A Composable Glitch-Aware Delay Model

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

Daniel Öhlinger
TU Wien
ECS Group
Vienna, Austria
daniel.oehlinger@tuwien.ac.at

Ulrich Schmid
TU Wien
ECS Group
Vienna, Austria
s@ecs.tuwien.ac.at

Matthias Függer
CNRS & Université Paris-Saclay
LMF, ENS Paris-Saclay & Inria
Gif-sur-Yvette, France
mfuegger@lsv.fr

Thomas Nowak
Université Paris-Saclay, CNRS
Orsay, France
thomas.nowak@lri.fr

ABSTRACT

We introduce the Composable Involution Delay Model (CIDM) for fast and accurate digital simulation. It is based on the Involution Delay Model (IDM) [Függer et al., IEEE TCAD 2020], which has been shown to be the only existing candidate model for faithful glitch propagation. The IDM, however, has shortcomings that limit its applicability. Our CIDM thus reduces the characterization effort by allowing independent discretization thresholds, improves composability and increases the modeling power by exposing canceled pulse trains at the gate interconnect. We formally show that, despite these improvements, the CIDM still retains the IDM's faithfulness.

CCS CONCEPTS

• **Hardware → Transition-based timing analysis**; **Compact delay models**; **Simulation and emulation**; *Electrical-level simulation.*

KEYWORDS

digital timing simulation; composable delay estimation model; faithful glitch propagation; pulse degradation

ACM Reference Format:
Jürgen Maier, Daniel Öhlinger, Ulrich Schmid, Matthias Függer, and Thomas Nowak. 2021. A Composable Glitch-Aware Delay Model. In *Proceedings of the Great Lakes Symposium on VLSI 2021 (GLSVLSI '21), June 22–25, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3453688.3461519

This research was partially funded by the Austrian Science Fund (FWF) projects DMAC (P32431) and ADynNet (P28182), the Centre National de la Recheche Scientifique projects ABIDE and BACON, the Agence Nationale de la Recherche project FREDDA (ANR-17-CE40-0013), and by the DigiCosme working group HicDiesMeus under ANR grant agreements ANR-11-LABEX-0045-DIGICOSME and ANR-11-IDEX-0003-02.

GLSVLSI '21, June 22–25, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8393-6/21/06.
https://doi.org/10.1145/3453688.3461519

1 INTRODUCTION AND CONTEXT

Accurate prediction of signal propagation is a crucial task in modern digital circuit design. Although the highest precision is obtained by analog simulations, e.g., using SPICE, they suffer from excessive simulation times. Digital timing analysis techniques, which rely on (i) discretizing the analog waveform at certain thresholds and (ii) simplified interconnect resp. gate delay models, are hence utilized to verify most parts of a circuit. Prominent examples of the latter are pure (constant input-to-output delay Δ) and inertial delays (constant delay Δ, pulses shorter than an upper bound are removed) [14]. To accurately determine Δ, which stays constant for all simulation runs, highly elaborate estimation methods like CCSM [13] and ECSM [3] are required.

Single-history delay models, like the *Degradation Delay Model (DDM)* [2], have been proposed as a more accurate alternative. Here, the input-to-output delay $\delta(T)$ depends on a single parameter, the previous-output-to-input delay T (see Fig. 1). Still, Függer *et al.* [4] showed that none of the existing delay models, including DDM, can reliably predict the propagation of glitches. Függer *et al.* [5] thus introduced the *Involution Delay Model (IDM)*, with the distinguishing property that the delay functions for rising ($\delta_\uparrow$) and falling ($\delta_\downarrow$) transitions form an involution, i.e., $-\delta_\uparrow(-\delta_\downarrow(T)) = T$, which enables faithful short pulse propagation. The Involution Tool [15], a simulation framework utilizing a digital simulation suite, has been used to confirm the models accuracy.

Nevertheless, the IDM shows, at the moment, several shortcomings which impair its composability:
(I) Ensuring the involution property requires specific ("matching") *discretization threshold voltages* V_{th}^{in*} and V_{th}^{out*} to discretize the analog waveforms at in- and output. These are not only unique for every single gate in the circuit but also difficult to determine.
(II) The discretization threshold voltages may vary among gates, i.e., V_{th}^{out*} of a given gate G_1 and V_{th}^{in*} of the successor gate G_2 in a path are not necessarily the same. Consequently, just adding the delay predictions for G_1 and G_2 cannot be expected to accurately model the delay of their composition. This is particularly true for circuits designs where different transistor threshold voltages [1] are used for tuning the delay-power trade-off [11] or reliability [7].

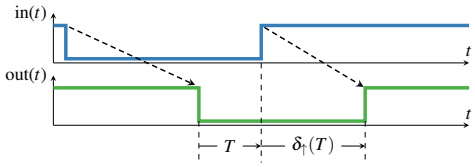

Figure 1: The delay $\delta_\uparrow$ as function of T. Taken from [15].

(III) Intermediate voltages, caused by creeping or oscillatory metastability, are expressed differently for various values of V_{th}^{out*}: Ultimately, a single analog trajectory may result in either zero, one or a whole train of digital transitions.

Main contributions: 1) We introduce the *Composable Involution Delay Model* (CIDM), whose discretization threshold voltages can be chosen arbitrarily. It enables the composition of successive gates, simplifies their characterization, and exposes canceled transitions at the gate interconnect. While CIDM is not strictly equivalent to IDM, we are able to show that every CIDM circuit has an equivalent IDM description. This allows a transfer of properties known to be true for IDM to CIDM; in particular, faithful propagation of glitches. 2) We developed the theoretical foundations and a simulation framework, which was incorporated into the Involution Tool [15].[1] A suite of experiments on an inverter chain with varying matching threshold voltages reveals an impressive increase in accuracy in many cases, and confirms the ability to model sub-threshold pulses.

Paper organization: We start with some basic properties of the IDM in Section 2. In Section 3, we empirically analyze the impact of changing V_{th}^{out*} and V_{th}^{in*} on the delay functions. Section 4 introduces and justifies the CIDM, while Section 5 proves its faithfulness. In Section 6, we describe the CIDM timing simulation algorithm and its implementation. Section 7 provides the results of our experiments. Some conclusion in Section 8 round-off our paper.

