage. This increase in threshold voltage increases the transistor switching delay and thereby reduces the transistor speed. Asymmetric aging, which denotes the uneven distribution of performance degradation among transistors within an IC, can lead to severe timing issues, including setup and hold timing violations.

The signal probability (SP) is a common technique [9] for assessing the BTI stress profile on logical elements. The SP quantifies the probability of a signal having a logical value of 1 and is defined as the ratio of the time a signal spends in the logical-1 state to the overall time. A decrease in SP intensifies the effect of the BTI, resulting in performance degradation or potentially causing failures in integrated circuits over time.

### *B. Prior work*

Common approaches involve incorporating additional timing margins to mitigate the effects of asymmetric aging. However, this approach often necessitates complex simulation analysis and can lead to overdesign [1]. Other studies [2]–[4] have proposed models for predicting aging degradation and have explored various solutions, including reducing clock cycle time, resizing transistors, VDD tuning, and power gating. Agrawal et al. [6] proposed a method to predict circuit failure by using sensors placed at various locations within the silicon die. Additional research [7] has explored techniques to analyze digital circuits and detect the most vulnerable gates affected by negative BTI (NBTI) stress. This involves using an aging model with BTI-aware libraries and conducting aging-aware timing analysis. Abbas et al. [8] proposed executing anti-aging programs instead of idle tasks during periods of low processor use. Reference [9] proposed an aging-aware microarchitecture to minimize the effects of asymmetric aging on execution units, register files, and memory hierarchy in microprocessors while keeping overhead to a minimum. Reference [10] analyzes asymmetric aging in the clock tree segments of power-efficient designs by using a 45 nm process node. The authors examined how BTI affects the clock tree as a result of clock gates and built-in clock tree skews. However, they did not consider how useful skew and the asymmetry between net and cell delays affect clock tree design. These important factors are extensively addressed in the present study using a 28 nm process node.

## III. Impact of Asymmetric Transistor Aging on Clock Trees

Clock trees are responsible for delivering the clock signal across digital circuits with minimal insertion delay and minimal clock skew between clock tree endpoints. The clock signal is crucial for the correct logical operation of digital circuits. Any failure in the clock tree can potentially manifest itself as an overall circuit failure. Therefore, to ensure reliable operation of the clock signal, setup and hold timing constraints are enforced.

This paper introduces below the three main factors that encourage asymmetric aging in clock trees.

### *A. Clock gating*

Clock gating is a widely accepted approach for achieving dynamic power savings. It entails selectively blocking the clock signal in inactive sections of the circuit, thereby reducing dynamic power consumption. Deactivating the clock in idle circuit areas eliminates unnecessary switching and associated power consumption. Clock gating typically involves using a clock gate cell, which comprises a latch and an AND gate.

Clock gating contributes to asymmetric aging by increasing the idleness of the clock network, as shown in Figs. 1(a) and 1(b). In Fig. 1(a), the clock gate is used in the launch path, increasing aging in this path compared with the capture path. This imbalance can result in setup-timing violations. Conversely, Fig. 1(b) shows that using the clock gate in the capture path accelerates its aging relative to the launch path, leading to hold-timing violations.

### *B. Asymmetry between cell and net delays*

Asymmetric aging in clock networks can also be caused by the disparity in accumulated delay between logical cells and nets. Unlike logical cells, nets are not affected by the BTI. When the launch and capture paths have varying accumulated logical-cell delays, the BTI can induce asymmetric aging, as depicted in Fig. 1(c). An accumulated logical-cell delay in the launch path exceeding that in the capture path can lead to setup-timing violations. Conversely, an accumulated cell delay in the launch path smaller than that in the capture path may result in hold-timing violations. Figure 1(c) illustrates a scenario where both the launch and capture clocks have a balanced 100 ps clock latency. However, the accumulated clock buffer delay in the capture path is 60 ps, whereas the total delay of the clock buffer in the launch path is 40 ps. Even assuming equal aging rates for all clock buffers, the asymmetry between accumulated cell and net delays, combined with the BTI, can cause hold-timing violations due to the delay shift in the capture clock.

