Gain and Pain of a Reliable Delay Model

Jürgen Maier
ECS Group, TU Wien, Vienna
jmaier@ecs.tuwien.ac.at

***Abstract*—In this paper we evaluate a promising delay estimation method, the Involution Delay Model. We apply it to three simple circuits (a combinatorial loop, an SR latch and an adder), interpret the delivered results and determine realistic overhead estimations. Comparisons to analog SPICE simulations reveal fine-grained behavioral coverage, whereat the commonly used digital inertial delay model shows severe shortcomings. Overall, the Involution Delay Model is able to identify a wide range of malicious behavior and is thus a viable upgrade to available delay models in modern digital timing simulation tools.**

***Index Terms*—glitch propagation, pulse degradation, faithful digital timing simulation, metastability analysis**

I. Introduction

In modern circuit designs a lot of effort is invested into predicting the circuit behavior at early development stages. The most accurate methods currently available for this task are *analog* simulation suites, like SPICE. These calculate time- and value-continuous signal traces based on very elaborate physical models, which is a computationally expensive task.

For larger circuits, simulations are thus executed in the *digital* domain (zero time transitions between `LO` and `HI`). *Static timing analysis* (STA) considers the gate's static delays to determine important characteristics, like the maximum clock frequency. However, for more evolved effects, e.g., signal degradation or interference leading to very short pulses, *timing simulations* are indispensable. These apply input trajectories and predict their propagation through the circuit. In detail, time and direction of a gate's output transitions are estimated based on time and direction of its input ones. From the currently available timing simulation methods (see Section II), the Involution Delay Model (IDM) is the sole faithful candidate [1], meaning that all the physical behavior can be depicted in the model and vice versa. Recently Öhlinger *et al.* [2] practically applied the IDM to basic circuits, however, primarily to evaluate the accuracy of the introduced simulation framework. Consequently little is known about the behavioral coverage and performance of the IDM in realistic setups.

Main contributions: In this paper we are thus extending the evaluation of the IDM to an `OR Loop`, an `SR Latch` and a ripple-carry `Adder`. We (i) run analog and digital simulations, (ii) evaluate the achieved results and finally (iii) determine the introduced overhead. Our analyses (1) confirm the simple applicability stated in [2], (2) show a high correlation between IDM and analog simulations leading to highly reliable results, and (3) reveal how commonly used approaches fail to describe a wide range of possible behaviors. Although its sometimes significant overhead in simulation time compared to inertial delay (up to 250 %), we consider the IDM a viable upgrade that allows to reliably identify potentially harmful locations resp. input trajectories in critical circuits.

In Section II we provide an introduction to existing delay models, while Section III describes the simulation setup and the investigated circuits. A discussion of the achieved results together with an evaluation of the introduced overhead follows in Section IV. Finally, we conclude the paper in Section V.

II. Background

The most simplistic approach for digital delay estimation is the *pure* delay, which introduces a constant retardation of rising ($\delta_\infty^\uparrow$) resp. falling ($\delta_\infty^\downarrow$) input transitions. To account for the suppression of small pulses, the *inertial* delay additionally blocks pulses with an input width Δ^i below a certain threshold. Although analog SPICE simulations show a comparable behavior, these simplistic models fail to depict the gradual change of the output pulse width Δ^o [3], and are thus unable to properly cover very short pulses (glitches). Juan-Chico *et al.* therefore developed the Degradation Delay Model (DDM) [4] that uses nonconstant delay functions $\delta_\uparrow(T)$ resp. $\delta_\downarrow(T)$. The parameter T is, thereby, defined as the time span from the last output transition to the current input transition.

For $T \gg 0$, a constant value $\delta(T) \approx \delta_\infty$, comparable to inertial/pure delay, can be assumed. For decreasing T the delay starts to decline, leading to significant degradation and eventually to the equilibrium point $-T = \delta(T)$. For even shorter input pulses the model schedules the digital output transitions in the wrong temporal order, e.g., when starting at `LO` a falling before a rising transition. In this case we speak of *cancellation* and both transition are removed. In the analog domain this corresponds to sub-threshold trajectories.