2 INVOLUTION DELAY MODEL BASICS

In this section, we briefly discuss the IDM. For further details, the interested reader is referred to the original publication [5].

The essential benefit of using involution delay functions is their ability to perfectly cancel zero-time input glitches: Suppose that the rising input transition in Fig. 1 is immediately, i.e., at the same time, succeeded by a falling one. As this essentially constitutes no pulse at all, the resulting output should also show no reaction. More specifically, the additional falling output transition has to be delayed *back in time* to exactly hit the time of the previous falling output transition: Note carefully that just placing the falling output transition at or before the rising one would not suffice, as the calculation of the parameter T for the next transition references its position in time. It is not difficult to verify that hitting the previous output transition time is indeed achieved by satisfying $-\delta_\uparrow(-\delta_\downarrow(T)) = T$ and $-\delta_\downarrow(-\delta_\uparrow(T)) = T$.

Lemma 3 in [5], restated as Lemma 1 below, shows that *strictly causal* involution channels, characterized by strictly increasing, concave delay functions with $\delta_\uparrow(0) > 0$ and $\delta_\downarrow(0) > 0$, give raise

[1]The original Involution Tool is accessible via https://github.com/oehlinscher/InvolutionTool, our extended version is provided at https://github.com/oehlinscher/CDMTool.

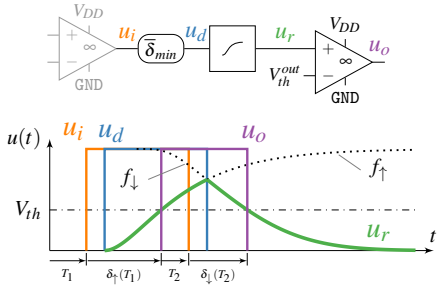

Figure 2: Analog channel model representation (upper part) with a sample execution (bottom part). Adapted from [5].

to a unique $\overline{\delta}_{min} > 0$ that (i) resides on the 2nd median $y = -x$ and (ii) is shared by $\delta_\uparrow$ and $\delta_\downarrow$ due to the involution property.

Lemma 1 ([5, Lem. 3]). *A strictly causal involution channel has a unique $\overline{\delta}_{min} > 0$ defined by $\delta_\uparrow(-\overline{\delta}_{min}) = \overline{\delta}_{min} = \delta_\downarrow(-\overline{\delta}_{min})$.*

In [5], Függer *et al.* have shown that self-inverse delay functions arise naturally in a (generalized) standard analog model that consists of a pure delay, a slew-rate limiter with generalized switching waveforms and an ideal comparator (see Fig. 2). First, the binary-valued input u_i is delayed by $\overline{\delta}_{min} > 0$, which assures causal channels, i.e., $\delta_{\uparrow/\downarrow}(0) > 0$. For every transition on u_d, the generalized slew-rate limiter switches to the corresponding waveform ($f_\downarrow/f_\uparrow$ for a falling/rising transition). Note that the value at u_r (the analog output voltage) does not jump, i.e., is a continuous function. Finally, the comparator generates the output u_o by discretizing the value of this waveform w.r.t. the discretization threshold V_{th}^{out}.

Using this representation, the need for $\delta_\uparrow(T), \delta_\downarrow(T) < 0$ can be explained by the necessity to cover sub-threshold pulses, i.e., ones that do not reach the output discretization threshold. In this case, the switching waveform has to be followed into the past to cross V_{th}^{out}, resulting in the seemingly acausal behavior.

3 DISCRETIZATION THRESHOLD VOLTAGES

In this section, we will empirically explore the relation of gate delays and discretization threshold voltages by means of simulation results.[2] In most of the following observations, we assume that a given physical (analog) gate is to be characterized as a zero-time Boolean gate with a succeeding IDM channel that models the delay. In order to accomplish concrete values, discretization threshold voltages V_{th}^{in} at the input and V_{th}^{out} at the output of a gate have to be fixed, and the pure delay component $\overline{\delta}_{min}$ of the IDM channel as well as the delay functions $\delta_\uparrow$ and $\delta_\downarrow$ are determined accordingly.

Definition 2. The input and output discretization voltages V_{th}^{in} and V_{th}^{out} are called *matching* for a gate, if the induced delay functions $\delta_\uparrow(T)$, $\delta_\downarrow(T)$ fulfill the condition $\delta_\uparrow(-\overline{\delta}_{min}) = \overline{\delta}_{min} = \delta_\downarrow(-\overline{\delta}_{min})$. To stress that a pair of input and output discretization threshold voltages is matching, they will be denoted as V_{th}^{in*} and V_{th}^{out*}.

[2]Our simulations have been performed for a buffer in the 15 nm NanGate library. However, since we are only reasoning about qualitative aspects common to all technologies, the actual choice has no significance.

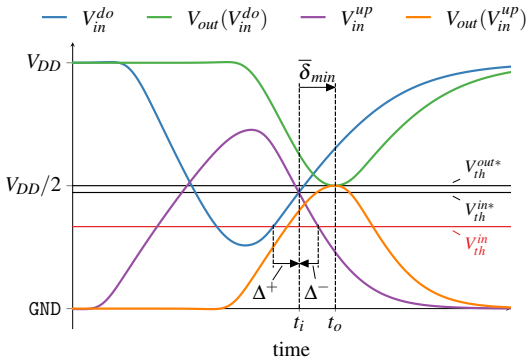

Figure 3: The relationship among V_{th}^{in*}, $\overline{\delta}_{min}$ and V_{th}^{out*}.

We will now characterize properties of matching discretization threshold voltages. They depend on many factors, including transistor threshold voltages [1] and the symmetry of the pMOS vs. nMOS stack. Since varying these parameters is commonly used in advanced circuit design to trade delay for power consumption [6, 11] and reliability [7], as well as for implementing special gates (e.g. logic-level conversion [12]), the range of suitable discretization threshold voltages could differ significantly among gates.