### *C. Useful skew*

Useful skew is a common design method in clock tree synthesis, where intentional delays are introduced in clock paths to mitigate setup or hold timing violations. A clock skew inserted on the capture path, as illustrated in Fig. 1(d), allows the design's critical timing path to be extended beyond the clock cycle time. This can only be applied if the timing path has extra positive hold slack to accommodate the delay in the capture clock. In this way, clock skew can either avoid an increase in the clock cycle time or mitigate timing violations. When useful skew is inserted into the launch path, it operates similarly, but this time it helps extend hold margins at the expense of setup.

Useful clock skew increases the susceptibility of clock trees to asymmetric transistor aging, particularly when it is used in the presence of clock gating or when an asymmetry exists between cell and net delays. Clock gating can encourage asymmetric aging of clock skew buffers, potentially contributing to timing violations when considering transistor aging. Additionally, the presence of clock skew buffers induces an intrinsic asymmetry between accumulated net delays and cell delays, which further encourages timing violations in the presence of transistor aging.

### *D. Timing constraints in the presence of asymmetric aging*

The timing constraints for a general synchronous digital circuit, as illustrated in Fig. 2, are defined by Eqs. (1) and (2). The corresponding timing parameters of the circuit are outlined in Table I, assuming a clock cycle time of $T$. The useful skew buffers shown in Fig. 2 are positioned on the capture path with a delay parameter $t_{\text{us}}$. When $t_{\text{us}} > 0$ ($t_{\text{us}} < 0$), the useful skew is applied to the capture (launch) path.

**TABLE I: Timing parameters.**

| Elements | Timing Parameters | | | |
|---|---|---|---|---|
| | ***Propagation delay*** | ***Containment delay*** | ***Setup time*** | ***Hold time*** |
| Launch clock buffers #1 | $t_{\text{CL}}$ | n/a | n/a | n/a |
| Launch clock nets #2 | $t_{\text{NL}}$ | n/a | n/a | n/a |
| Capture clock buffers #3 | $t_{\text{CC}}$ | n/a | n/a | n/a |
| Capture clock nets #4 | $t_{\text{NC}}$ | n/a | n/a | n/a |
| Flip-flops #5 | $t_{\text{pdFF}}$ | $t_{\text{cdFF}}$ | $t_{\text{s}}$ | $t_{\text{h}}$ |
| Combinational circuit #6 | $t_{\text{pdC}}$ | $t_{\text{cdC}}$ | n/a | n/a |
| Useful skew buffers #7 | $t_{\text{us}}$ | n/a | n/a | n/a |

$$\begin{aligned}\Delta slack_{\text{setup}} &= T - t_{\text{pdFF}} - t_{\text{pdC}} - t_{\text{s}} + t_{\text{us}} \\ &\quad +(t_{\text{CC}} + t_{\text{NC}}) - (t_{\text{CL}} + t_{\text{NL}}), \qquad (1)\\ \Delta slack_{\text{hold}} &= t_{\text{cdFF}} + t_{\text{cdC}} + (t_{\text{CL}} + t_{\text{NL}}) \\ &\quad -(t_{\text{CC}} + t_{\text{NC}}) - t_{\text{us}} - t_{\text{h}}. \qquad (2)\end{aligned}$$