Although canceled transitions are not visible, their respective transition times are of utmost importance, as the latest one serves as reference for calculating T. To predict the delay for $T < -\delta(T)$, the DDM simply extends the fitting of $\delta(T)$ derived for $T > -\delta(T)$. While this seems, at a first glance, like a legitimate choice, it causes the delay estimation to fail in certain circumstances. Actually, Függer *et al.* [5] were able to prove this deficiency in all existing delay estimation approaches. To prevent this, the Involution Delay Model [1] ensures diminishing input-to-output impact for $\Delta_i \to 0$ by mirroring the delay functions along $T = -\delta(T)$ (see Fig. 1).

This research was funded by the Austrian Science Fund (FWF) project DMAC (P32431).

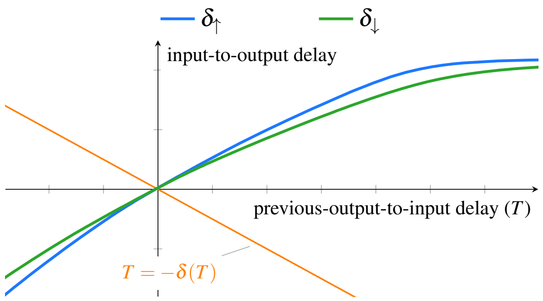


Fig. 1. Delay functions of the Involution Delay Model.

In the analog domain the IDM can be interpreted in the following fashion: The digital input signal first passes a pure delay component and is then transformed to a continuous trajectory using two unique waveforms ($f_\uparrow$ and $f_\downarrow$), which are switched instantaneously upon a transition. Finally the analog trajectory is fed into a comparator, which issues a digital output event whenever the voltage V_{th}^{out} is crossed.

Although the IDM has much higher expressive power, modern circuit designs still heavily rely on the simple pure and inertial delay models. This is not surprising, given the very good integration in state-of-the-art simulation suites and thus its simple applicability. To ease the application of the IDM, Öhlinger *et al.* [2] developed the `InvTool`, whose VHDL procedures simply have to be linked to an existing design.

III. Experimental Setup

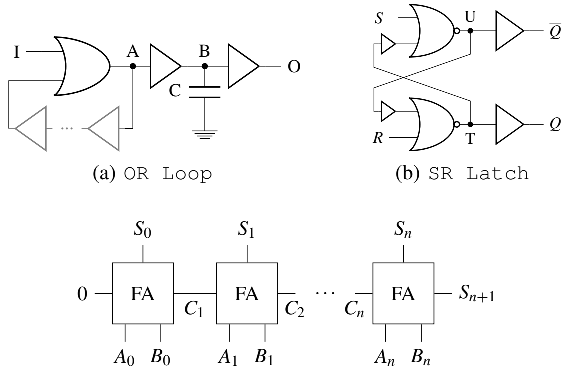
In this section we describe the framework we used to determine the performance and behavioral coverage of the IDM. The simulation data is freely available on-line[1].

A. Design Flow

For the sake of realistic results we utilize the Cadence tools Genus and Innovus (version 19.11) to place & route the design using the 15 nm Nangate Open Cell Library with FreePDK15$^{\text{TM}}$ FinFET models [6] ($V_{DD} = 0.8\,\text{V}$). Then we automatically extract the parasitics and static delay values. For analog transient simulations we back-annotate the extracted parasitics to a transistor level model, which is then executed using Cadence Spectre (version 19.1).

The digital simulations are run with Mentor ModelSim (version 10.5c). Two prediction approaches were executed: The default one provided by ModelSim (INE), essentially an inertial delay, and the Involution Delay Model (IDM). For the latter we use the exp-channel model from the `InvTool`[2]. Since we are only able to extract the static delay values from the layout, we set, for the sake of simplicity, the pure delay to a constant value of 1 ps.