Considering these circumstances it seems impossible that output and input discretization values coincide among connected gates. However, the following observation shows that there is an unlimited number of matching discretization threshold pairs for IDM:

Observation 3. For every choice of V_{th}^{in}, there is exactly one matching V_{th}^{out}. Fixing either of them uniquely determines the other and, in addition, also the pure delay $\overline{\delta}_{min}$.

Justification. Let us fix V_{th}^{out} and investigate how V_{th}^{in} and $\overline{\delta}_{min}$ can be determined. For this purpose, we consider an analog pulse at V_{out} that barely touches V_{th}^{out}, i.e., results in a zero-time glitch in the digital domain. There is a *unique* positive and a *unique* negative analog output pulse with this shape, which is both confirmed by simulation results and analytic considerations on the underlying system of differential equations [8]. Now shift the positive and negative pulses in time such that their output voltages touch V_{th}^{out}, one from below and the other from above, at time t_o (see Fig. 3). Due to the condition $\delta_\downarrow(-\overline{\delta}_{min}) = \overline{\delta}_{min} = \delta_\uparrow(-\overline{\delta}_{min})$, this implies that the falling transition of the positive pulse and the rising transition of the negative pulse at the input must both cross V_{th}^{in} at time $t_i = t_o - \overline{\delta}_{min}$. Thus, fixing V_{th}^{out*} uniquely determines the matching V_{th}^{in*} and $\overline{\delta}_{min} = t_o - t_i$. □

Observation 3 has a severe consequence for the simulation of circuits in any model, like IDM, where Lemma 1 holds:

Observation 4. Fixing either V_{th}^{in} or V_{th}^{out} for a single gate G fixes the threshold voltages of all gates in the circuit when it is simulated in a model where Observation 3 holds.

It may take a large effort to properly characterize every gate such that the dependencies among discretization thresholds in a heavily interconnected circuit are fulfilled; in case of feedback loops, this may even be impossible. By contrast, an ideally composable delay model would use a uniform discretization threshold such as $V_{th}^{out} = V_{th}^{in} = V_{DD}/2$. To investigate whether IDM allows such a uniform choice, we proceed with Observation 5:

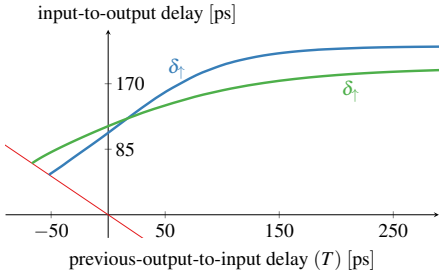

Figure 4: Characterizing a gate with $V_{th}^{in} = V_{th}^{out} = V_{DD}/2$.

Observation 5. Characterizing a gate with non-matching discretization thresholds V_{th}^{in} and V_{th}^{out*}, where matching V_{th}^{in*} and V_{th}^{out*} lead to an IDM channel with pure delay $\overline{\delta}_{min}$, results in delay functions $\delta_\uparrow(T)$, $\delta_\downarrow(T)$, which satisfy $\delta_\uparrow(-\delta_{min}^\uparrow) = \delta_{min}^\uparrow$ and $\delta_\downarrow(-\delta_{min}^\downarrow) = \delta_{min}^\downarrow$ for $\delta_{min}^\uparrow = \overline{\delta}_{min} + \Delta^+ \neq \delta_{min}^\downarrow = \overline{\delta}_{min} + \Delta^-$. Δ^+ and Δ^- have opposite sign, with $\Delta^+ > 0$ for $V_{th}^{in} < V_{th}^{in*}$.

Justification. The observation follows from refining the argument used for confirming Observation 3, where it was shown how matching V_{th}^{in*} and V_{th}^{out*} are achieved. For the non-matching case, we increase resp. decrease V_{th}^{in}, starting from V_{th}^{in*}, while keeping everything else, i.e., electronic characteristics, waveforms and V_{th}^{out}, unchanged. As illustrated in Fig. 3 for $V_{th}^{in} < V_{th}^{in*}$, it still takes $\overline{\delta}_{min}$ from hitting V_{th}^{in*} (at time $t_o - \overline{\delta}_{min}$) to seeing a zero time glitch (at time t_o) at the output. The falling transition has already crossed V_{th}^{in*} when it hits on V_{th}^{in}, whereas the rising transition still has some way to go: Denoting the switching waveforms of the preceding gate (driving the input) by $f_\uparrow$ and $f_\downarrow$, the pure delay for the rising resp. falling transition evaluates to $\delta_{min}^\uparrow = \overline{\delta}_{min} + \Delta^+$ and $\delta_{min}^\downarrow = \overline{\delta}_{min} + \Delta^-$ with

$$\Delta^+ = f_\uparrow^{-1}(V_{th}^{in*}) - f_\uparrow^{-1}(V_{th}^{in}), \quad \Delta^- = f_\downarrow^{-1}(V_{th}^{in*}) - f_\downarrow^{-1}(V_{th}^{in}). \quad (1)$$

Consequently, $\delta_\uparrow(-\delta_{min}^\uparrow) = \delta_{min}^\uparrow$ and $\delta_\downarrow(-\delta_{min}^\downarrow) = \delta_{min}^\downarrow$ indeed holds. Finally, since $f_\uparrow$ must obviously rise and $f_\downarrow$ must fall, it follows that if $\Delta^+ > 0$ (the case in Fig. 3) then $\Delta^- < 0$. □

Fig. 4 shows the derived delay function for non-matching discretization thresholds. Note the different pure delays $\delta_{min}^\uparrow \neq \delta_{min}^\downarrow$. Finally, the dependency of the IDM on the particular choice of the discretization threshold voltages also reveals a different problem:

Observation 6. Different choices of V_{th}^{out} can significantly change the digital model prediction of the IDM.

Justification. An oscillatory output behavior within range $[V_0, V_1]$ would only be reflected in the digital discretization if $V_{th}^{out} \in (V_0, V_1)$. Otherwise they are automatically removed by the comparator in Fig. 2, i.e., totally suppressed. □

4 COMPOSABLE INVOLUTION DELAYS

In this section, we define our *Composable Involution Delay Model* (CIDM), which allows to circumvent the problems presented in the previous section: According to Observation 5, using non-matching thresholds introduces a pure delay shift. The major building blocks of our CIDM are hence *PI channels*, which consist of a pure delay shifter with different shifts Δ^+ and Δ^- for rising and falling transitions followed by an IDM channel. In order to also alleviate the problem of invisible oscillations identified in Observation 6, we reshuffle the internal architecture of the original involution channels shown in Fig. 2 in order to expose trains of canceled transitions on the interconnecting wires.