Assuming that the launch and capture paths have undergone asymmetric aging due to different clock gating activities (or different SPs), we can represent the derate factors resulting from the BTI as $d_{\text{L}} > 1$ for the launch path and $d_{\text{C}} > 1$ for the capture path. By applying $d_{\text{L}}$ to the launch cells and $d_{\text{C}}$ to the capture cells, as described in Eqs. (1) and (2), we can calculate the setup and hold slacks for an aged circuit using Eqs. (3) and (4), respectively:

$$\begin{aligned}\Delta slack^{\text{aged}}_{\text{setup}} &= T - d_{\text{L}}(t_{\text{pdFF}} + t_{\text{pdC}} + t_{\text{CL}}) - t_{\text{NL}} \\ &\quad + d_{\text{C}}(t_{\text{CC}} + t_{\text{us}}) + t_{\text{NC}} - t_{\text{s}}, \qquad (3)\\ \Delta slack^{\text{aged}}_{\text{hold}} &= d_{\text{L}}(t_{\text{cdFF}} + t_{\text{cdC}} + t_{\text{CL}}) + t_{\text{NL}} \\ &\quad - d_{\text{C}}(t_{\text{CC}} + t_{\text{us}}) - t_{\text{NC}} - t_{\text{h}}. \qquad (4)\end{aligned}$$

Let $\delta_{\text{L}}$ and $\delta_{\text{C}}$ represent the added shift fractions in the launch and capture paths, respectively, such that $\delta_{\text{L}} = d_{\text{L}} - 1$, and let $\delta_{\text{C}} = d_{\text{C}} - 1$. By combining Eq. 1 with Eq. 3 and Eq. 2 with Eq. 4, we can derive the setup and hold slacks for an asymmetrically aged circuit, as described by Eqs. 5 and 6, respectively:

$$\begin{aligned}\Delta slack^{\text{aged}}_{\text{setup}} &= \Delta slack_{\text{setup}} - [\delta_{\text{L}}(t_{\text{pdFF}} + t_{\text{pdC}} + t_{\text{CL}}) \\ &\quad - \delta_{\text{C}}(t_{\text{CC}} + t_{\text{us}})] \qquad (5)\\ &= \Delta slack_{\text{setup}} - \Delta slack^{\text{degradation}}_{\text{setup}}, \\ \Delta slack^{\text{aged}}_{\text{hold}} &= \Delta slack_{\text{hold}} - [\delta_{\text{C}}(t_{\text{CC}} + t_{\text{us}}) \\ &\quad - \delta_{\text{L}}(t_{\text{cdFF}} + t_{\text{cdC}} + t_{\text{CL}})] \qquad (6)\\ &= \Delta slack_{\text{hold}} - \Delta slack^{\text{degradation}}_{\text{hold}}.\end{aligned}$$

The degradation in the setup slack ($\Delta slack^{\text{degradation}}_{\text{setup}}$) and hold slack ($\Delta slack^{\text{degradation}}_{\text{hold}}$) due to asymmetric aging can be expressed by Eqs. 7 and 8, respectively:

$$\begin{aligned}\Delta slack^{\text{degradation}}_{\text{setup}} &= \delta_{\text{L}}(t_{\text{pdFF}} + t_{\text{pdC}} + t_{\text{CL}}) \\ &\quad - \delta_{\text{C}}(t_{\text{CC}} + t_{\text{us}}), \qquad (7)\\ \Delta slack^{\text{degradation}}_{\text{hold}} &= \delta_{\text{C}}(t_{\text{CC}} + t_{\text{us}}) \\ &\quad - \delta_{\text{L}}(t_{\text{cdFF}} + t_{\text{cdC}} + t_{\text{CL}}). \qquad (8)\end{aligned}$$

Equations 7 and 8 express two opposing forces that govern slack degradation. In setup, if the degradation in the capture path exceeds that in the launch path, the setup slack is degraded. Conversely, if the degradation in the launch path exceeds that in the capture path, the hold slack is degraded. Even if $\delta_{\text{L}} = \delta_{\text{C}}$, indicating that both the capture and launch paths experience symmetric aging, both slacks can still be degraded. Moreover, setup timing violations can occur when $\Delta slack^{\text{degradation}}_{\text{setup}} > \Delta slack_{\text{setup}}$, whereas hold timing violations can happen when $\Delta slack^{\text{degradation}}_{\text{hold}} > \Delta slack_{\text{hold}}$.