We want to emphasize at this point that we were able to confirm the simplicity of applying IDM to an existing design flow. Starting from the test setup for INE we solely had to compile and link the respective IDM files. Nevertheless, we were not able to reuse our testbench, since the IDM and the tool's delay model are implemented in differing hardware description languages (Verilog vs. VHDL). Some commands, such as forcing signals, do not properly work across language boundaries. This made it necessary to duplicate the testbench, while, of course, conserving the behavior.



(a) `OR Loop` (b) `SR Latch`

(c) `Adder`

Fig. 2. gate level implementations

B. Circuits

In the sequel we introduce the circuits used in our simulations. Note that additional buffers, which we added at the in- and output to emulate the settings far away from the chip boundaries, are not explicitly depicted.

1) `OR Loop`: The circuit shown in Fig. 2a has been used for formal proofs in [1]. It utilizes an arbitrary amount of buffers to create a combinatorial loop, whereat up-pulses are coupled in via a single `OR`-gate. Based on the input pulse width Δ^I the signal may oscillate for a possibly infinite amount of time before vanishing or setting the loop to `HI`. Depending on the length of the feedback path either distinguished pulses or intermediate voltage values are observable. While the former corresponds to a simple ring oscillator the latter depicts *metastability* [7]. To ease their descriptions we are going to use Δ_n^{HI} and Δ_n^{LO} to denote the high respectively low time of the n^{th} oscillation at node A.

2) `SR Latch`: The second circuit is the Set Reset Latch, a well-known circuit with the possibility for metastability, as shown in Fig. 2b. Note that we added a single buffer on the coupling paths between the `NOR`-gates to pronounce the observable effects and thus ease their detection.

The `SR Latch` operates very intuitively: If the set (S) input turns `HI`, Q switches to `HI`, for a `HI` on the reset (R) input, Q changes to `LO`. $\overline{Q}$ represents the inverse of Q. Note that iff one input is `LO`, the `SR Latch` behaves, w.r.t. the other one, just like the `OR Loop`: Very short pulses are blocked, very long ones immediately set the loop, while ones in between may lead to metastability. Significantly different behavior is possible, however, if both inputs are allowed to change. While one steers the loop into a metastable state, the other one can either support or impair its resolution. We will exploit this fact in our simulations.

[1] https://github.com/jmaier0/idm_evaluation

[2] https://github.com/oehlinscher/InvolutionTool

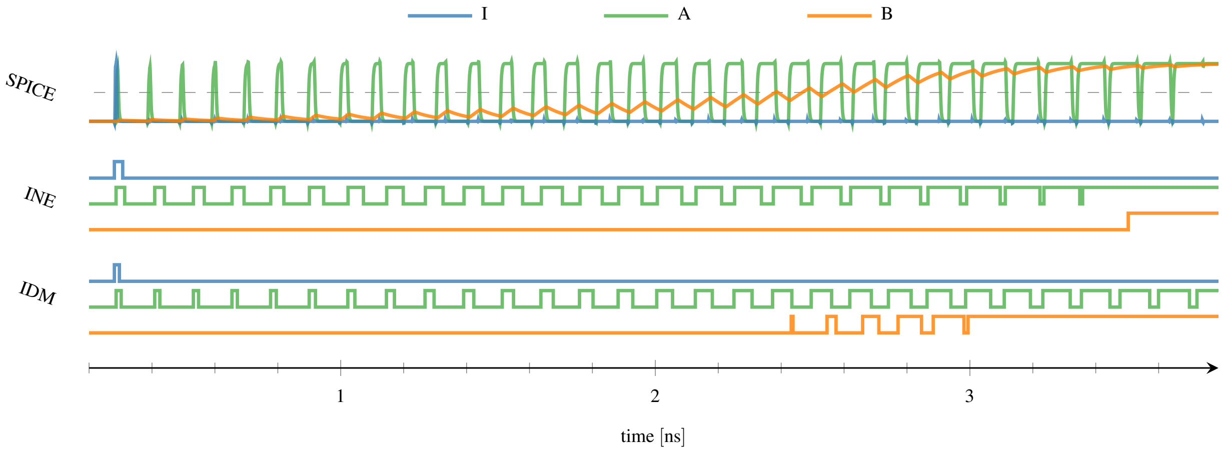


Fig. 3. Analog and digital simulation results for the `OR Loop` with long feedback.