Theorem 7 (PI channel properties). *Consider a channel PI formed by the concatenation of a pure delay shifter (Δ^+, Δ^-) with $\Delta^+, \Delta^- \in \mathbb{R}$ followed by an involution channel c, given via $\overline{\delta}_\uparrow(.)$ and $\overline{\delta}_\downarrow(.)$ with minimum delay $\overline{\delta}_{min}$. Then PI is* not *an involution channel, but rather characterized by delay functions defined as*

$$\delta_\uparrow(T) = \Delta^+ + \overline{\delta}_\uparrow(T + \Delta^+) \qquad \delta_\downarrow(T) = \Delta^- + \overline{\delta}_\downarrow(T + \Delta^-). \tag{2}$$

These functions satisfy

$$\delta_\uparrow\big(-\delta_\downarrow(T) - (\Delta^+ - \Delta^-)\big) = -T + (\Delta^+ - \Delta^-) \tag{3}$$

$$\delta_\downarrow\big(-\delta_\uparrow(T) + (\Delta^+ - \Delta^-)\big) = -T - (\Delta^+ - \Delta^-) \tag{4}$$

$$\delta_\uparrow(-\delta^\uparrow_{min}) = \delta^\uparrow_{min} \tag{5}$$

$$\delta_\downarrow(-\delta^\downarrow_{min}) = \delta^\downarrow_{min} \tag{6}$$

for $\delta^\uparrow_{min} = \overline{\delta}_{min} + \Delta^+$ and $\delta^\downarrow_{min} = \overline{\delta}_{min} + \Delta^-$.

Proof. Consider an input signal consisting of a simple negative pulse, as depicted in Fig. 1. Let t_i' resp. t_i be the time of the falling resp. rising input transition, t_p' resp. t_p the time of the falling resp. rising transition at the output of the pure delay shifter, and t_o' resp. t_o the time of the falling resp. rising transition after the involution channel. With $\overline{T} = t_p - t_o'$, we get $\overline{\delta}_\uparrow(\overline{T}) = t_o - t_p$ as well as $t_p = t_i + \Delta^+$ and $t_p' = t_i' + \Delta^-$.

For the delay function $\delta_\uparrow(T)$ of the PI channel, if we set $T = t_i - t_o' = t_i - t_p + t_p - t_o' = -\Delta^+ + \overline{T}$, we find

$$\begin{aligned}\delta_\uparrow(T) &= t_o - t_i = t_o - t_p + t_p - t_i = \Delta^+ + \overline{\delta}_\uparrow(\overline{T}) \\ &= \Delta^+ + \overline{\delta}_\uparrow(T + \Delta^+)\end{aligned} \tag{7}$$

as asserted. By setting $T = -\overline{\delta}_{min} - \Delta^+$ and using $\overline{\delta}_\uparrow(-\overline{\delta}_{min}) = \overline{\delta}_{min}$ the equality $\delta_\uparrow(-\overline{\delta}_{min} - \Delta^+) = \Delta^+ + \overline{\delta}_{min}$ is achieved, which confirms (5).

By analogous reasoning for an up-pulse at the input, which results in the same equations as above with Δ^+ exchanged with Δ^- and $\overline{\delta}_\uparrow(\overline{T})$ with $\overline{\delta}_\downarrow(\overline{T})$, we also get

$$\begin{aligned}\delta_\downarrow(T) &= t_o - t_i = t_o - t_p + t_p - t_i = \Delta^- + \overline{\delta}_\downarrow(\overline{T}) \\ &= \Delta^- + \overline{\delta}_\downarrow(T + \Delta^-)\end{aligned} \tag{8}$$

as asserted. Setting $T = -\overline{\delta}_{min} - \Delta^-$ and using $\overline{\delta}_\downarrow(-\overline{\delta}_{min}) = \overline{\delta}_{min}$ confirms (6) as well.

Using a simple parameter substitution allows to transform (7) and (8) to

$$\overline{\delta}_\uparrow(\overline{T}) = \delta_\uparrow(\overline{T} - \Delta^+) - \Delta^+ \tag{9}$$

$$\overline{\delta}_\downarrow(\overline{T}) = \delta_\downarrow(\overline{T} - \Delta^-) - \Delta^-. \tag{10}$$

Utilizing these in the involution property of $\overline{\delta}_\uparrow$ and $\overline{\delta}_\downarrow$ provides

$$\begin{aligned}\overline{T} &= -\overline{\delta}_\uparrow\big(-\overline{\delta}_\downarrow(\overline{T})\big) \\ &= -\delta_\uparrow\big(-\overline{\delta}_\downarrow(\overline{T}) - \Delta^+\big) + \Delta^+ \\ &= -\delta_\uparrow\big(-\big(\delta_\downarrow(\overline{T} - \Delta^-) - \Delta^-\big) - \Delta^+\big) + \Delta^+ \\ &= -\delta_\uparrow\big(-\delta_\downarrow(\overline{T} - \Delta^-) + \Delta^- - \Delta^+\big) + \Delta^+.\end{aligned}$$

If we substitute $T = \overline{T} - \Delta^-$ in the last line, we arrive at

$$T - (\Delta^+ - \Delta^-) = -\delta_\uparrow\big(-\delta_\downarrow(T) - (\Delta^+ - \Delta^-)\big), \tag{11}$$

which confirms (3).