## IV. Experimental analysis

This study shows how asymmetric transistor aging affects GPGPU processing elements. Functional simulations are analyzed to extract the aging profile of GPGPUs, which includes SP measurements. Next, we comprehensively analyze the timing with aging models incorporated into the aging profile. As part of our experimental analysis, we examine potential timing violations due to asymmetric aging while considering the impact of (i) useful skew optimization and (ii) the asymmetry between net and cell delays. Additionally, we show that certain timing paths may require extra timing margins to compensate for the degradation due to asymmetric aging. Finally, we summarize the requisite extensions for conventional physical design flow to account for the impact of asymmetric transistor aging.

Our functional experiments were conducted using the GPGPU simulator [11]. The simulation environment incorporates cycle-level modeling of the NVIDIA Volta V100 GPGPU [12], enabling the execution of CUDA or OpenCL computing workloads. To meet the specific requirements of our experiments, we modified the simulation platform and integrated essential mechanisms for the required measurements. For benchmarking, we used the Neural Network (NN) benchmark from the gpgpu-sim benchmark suite of IPSS [11]. The NN benchmark encompasses training and inference of neural networks.

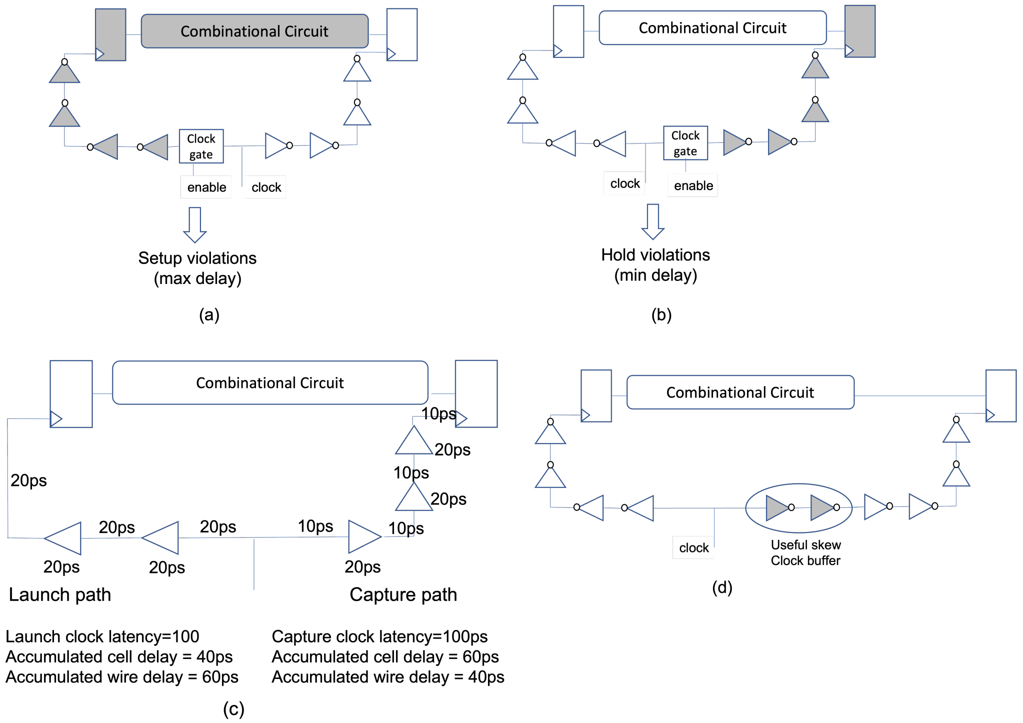

**Fig. 1: Asymmetric aging in clock trees: (a) clock gate on the launch clock, (b) clock gate on the capture clock, (c) asymmetry between accumulated logical cells delay and net delay, and (d) useful clock skew.**