3) `Adder`: To investigate the scaling of the IDM and its predictions on loop-free circuits we, also simulated the simple ripple carry adder shown in Fig. 2c. Out of the manifold input possibilities, those leading to a maximum number of transitions allow an investigation of the whole circuit in a single simulation run. For this purpose we chose $B_0B_1B_2B_3 = 1111$, $A_0A_1A_2A_3 = 0000$ and introduced an up-pulse on signal A_0. For a down-pulse on signal A_0 we used a very similar setup, with the sole difference of setting $A_0A_1A_2A_3 = 1000$ initially.

IV. Results

In this section we present and compare the analog resp. digital simulation results for the circuits introduced in Section III.

A. `OR Loop` with Long Feedback

For our first experiments we added thirty buffers to the feedback path. For this setup we extracted $\delta_\infty^\uparrow \neq \delta_\infty^\downarrow$ after place & route. This implies, that only a single Δ^I leads to an infinite oscillation.

1) SPICE: Fig. 3 (top) shows an initially very short pulse that grows and eventually settles the loop at V_{DD}. Due to the capacitive load at node B, it seems as if charging and discharging curves are switched instantaneously[3]. Consequently, the threshold (dashed line) is crossed multiple times. Noteworthy is the high sensitivity of the feedback loop in this state and thus also the very low probability to reach it. We had to vary Δ^I in steps of $1\,\mathrm{as}$ in order to eventually generate an oscillation trace inside the loop that lasted at most $4\,\mathrm{ns}$.

2) INE: At a first glance the inertial delay results shown in Fig. 3 (middle) look comparable. However, on closer examination the observed pulse turns out to be the shortest one that can be inserted into the loop (smaller ones are removed by a high-delay buffer upstream). This indicates a general problem: A gate with long delay may remove a big share of all input pulses, potentially including highly relevant ones. Consequently, it is impossible to detect any infinite or decaying oscillations for the shown circuit using INE.

Note that the rising transition at node B only occurs after the loop has fully settled, i.e., the oscillations have ceased. This can again be explained by the big delay of the succeeding gate, which thus serves as a metastability filter. This does, however, not correspond well to the analog simulations, where the threshold is already crossed way before the loop is fully locked. Therefore, INE is not suited to properly describe the exact behavior of the circuit in such circumstances. In particular, it is impossible to achieve pulses at node B for the inertial delay model: only a single transition is observed or none at all.

3) IDM: Compared to INE, the Involution Delay Model achieves a much more fine-grained behavioral description. First and foremost, any value of Δ_0^{HI} can be generated, also ones that quickly decay. Fig. 3 (bottom) shows a simulation with increasing Δ_n^{HI} for ascending n, which matches analog simulations qualitatively very well.

By properly tuning Δ^I it is even possible to achieve an infinite pulse train, i.e., one that perfectly recreates itself. Note that, although the loop is highly unstable in this configuration, not a single transition on B could be observed, which reveals a problem of the IDM: Depending on the value of the discretization threshold voltage V_{th}^{out}, zero, one or infinitely many transitions are indicated for the same analog trajectory.

B. `OR Loop` with Direct Feedback

For the simulations presented in the sequel we remove all gates in the feedback path and thus force a direct transition to the constant metastable voltage.

1) SPICE: Analog simulations in Fig. 4 show two traces on node A, which stay at a constant value near V_{th} for some time and then resolve to either `LO` or `HI`. The fact that the corresponding Δ^I only differ by $1\,\mathrm{as}$ and, nonetheless, it is only possible to stay in the metastable state for a few picoseconds, indicate the very high sensitivity of this circuit configuration.

[3] Recall that this perfectly matches the analog domain model of the IDM.