Doing the same for the reversed involution property of c, provides

$$\begin{aligned}\overline{T} &= -\overline{\delta}_\downarrow\big(-\overline{\delta}_\uparrow(\overline{T})\big) \\ &= -\delta_\downarrow\big(-\overline{\delta}_\uparrow(\overline{T}) - \Delta^-\big) + \Delta^- \\ &= -\delta_\downarrow\big(-\big(\delta_\uparrow(\overline{T} - \Delta^+) - \Delta^+\big) - \Delta^-\big) + \Delta^- \\ &= -\delta_\downarrow\big(-\delta_\uparrow(\overline{T} - \Delta^+) + \Delta^+ - \Delta^-\big) + \Delta^-.\end{aligned}$$

If we substitute $T = \overline{T} - \Delta^+$ in the last line, we arrive at

$$T + (\Delta^+ - \Delta^-) = -\delta_\downarrow\big(-\delta_\uparrow(T) + \Delta^+ - \Delta^-\big), \tag{12}$$

which confirms (4).

□

Eq. (2) implies that $\delta_\uparrow(.)$ resp. $\delta_\downarrow(.)$ are the result of shifting $\overline{\delta}_\uparrow(.)$ resp. $\overline{\delta}_\downarrow(.)$ along the 2nd median by Δ^+ resp. Δ^-. It is apparent from Fig. 4, though, that the choice of Δ^+, Δ^- cannot be arbitrary, as it restricts the range of feasible values for T via the domain of $\overline{\delta}_\uparrow(.)$ resp. $\overline{\delta}_\downarrow(.)$ (see Definition 10 for further details).

Applying the analog channel model Fig. 2 to a PI channel suggests the following changes to the internal architecture: First, we add a (single-input, single-output) zero-time Boolean gate G followed by a pure delay shifter $\Delta^{+/-}$ before the pure delay unit $\overline{\delta}_{min}$, and split the comparator at the end into a *thresholder unit Th* and a cancellation unit C. The former outputs, for each transition on u_d, a corresponding V_{th}-crossing time of u_r, independently of whether it will actually be reached. For subthreshold pulses, the transition might even be scheduled in the past. The *cancellation unit C* only propagates transitions that are in the correct temporal order.

Unfortunately, in the usual case of $\Delta^+ < 0$ or $\Delta^- < 0$, the output of $\Delta^{+/-}$ may reverse the temporal order of two consecutive input transitions. The slope delimiter, however, is only defined on traces encoded via the alternating Boolean signal transitions' *Waveform Switching Times (WST)*, which must be strictly increasing.

Our solution is to move both $\Delta^{+/-}$ and C to the front of our PI channel. Whereas this solves the above problem, it requires that transitions exchanged between PI channels must use the *Threshold Crossing Times (TCT)* encoding. It provides, in sequential order, the points in time when the analog switching waveform would have crossed V_{th} (it is not required that it actually does). Since a

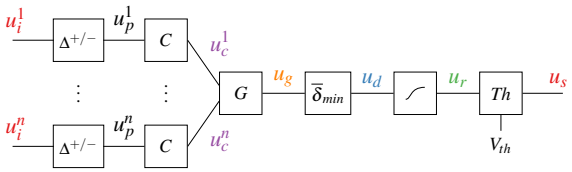

Figure 5: Channel model for the CIDM.

signal in TCT format also exposes canceled transitions, this architectural change inherently solves challenge (III) described in the introduction.

Now we are finally ready to formally define a Composable Involution Delay Model (CIDM) channel (see Fig. 5). Note that, although a PI channel differs by its internal structure significantly from the CIDM channel, they are equivalent with respect to Theorem 7.

Definition 8. A CIDM channel is the succession of one pure delay shifter $\Delta^{+/-}$ and cancellation unit C per input, a Boolean gate G, a pure-delay unit $\overline{\delta}_{min}$, a shaping unit and a thresholding unit Th.

The main practical advantage of a CIDM channel, which is a generalization of an IDM channel (just set $\Delta^- = \Delta^+ = 0$), is the additional degree of freedom for gate characterization in conjunction with the fact that a single channel encapsulates a single gate.

5 GLITCH PROPAGATION IN THE CIDM

Since CIDM channels do not satisfy the involution property, the question about faithful glitch propagation arises. After all, the proof of faithfulness of IDM [5] rests on the continuity of IDM channels, which has been proved only for involution delay functions. In this section, we will show that, for every modeling of a circuit with our CIDM channels, there is an equivalent modeling with IDM channels. Consequently, faithfulness of the IDM carries over to the CIDM.

For this purpose, we consider two successive CIDM channels and investigate the *logical channel*, i.e., the interconnection between two gates G_1 and G_2 as shown in Fig. 6. For conciseness, we integrate $\overline{\delta}_{min}$, the slew-rate limiter and the scheduler S in a new block *DSS*, and $\Delta^{+/-}$ followed by *Th* in the new block *PTh*. Using this notation, the logical channel consists of the *DSS* block of the predecessor gate G_1 and the *PTh* block of the successor gate G_2. Overall, this is just an IDM channel followed by a pure delay shifter, which will be denoted in the sequel as *IP channel*. The following Theorem 9 proves the somewhat surprising fact that every IP channel satisfies the properties of an involution channel:

Theorem 9 (IP channel properties). *Consider an IP channel formed by an involution channel given via $\overline{\delta}_\uparrow(.)$, $\overline{\delta}_\downarrow(.)$, followed by a pure delay shifter (Δ^+, Δ^-) with $\Delta^+, \Delta^- \in \mathbb{R}$. Then, it is an involution channel, characterized by some delay functions $\delta_\uparrow(.)$, $\delta_\downarrow(.)$.*

Proof. Consider an input signal consisting of a single negative pulse, as depicted in Fig. 1. Let t_i' resp. t_i be the time of the falling resp. rising input transition, t_c' resp. t_c the time of the falling resp. rising transition at the output of the involution channel, and t_o' resp. t_o the time of the falling resp. rising transition after the pure delay shifter. With $\overline{T} = t_i - t_c'$, we get $\overline{\delta}_\uparrow(\overline{T}) = t_c - t_i$ as well as $t_o' = t_c' + \Delta^-$ and $t_o = t_c + \Delta^+$.