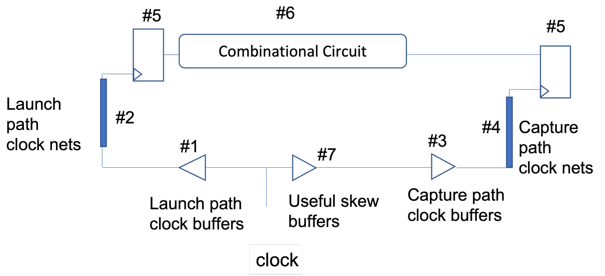

**Fig. 2: Asymmetric aging in clock trees in the presence of useful skew and net delays.**

As part of our functional experimental analysis we measured the activity ratio and SP of the integer execution unit and the single-precision floating-point unit (FPU). Figures 3 and 4 illustrate the activity measured in the Volta V100 Streaming Multiprocessors while benchmarking the NN. Activity is quantified as the percentage of time the execution unit remains active relative to the total elapsed time. Our experimental findings indicate that the integer execution units within all streaming multiprocessors are idle 80%–85% of the time, while the FPUs are idle over 98% of the time. These observations imply that the GPGPU processing elements are susceptible to transistor aging due to their extended idle periods. Being in an idle state for a long time amplifies their exposure to asymmetric aging, which can introduce serious reliability concerns.

To examine the GPGPU case study, we use the integer execution unit and FPU from an open-source Nyuzi Processor GPGPU[1] [16] to investigate how asymmetric aging affects timing. We fully synthesize the place and route on the GPGPU modules in the 28 nm process node. Cadence® $Genus^{TM}$ implements the synthesis, and the place and route is executed within Cadence® $Innovus^{TM}$. The clock frequency of the integer execution unit is 250 MHz, and the FPU is assumed to operate at 167 MHz. For the timing analysis, we use the measured aging profile of the Volta V100 in conjunction with aging-aware library models, as detailed in Ref. [9]. These models consider the impact of the BTI by adjusting cell delays based on NBTI degradation factors derived from SP values extracted from the functional simulations depicted in Figs. 3 and 4.

**TABLE II: Summary of timing analysis in presence of asymmetric aging.**

| Func, Units | Setup WNS [ps] Number of Violations | | | |
|---|---|---|---|---|
| | *No Aging* | *Aging* | *Aging+U.S.* | *Aging+U.S.+N.D.* |
| I-EXU | 0 / 0 | −157 / 8 | −128 / 4 | −181 / 22 |
| FPU | 0 / 0 | −238 / 7077 | −237 / 7042 | −337 / 9753 |

| Func. Units | Hold WNS [ps] Number of Violations | | | |
|---|---|---|---|---|
| | *No Aging* | *Aging* | *Aging+U.S.* | *Aging+U.S.+N.D.* |
| I-EXU | +13 / 0 | +9.5 / 0 | −1 / 2 | −1 / 10 |
| FPU | +4.5 / 0 | +4.5 / 0 | +3.1 / 0 | −2 / 7 |

The timing results in Table II present the worst negative slack (WNS) and the number of timing violations for both the