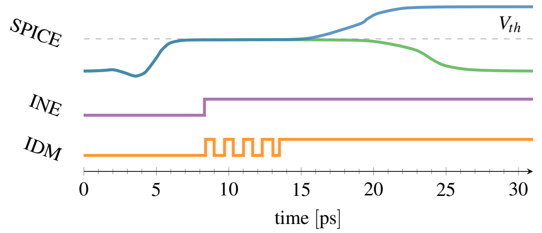


Fig. 4. Analog and digital simulation results of node A for the `OR Loop` with direct feedback path.

2) INE: As described in Section IV-A a gate upstream filters many incoming pulses. In fact, only those longer than the delay of the storage loop are able to pass, causing an immediate switch to `HI`. Consequently, for INE, the simulation either delivers a single rising transition on all wires or none at all. While this might seem reasonable at a first glance, metastability, and thus the increase in delay, is not revealed, suggesting falsely a settled and well defined behavior.

3) IDM: Although the analog simulations do not show any V_{th} crossing during metastability, the IDM again delivers an oscillatory behavior, which seems to be awfully wrong. However, considering the analog representation, more specifically the switching between $f_\uparrow$ and $f_\downarrow$, it becomes apparent, that the closest the analog trajectory in the IDM can get to a constant intermediate value is to oscillate around it.

The fact that the IDM uses a pulse train to describe both real oscillations (cf. Section IV-A) as well as metastability begs the question how these can be distinguished? Solely based on the digital predictions, this is impossible. Only in combination with the switching waveforms $f_\uparrow$ & $f_\downarrow$, or the static delays $\delta_\infty^{\uparrow/\downarrow}$, one is able to estimate the voltage gain during the `HI` resp. `LO` period. Although this seems very disadvantageous for the IDM, be advised that also for INE the delay values are required to determine if a pulse is close to suppression.

We can conclude that an oscillating simulation trace in IDM does not necessarily indicate an undesired behavior. Just as periods drop below a circuit dependent value (≈ the static delay) ill shaped pulses resp. metastability have to be inferred.

C. `SR Latch`

For the `SR Latch` INE again fails to cover very important parts of the real behavior. Consequently, we present these results only in the extended version of this article [8].

1) Set or Reset Input Pulse: For setting either S or R `LO`, the IDM describes the behavior in, and also the resolution out of, metastability faithfully, as shown in Fig. 5. This enables us to search for "malicious" input conditions that prolong the metastable state. In the figure, a very long `HI` phase (Δ_5^{HI}) on node T is visible as Q switches to constant `HI`. To prevent the oscillation from resolving, it would be necessary to decrease Δ_5^{HI} and simultaneously increase Δ_5^{LO}. It can be easily retraced that a properly placed up-pulse on the reset input R does the trick. Note that a similar approach was used by Reiher *et al.* [9] to prolong the metastability of synchronizers.

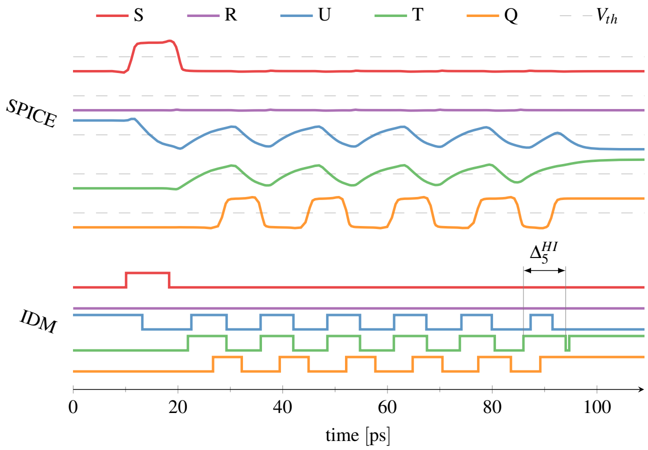


Fig. 5. Simulation results showing metastability in the `SR Latch`.