By setting $T = t_i - t_o' = t_i - t_c' + t_c' - t_o' = \overline{T} - \Delta^-$ for the delay function $\delta_\uparrow(T)$ of the IP channel we find

$$\begin{aligned}\delta_\uparrow(T) &= t_o - t_i = t_o - t_c + t_c - t_i = \Delta^+ + \overline{\delta}_\uparrow(\overline{T}) \\ &= \Delta^+ + \overline{\delta}_\uparrow(T + \Delta^-). \end{aligned} \tag{13}$$

By analogous reasoning for a an up-pulse at the input, which results in the same equations as above with Δ^+ exchanged with Δ^- and $\overline{\delta}_\uparrow(\overline{T})$ with $\overline{\delta}_\downarrow(\overline{T})$, we also get

$$\begin{aligned}\delta_\downarrow(T) &= t_o - t_i = t_o - t_c + t_c - t_i = \Delta^- + \overline{\delta}_\downarrow(\overline{T}) \\ &= \Delta^- + \overline{\delta}_\downarrow(T + \Delta^+). \end{aligned} \tag{14}$$

Equations (13) and (14) are equivalent to

$$\overline{\delta}_\uparrow(\overline{T}) = \delta_\uparrow(\overline{T} - \Delta^-) - \Delta^+ \tag{15}$$

$$\overline{\delta}_\downarrow(\overline{T}) = \delta_\downarrow(\overline{T} - \Delta^+) - \Delta^- \tag{16}$$

which can be used in the involution property of $\overline{\delta}_\uparrow$ and $\overline{\delta}_\downarrow$ to achieve

$$\begin{aligned}\overline{T} &= -\overline{\delta}_\uparrow\big(-\overline{\delta}_\downarrow(\overline{T})\big) \\ &= -\delta_\uparrow\big(-\overline{\delta}_\downarrow(\overline{T}) - \Delta^-\big) + \Delta^+ \\ &= -\delta_\uparrow\big(-\big(\delta_\downarrow(\overline{T} - \Delta^+) - \Delta^-\big) - \Delta^-\big) + \Delta^+ \\ &= -\delta_\uparrow\big(-\delta_\downarrow(\overline{T} - \Delta^+)\big) + \Delta^+ \end{aligned} \tag{17}$$

which confirms that the IP channel is indeed an involution channel. □

We note that δ_{min} of the IP channel is usually different from $\overline{\delta}_{min}$ of the constituent IDM channel. Indeed, (13) above shows that δ_{min} is defined by $\delta_{min} - \Delta^+ = \overline{\delta}_\uparrow(-\delta_{min} + \Delta^-)$, for example, which reveals that we may (but need not) have $\delta_{min} \neq \overline{\delta}_{min}$. In addition, the IP channel is strictly causal only if Δ^+, Δ^- satisfy certain conditions: From (13) and (14) and the required conditions $\delta_\uparrow(0) > 0 \Leftrightarrow \delta_\downarrow(0) > 0$, we get

$$\delta_\uparrow(0) = \Delta^+ + \overline{\delta}_\uparrow(\Delta^-) > 0 \Leftrightarrow \delta_\downarrow(0) = \Delta^- + \overline{\delta}_\downarrow(\Delta^+) > 0. \tag{18}$$

At this point, the question arises whether it can be ensured that the logical channels in Fig. 6 are strictly causal. The answer is yes, provided the interconnected gates are *compatible*, in the sense that the joined *PTh* block of G_2 and the *DSS* block of G_1 are compatible w.r.t. Observation 5. More specifically, the pure delays Δ^+, Δ^- embedded in *PTh* of G_2 should ideally match (the switching waveforms of) the *DSS* block in G_1: According to (1), $-\Delta^+$ is the time the rising input waveform of G_2 requires to change from V_{th}^{in*} to the actual threshold voltage V_{th}^{in}, while $-\Delta^-$ denotes the same for the falling input. Assuming $\Delta^+ > 0$ and $\Delta^- < 0$, it can be seen from (14) that the overall delay function for falling transitions is derived by shifting the original one to the left and downwards (cp. Fig. 4). Note that the 2nd median is crossed at the same location since $\overline{\delta}_\downarrow(\delta_{min} + \Delta^+) = \overline{\delta}_{min} - \Delta^-$, and thus results in $\overline{\delta}_{min} = \delta_{min} > 0$ (which implies causality). The case of $\Delta^+ < 0$ and $\Delta^- > 0$ can be argued analogously, starting from (13).

These considerations justify the following definition:

Definition 10 (Compatibility of CIDM channels)**.** Two interconnected CIDM channels are called *compatible*, if the logical channel between them is strictly causal.

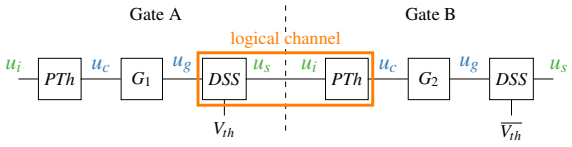

Figure 6: Channel model for proofs of the CIDM. Signals in blue have data type WST, those in green VCT.

It follows that every chain of gates properly modeled in the CIDM can be represented by a chain of Boolean gates interconnected by strictly causal IDM channels, with a "dangling" *PTh* block at the very beginning and a *DSS* block at the very end. Whereas the latter is just an IDM channel, the former only forms a valid channel when combined with the *DSS* block of the gate that drives the input port. For an "outermost" input port of a circuit, we can just require that the connected gate must have a threshold voltage matching the external input signal, such that $\delta^{\uparrow}_{min} = \delta^{\downarrow}_{min} = 0$ for the dangling *PTh* component. As a consequence, all the results and all the machinery developed for the original IDM [5] could, in principle, be also applied to circuits modeled with CIDM channels.

6 SIMULATING EXECUTIONS OF CIRCUITS

In this section, we provide Algorithm 1 for timing simulation in the CIDM. Since our algorithm is supposed to be run in the Involution Tool [15], which utilizes Mentor® ModelSim® (version 10.5c), our implementation had to be adapted to its internal restrictions.

The general idea is to replace the gates from standard libraries by custom gates represented by CIDM channels. According to Fig. 5, a custom gate C consists of three main components: (i) a pure delay shifter $P_I = \Delta^{+/-}$ for each input I (cancellation is done automatically by ModelSim), (ii) a Boolean function representing the embedded gate G, and (iii) an IDM channel c. Note that the output of G is in WST format, which facilitates direct comparison of the events (*evGO*) occurring in our CIDM simulation with the gate outputs obtained in classic simulations based on standard libraries.