[1] https://github.com/jbush001/NyuziProcessor/tree/master/hardware/core

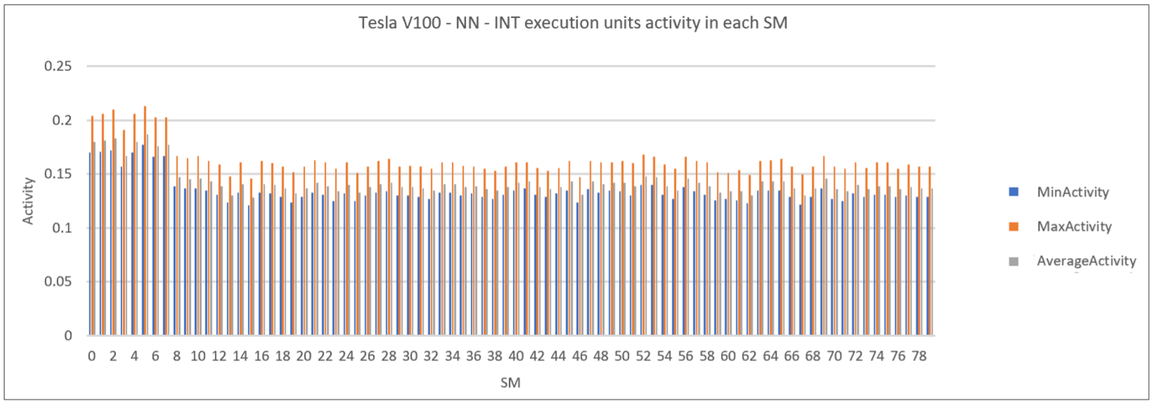

**Fig. 3: Activity fraction of integer execution units in Volta V100 Streaming Multiprocessors for NN benchmark.**

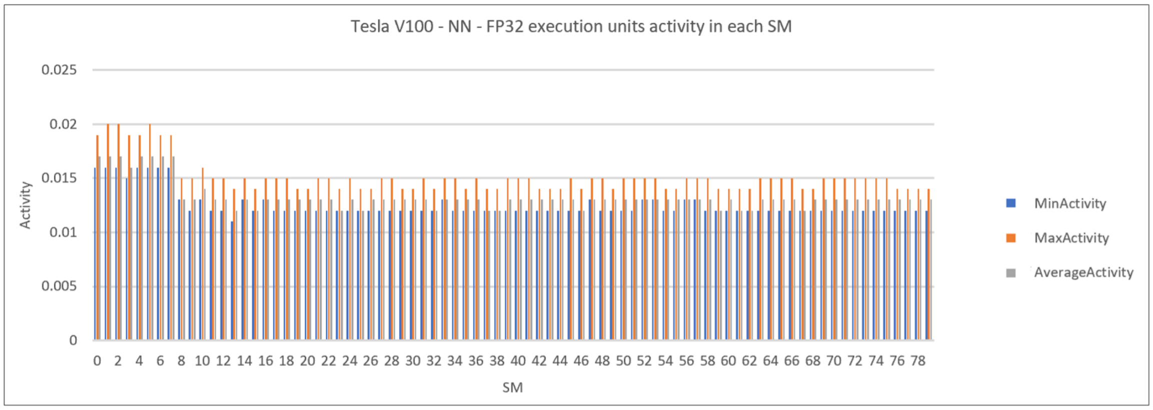

**Fig. 4: Activity fraction of single precision floating-point units in Volta V100 Streaming Multiprocessors for NN benchmark.**

setup and hold analyses in the following cases:

(1) when the effect of asymmetric aging is not considered (No Aging);
(2) when the effect of asymmetric aging is considered in timing analysis (Aging);
(3) when using useful skew optimization and considering asymmetric aging impact (Aging+U.S.);
(4) when using useful skew optimization and considering asymmetric aging in conjunction with the asymmetry between net delay and cell delay (Aging+U.S.+N.D.).

The results presented in Table II indicate that, upon considering asymmetric aging with no useful skew optimization, setup violations are introduced in the integer execution unit (I-EXU) and FPU. In addition, the hold slack of the I-EXU degrades. When useful optimization is applied while considering the impact of asymmetric aging, the setup WNS improves and the number of violations decreases since it delays the capture edge. However, useful skew further degrades the hold slack and introduces hold violations in the I-EXU. Using useful skew optimization and considering asymmetric aging in conjunction with the asymmetry between net delay and cell delay introduces a significant number of violations for both setup and hold. In the case of setup analysis, the I-EXU WNS is reduced from $-128$ ps to $-181$ ps, while the FPU WNS is reduced from $-237$ ps to $-337$ ps. Additionally, the number of setup violations increases from 4 to 22 in the I-EXU and from 7042 to 9753 in the FPU. In the hold analysis, the WNS remains the same for the I-EXU, whereas the number of violations increases from 2 to 10. In the FPU, the hold WNS drops from +3.1 to $-2.0$ with a total of 7 violations.