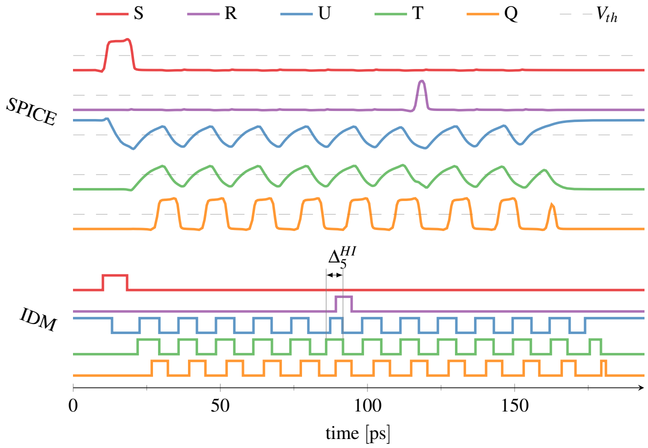


Fig. 6. Simulation results of steering the `SR Latch` back into metastability.

2) Set and Reset Input Pulse: SPICE simulations shown in Fig. 6 (middle) confirm our predictions. Not only is metastability extended but also a resolution to `HI` is forced. We want to emphasize that the pulse on R used to prolong metastability is too short to have any impact on a fully settled memory loop. Only in combination with this particular unstable circuit state a change in value becomes possible. Consequently, a close observation of unstable states and short pulses is very important. Finally, an execution of the IDM shown in Fig. 6 (bottom) delivers exactly the predicted behavior. Cutting Δ_5^{HI} indeed sets the loop back into metastability, resulting in a very realistic representation of the underlying analog behavior.

D. `Adder`

Due to page constraints we only shortly discuss the results for the `Adder`. An extensive analysis can be found in [8]. Overall, analog simulations reveal that the circuit may transform a single, short input pulse to a glitch of varying length on each of its output. INE shows an inconsistent behavior in this regard, while the smooth pulse width changes are naturally much better modeled by the IDM.

TABLE I
SIMULATION TIME MEAN AND VARIANCE σ OF THE ADDER.

	INE		IDM		
#	$\overline{x}$ [s]	σ [s]	$\overline{y}$ [s]	σ [s]	overhead [%]
1	4.80	0.92	8.65	0.90	80.23
2	5.95	2.03	12.00	0.41	101.58
4	6.78	0.90	18.80	0.86	177.16
10	11.74	0.24	37.75	1.15	221.43
20	20.02	0.42	69.24	2.09	245.93
40	37.30	1.15	132.53	1.31	255.27

TABLE II
SIMULATION TIME MEAN AND VARIANCE OF THE CLOCK TREE.

	INE		IDM		
#	$\overline{x}$ [s]	σ [s]	$\overline{y}$ [s]	σ [s]	overhead [%]
1	26.07	2.18	41.46	1.12	59.06
2	41.17	0.46	69.58	1.56	69.01
4	71.32	1.27	122.09	1.25	71.17
10	188.27	49.26	368.30	127.09	95.62
20	1016.23	265.44	1294.92	451.77	27.42
40	2430.30	406.60	3554.59	576.95	46.26

E. Overhead

Calculating delay values for the IDM, which includes exponential and logarithmic operations, is obviously computationally more expensive than applying constant values paired with some minor removal checks for INE. To evaluate the overhead, we run extensive simulations and measure the execution time (Intel Xeon X5650, 1600 MHz, 32 GB RAM, CentOS 6.10). As test circuits we use the `Adder` and the `Clock Tree` of an open source MIPS processor [10]. The latter comprises of 227 inverters, which drive 123 flip-flops. To investigate the scaling we simply instantiate each unit multiple times. Due to a rather high variance σ, we run each simulation 30 times and calculate the average $\overline{x}$ respectively $\overline{y}$. Be advised that the presented values serve as lower bound, since real input signals may lead to very short internal pulses that increase the workload of IDM compared to INE.

The results in Table I and Table II clearly show the price for the improved coverage provided by the IDM. For the `Adder` the overhead increases with circuit size, while for 40 instances it is almost 260 %. For the `Clock Tree`, the overhead is lower and more constant, ranging from 27 to almost 100 %.