Unfortunately, the input and output of a CIDM channel, i.e., the input of component P_I and the output of component c, is of type TCT, which is incompatible with discrete event simulations: In ModelSim signal transitions are represented by events, which are processed in ascending order of their scheduled time. Consequently, they cannot be scheduled in the past, which may, however, be needed for a transition (t_n, x_n, o_n) with $o_n < 0$.

Therefore, for every CIDM channel C we maintain a dedicated file $F(C)$, in which the simulation algorithm writes all the transitions (t_n, x_n, o_n) generated by C. The event $evTI(C)$ that ModelSim inherently assigns to the output signal of C is only used as a *transition indicator*: For every edge (C, I, Γ), it instantaneously triggers the occurrence of the "duplicated" ModelSim event $evTI(C, I, \Gamma)$, which signals to I of Γ the event that there is a new transition in the file $F(C)$. For an input port i, exactly the same applies, except that the input transitions in $F(i)$ and the transition indicator are externally supplied.

By contrast, both the inputs and output of the gate G embedded in a CIDM channel C, i.e., the output of component P_I and the input of component c, are of type WST. Consequently, we can directly use the ModelSim events $evGI(I, C)$ resp. $evGO(C)$ assigned to the output of P_I for input I of C resp. the output of G of C for conveying WST transitions. Still, cancellations that would cause occurrence times $t' < t_{now}$ must be prohibited explicitly (see Line 24 in Algorithm 1).

Our simulation algorithm uses the following functions:

- $(t, ev, Par) \leftarrow getNextEvent()$: Returns the event ev with the smallest scheduled time t and possibly some additional parameters Par. If t' denotes the time of the previous event, then $t \geq t'$ is guaranteed to hold. The possible types of events are $ev \in \{evTI(\Gamma, I, C), evGI(I, C), evGO(C)\}$, where C is the channel the event belongs to, I one of its inputs, and Γ the vertex that feeds I. If multiple different events are scheduled for the same time t, they occur in the order first $evTI(\Gamma, I, C)$, then $evGI(I, C)$ and finally $evGO(C)$. If multiple instance of the same event are scheduled for the same time t, then only the event that has been scheduled last actually occurs.
- $sched(ev, (t, x))$: Schedules a new event ev signaling the transition $x \in \{0, 1, toggle\}$ at time t; the case $x = toggle$ means $x = 1 - x'$ for x' denoting the last (= previous) transition. If the current simulation time is t_{now}, only $t \geq t_{now}$ is allowed.
- $init(ev, (-\infty, x))$: Initializes the initial state of the event ev, without scheduling it.
- $value \leftarrow s(ev, t)$: Returns the value of the state function for event $ev \in \{evGI(I, C), evGO(C)\}$ at time t.
- $F(C) \leftarrow (t, x, o)$: Adds a new TCT transition (t, x, o) generated by the output of C to the file $F(C)$, which buffers the TCT transitions of C.
- $(t', x) \leftarrow F(\Gamma)$ reads the most recently added TCT transition (t, x, o) from the file $F(\Gamma)$ and returns it as $(t', x) = (t + o, x)$.
- $Init(\Gamma)$: For both channels and input ports, the a fixed initial value $(-\infty, Init(\Gamma))$.
- $x \leftarrow G.f(x_1, \ldots, x_{G.d})$: Applies the combinatoric function $G.f$ (of the $G.d$-ary gate G embedded in some channel C) to the list of logic input values $x_1, \ldots, x_{G.d}$, and returns the result x.
- $\delta_{min} \leftarrow calcDelta(G.f, x, \Delta^+, \Delta^-)$ calculates the delay to be applied by the pure delay shifter. Note that it depends both on the gate's function $G.f$ and the transition type x.

Algorithm 1 conceptually starts at time $t = -\infty$ and first takes care of ensuring a clean initial state. The simulation of the actual execution commences at $t = 0$, where the conceptual "reset" that froze the initial state is released: Every channel whose initial state differs from the computation of its embedded gate in the initial state causes a corresponding transition of the gate output at $t = 0$, which will be processed subsequently in the main simulation loop.

The algorithm uses the procedure *calcDelta* for computing the actual Δ^+, Δ^- employed in the P_I component of a channel, which unfortunately depend on the embedded gate G: For an inverter, for example, a rising transition at the input leads to a falling transition at the output, so $\delta^{\downarrow}_{min}$ must be used. This is unfortunately not as easy for multi-input gates G, since the effect of any particular input transition *on the output* needs to be known in advance. This is difficult in the case of an XOR gate, for example, as it depends on the (future) state of the other inputs. Currently, we hence only support $\Delta^+ = \Delta^-$ for multi-input gates in our simulation algorithm.