The experimental results summarized in Table II indicate that asymmetric aging can introduce severe timing violations related to clock trees. Furthermore, a combination of useful clock skew optimization and asymmetry between the net delay and cell delay can even further intensify timing violations and degrade circuit reliability. Hold violations are considered to be more severe than setup violations because the latter can be mitigated by reducing the clock frequency, whereas hold violations cannot be mitigated. Therefore, given asymmetric transistor aging, the timing constraints should be extended as indicated by Eqs. (7) and (8), and extra timing margins should be added to logical setup and hold paths that exhibit marginal and negative slacks. Finally, Fig. 5 summarizes the full flow used in our analysis. Our physical design flow extends the conventional flow, which includes (1) synthesis, (2) place and route, (3) extraction, (7) timing report generation, and (8) timing fixes, with new capabilities to incorporate the dependence of timing on asymmetric aging. The new components added to the flow include

(5) functional simulation of the design, which is used to

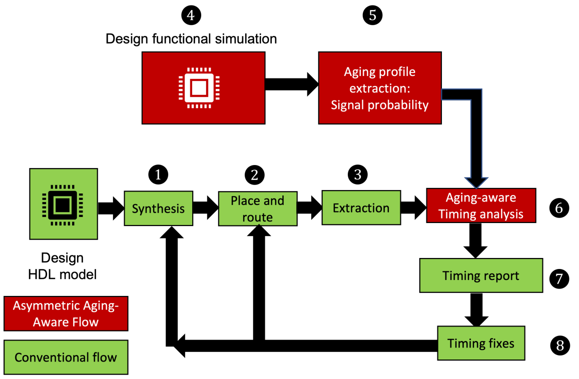
4
5
Design functional simulation
Aging profile extraction: Signal probability
1
2
3
Synthesis
Place and route
Extraction
Aging-aware Timing analysis
6
Design HDL model
Timing report
7
Timing fixes
8
Asymmetric Aging-Aware Flow
Conventional flow

**Fig. 5: Extended physical design flow in presence of asymmetric transistor aging.**

measure the idleness for a given workload;

(6) extraction of the aging profile in the form of SP;

(7) aging-aware timing analysis, which incorporate aging-aware libraries that are derated based on their corresponding SP.

In summary, by incorporating these new capabilities into our physical design flow, we address the challenges posed by asymmetric aging, thereby ensuring improved timing analysis and design reliability, as demonstrated on the GPGPU processing elements.

## V. Conclusions

This study investigates how asymmetric aging affects clock tree design considerations. Clock trees are vital circuit resources responsible for distributing a balanced clock signal across the chip die, and their reliability is critical to overall IC operation. This study reveals that clock trees are highly susceptible to reliability issues arising from asymmetric aging. While previous studies primarily focused on how asymmetric aging affects gated clocks, the present work expands the scope of such investigations, highlighting the significance of factors such as useful skews and the asymmetry between net and cell delays in causing substantial timing violations in clock trees, potentially leading to IC failure. Using General Purpose Graphic Units (GPGPUs) as a case study, our experimental analysis reveals the timing violations resulting from asymmetric transistor aging. In response to the challenges posed by asymmetric aging, we introduce an aging-aware design flow, which includes new extensions to the timing constraints. These extensions add extra timing margins to logical setup and hold paths that exhibit marginal and negative slacks. Understanding how asymmetric aging affects clock tree design is crucial for ensuring the reliability and performance of ICs. By incorporating aging-aware design flow and timing constraints, potential timing violations can be mitigated, thereby improving overall IC operation and longevity.

## References

[1] S. Ogawa and N. Shiono, Generalized diffusion-reaction model for the low-field charge build up instability at the Si-SiO2 interface, Physical Review, 51(7):4218–4230, Feb. 1995.