F. Summary

Our simulation results have shown, that INE fails to model wide ranges of the analog behaviors, especially high frequency oscillations and metastable intermediate voltages. The causes are single gates with larger delays, which have to be expected in almost every real world circuit. Relying exclusively on these predictions thus leads to a false sense of security. In these cases the IDM can significantly enhance the results, as it is able to stick much closer to the analog circuit behavior. This enables a more reliable identification of a wider range of malicious behavior in the digital domain and thus a better guidance of succeeding analog simulations, which are still mandatory to either confirm or dismiss the problems discovered in the digital domain.

V. Conclusion and Future Work

In this paper we evaluated the Involution Delay Model (IDM), an elaborate alternative to classic digital timing analysis approaches. To motivate this statement we ran analog (SPICE) and digital (inertial delay, IDM) simulations on three different circuits (`OR`-loop, an SR-latch and an adder) and compared the derived results. Appropriate interpretation of the predictions by the IDM confirmed the high behavioral coverage, especially for short pulses. On the contrary, a single high delay gate, which blocks a large share of incoming pulses, caused massive mispredictions for inertial delay. Consequently, state-of-the-art simulation suites tend to miss potentially malicious circuit behaviors like infinite oscillations or metastability and thus fail to deliver faithful predictions. Although an evaluation of the overhead showed a significant increase in simulation time, we think that the IDM poses a viable alternative to identifying malicious behavior, especially if confined to the most critical parts.

In our simulations we identified some problems regarding the discretization in the IDM. Future work will, thus, be denoted to developing an extension that allows a consistent description of a unique analog trajectory in the digital domain. Further improvements are also required towards accuracy. Characterizing each single gate by relying heavily on analog simulations is computationally expensive, thus approaches that yield reasonable results based on available, or easily achievable, data are instrumental for making the IDM a truly competitive alternative to existing delay models.

References

[1] M. Függer, R. Najvirt, T. Nowak, and U. Schmid, "A faithful binary circuit model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, August 2019.

[2] D. Öhlinger, J. Maier, M. Függer, and U. Schmid, "The involution tool for accurate digital timing and power analysis," *Integration*, vol. 76, pp. 87 – 98, 2021. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167926020302777

[3] J. Juan-Chico, P. Ruiz de Clavijo, M. J. Bellido, A. J. Acosta, and M. Valenia, "Inertial and degradation delay model for cmos logic gates," in *2000 IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 1, 2000, pp. 459–462 vol.1.

[4] M. J. Bellido, J. Juan, and M. Valencia, *Logic-Timing Simulation and the Degradation Delay Model.* Imperial College, 2005. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/p411

[5] M. Függer, T. Nowak, and U. Schmid, "Unfaithful glitch propagation in existing binary circuit models," in *Proceedings of the 19th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC).* New York City: IEEE Press, 2013, pp. 191–199.

[6] M. Martins, J. M. Matos, R. P. Ribas, A. Reis, G. Schlinker, L. Rech, and J. Michelsen, "Open cell library in 15nm freepdk technology," in *Proceedings of the 2015 Symposium on International Symposium on Physical Design*, ser. ISPD '15. New York, NY, USA: ACM, 2015, pp. 171–178. [Online]. Available: http://doi.acm.org/10.1145/2717764.2717783

[7] R. Ginosar, "Metastability and synchronizers: A tutorial," *IEEE Design & Test of Computers*, vol. 28, no. 5, pp. 23–35, 2011.

[8] J. Maier, "Gain and Pain of a Reliable Delay Model," June 2021. [Online]. Available: https://www.doi.org/10.36227/techrxiv.14872116

[9] J. Reiher, M. R. Greenstreet, and I. W. Jones, "Explaining Metastability in Real Synchronizers," in *2018 24th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, May 2018, pp. 59–67.

[10] J. C. Ferreira, "Physical synthesis with encounter (cadence)," https://paginas.fe.up.pt/~jcf/ensino/disciplinas/mieec/pcvlsi/2015-16/tut_encounter/tut_encounter.html, 2015/16.