[2] M. A. Alam, H. Kufluoglu, D. Varghese, and S. Mahapatra, “A comprehensive model for PMOS NBTI degradation,” Microelectron. Rel., vol. 47, no. 6, pp. 853–862, Jun. 2007. https://doi.org/10.1016/j.microrel.2006.10.012

[3] S. Bharadwaj, W. Wang, R. Vattikonda, Y. Cao, and S. Vrudhula, “Predictive modeling of the NBTI effect for reliable design,” in Proc. Custom Integrated Circuits Conf., Sep. 2006, pp. 189–192.

[4] W. Wang, V. Reddy, A. T. Krishnan, R. Vattikonda, S. Krishnan, and Y. Cao, “Compact modeling and simulation of circuit reliability for 65 nm CMOS technology,” IEEE Trans. Device Mater. Rel., vol. 7, no. 4, pp. 509–517, Dec. 2007.

[5] A. Ricketts, J. Singh., K. Ramakrishnan, N. Vijaykrishnan, and D. K. Pradhan, “Investigating the impact of NBTI on different power saving cache strategies,” in Proc. DATE, Mar. 2010, pp. 592–597.

[6] M. Agarwal, B. C. Paul, Ming Zhang, and S. Mitra, “Circuit failure prediction and its application to transistor aging”, VLSI Test Symposium, pages 277–286, May 2007.

[7] W. Wang, Z. Wei, S. Yang, and Y. Cao, “An efficient method to identify critical gates under circuit aging,” in Proc. Int. Conf. Comput. Aided Des., Nov. 2007, pp. 735–740.

[8] Haider Muhi Abbas, Mark Zwolinski, and Basel Halak. Aging Mitigation Techniques for Microprocessors Using Anti-aging Software. Chapter 3, Ageing of Integrated Circuits - Causes, Effects and Mitigation Techniques, Springer, Cham. ISBN 978-3-030-23781-3.

[9] F. Gabbay and A. Mendelson, “Asymmetric aging effect on modern microprocessors,” Microelectronics Reliability, vol. 119, pp. 71–81, 2021, https://doi.org/10.1016/j.microrel.2021.114090.

[10] S. Arasu, M. Nourani, F. Cano, J. M. Carulli and V. Reddy, ”Asymmetric aging of clock networks in power efficient designs,” Fifteenth International Symposium on Quality Electronic Design, Santa Clara, CA, USA, 2014, pp. 484-489, doi: 10.1109/ISQED.2014.6783365.

[11] M. Khairy, Z. Shen, T. M. Aamodt, T. G. Rogers, ”Accel-Sim: An Extensible Simulation Framework for Validated GPU Modeling”, In proceedings of the 47th IEEE/ACM International Symposium on Computer Architecture (ISCA), May 29 - June 3, 2020.

[12] A.A. Awan, H. Subramoni, and D.K. Panda. An In-depth Performance Characterization of CPU- and GPU-based DNN Training on Modern Architectures. Proceedings of the Machine Learning on HPC Environments, 2017.

[13] Component Technical Committee Automotive Electronics Council. Failure mechanism based stress test qualification for integrated circuit. AEC – Q100 – REV-G standard.

[14] J. Huang, J. Chai, and S. Cho, “Deep learning in finance and banking: A literature review and classification,” Frontiers of Business Research in China, vol. 14, 2020, https://doi.org/10.1186/s11782-020-00082-6.

[15] M. Macas, C. Wu, and W. Fuertes, “A survey on deep learning for cybersecurity: Progress, challenges, and opportunities,” Computer Networks, vol. 212, 2016, https://doi.org/10.1016/j.comnet.2022.109032.

[16] Bush, J., TaheriNejad, N., Willegger, E., Wojcik, M., Kessler, M., Blatnik, J., Daktylidis, I., Ferdigg, J. and Haslauer, D., 2021. Nyuzi: An Open Source GPGPU for Graphics, Enhanced with OpenCL Compiler for Calculations. In IEEE Design, Automation & Test in Europe (p. 1). IEEE.