Algorithm 1 CIDM circuit simulation algorithm

▷ Executed at simulation time $t = -\infty$:
for all channels C **do**
  $F(C) \leftarrow (-\infty, Init(C), 0)$ ▷ init file
  $init(evTI(C), (-\infty, 0))$ ▷ init toggle indicator
  $init(evGO(C), (-\infty, Init(C)))$ ▷ init gate output
  for all incoming edges (Γ, I, C) of C: **do**
    $init(evGI(I, C), (-\infty, Init(\Gamma)))$ ▷ init gate inputs
  end for
end for
▷ Executed at simulation time $t = 0$:
for all channels C, with d-ary gate G and incoming edges $(\Gamma_1, I_1, C), \ldots, (\Gamma_d, I_d, C)$ **do**
  $x = G.f(s(evGI(I_1, C), 0), \ldots, s(evGI(I_d, C), 0))$
  if $x \neq Init(C)$ **then**
    $sched(evGO(C), (0, x))$ ▷ add reset transition
  end if
end for
▷ Main simulation loop:
$(t, ev, Par) \leftarrow getNextEvent()$
while $t \leq \tau$ **do**
  if $ev = evTI(\Gamma, I, C)$ **then** ▷ $evTI$ go first
    $(\Delta^+, \Delta^-, G) \leftarrow Par$
    $(t', x) \leftarrow F(\Gamma)$
    $\delta_{min} \leftarrow calcDelta(G.f, x, \Delta^+, \Delta^-)$
    $sched(evGI(I, C), (\max\{t, (t' + \delta_{min})\}, x)$
  else if $ev = evGI(I, C)$ **then** ▷ $evGI$ come next
    $(G) \leftarrow Par$
    $x \leftarrow s(evGO(C), t)$ ▷ Current gate output
    $y \leftarrow G.f(s(evGI(I_1, C), t), \ldots, s(evGI(I_{G.d}, C), t))$
    if $x \neq y$ **then**
      $sched(evGO(C), (t, y))$
    end if
  else if $ev = evGO(C)$ **then** ▷ and finally $evGO$
    $(x, \delta_\uparrow(.), \delta_\downarrow(.), \Delta^+, \Delta^-) \leftarrow Par$
    $(t', x') \leftarrow F(C)$
    $T \leftarrow t - t'$
    if $x = 1$ **then**
      $o \leftarrow \delta_\uparrow(T - \Delta^+) - \Delta^+$
    else
      $o \leftarrow \delta_\downarrow(T - \Delta^-) - \Delta^-$
    end if
    $F(C) \leftarrow (t, x, o)$
    $sched(evTI(C), (t, toggle))$ ▷ triggers $evTI(C, I, \Gamma')$
  end if
  $(t, ev, Par) \leftarrow getNextEvent()$
end while
$postprocess()$

procedure $calcDelta(func, x, \Delta^+, \Delta^-)$
  if $func = not$ **then**
    if $x = 1$ **then return** Δ^-
    else return Δ^+
    end if
  else if $func = id$ **then**
    if $x = 1$ **then return** Δ^+
    else return Δ^-
    end if
  else
    assert($\Delta^+ = \Delta^-$)
    return Δ^+
  end if
end procedure

Theorem 11 shows that Algorithm 1 indeed computes valid executions for a circuit, provided all its logical channels are strictly causal. As argued in Section 5, this is the case if all interconnected CIDM channels are compatible. Its proof relies on a formal model of signals, circuits and executions and a suite of technical lemmas, which had to be omitted here due to lack of space but can be found in the extended version of our paper [9].

Theorem 11 (Correctness of simulation). *For any $0 \leq \tau < \infty$, the simulation Algorithm 1 applied to a circuit with compatible CIDM channels always terminates with a unique execution up to time τ.*

7 EXPERIMENTS

In this section, we validate our theoretical results by means of simulation experiments. This requires two different setups: (i) To validate the CIDM, we incorporated Algorithm 1 in our Involution Tool [15] and compared its predictions to other models. (ii) To establish the mandatory prerequisite for these experiments, namely, an accurate characterization of the delay functions, we employed a fairly elaborate analog simulation environment.

Relying on the 15 nm Nangate Open Cell Library featuring FreePDK15™ FinFET models [10] ($V_{DD} = 0.8$ V), we developed a Verilog description of our circuits and used the Cadence® tools Genus™ and Innovus™ (version 19.11) for optimization, placement and routing. We then extracted the parasitic networks between gates from the final layout, which resulted in accurate SPICE models that were simulated with Spectre® (version 19.1). These results were used both for gate characterization and as a golden reference for our digital simulations.

Like in [5], our main target circuit is a custom inverter chain. In order to highlight the improved modeling accuracy of CIDM, it consists of seven alternating high- and low-threshold inverters. They were implemented by increasing the channel length of p- respectively nMOS transistors, which varies the transistor threshold voltages [1, Fig. 2]. For comparison purposes, we conducted experiments with a standard inverter chain as well.

Regarding gate characterization for IDM, we used two different approaches. Recall from Observation 4 that fixing a single discretization threshold pins the value of all consistent δ_{min}, V_{th}^{in} and V_{th}^{out} throughout the circuit. In the variant of IDM called IDM*, we chose $V_{th}^{out*} = V_{DD}/2$ for the last inverter in the chain, and determined the actual value of its matching V_{th}^{in*} by means of analog simulations. To obtain consistent discretization thresholds for the whole circuit, we repeated this characterization, starting from $V_{th}^{out*} = V_{th}^{in*}$ for the next inverter up the chain. We thereby obtained values in the range $[0.301, 0.461]$ V, with $V_{th}^{in*} = 0.455$ V for the first gate. Obviously, characterizing a circuit in this fashion is very time-consuming, as only a single gate in a path can be processed at a time.

Alternatively, we also separately characterized every gate for $V_{th}^{out*} = V_{DD}/2$ and determined the matching V_{th}^{in*}, which we will refer to as IDM+. Note carefully that the discretization thresholds of connected gate out- and inputs differ for IDM+, such that an error is introduced at every interconnecting edge.

Although it is typically small and may even out, it can add up for larger circuits. Indeed, for the gates in our library, we recognized a clear bias towards $V_{th}^{in*} < V_{DD}/2$ for $V_{th}^{out*} = V_{DD}/2$. Finally, characterizing gates for CIDM was simply executed for $V_{th}^{out} = V_{th}^{in} = V_{DD}/2$.

The results for stimulating the standard inverter chain, with 2,500 normally distributed pulses of average duration μ and standard deviation σ, obtained by the Involution Tool for IDM*, IDM+, CIDM and the default inertial delay model, are shown in Fig. 7 (top). The accuracy of the model predictions are presented relative to our digitized SPICE simulations, which gets subtracted from the

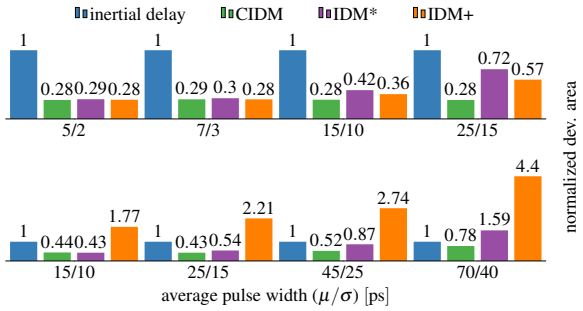

Figure 7: Accuracy, expressed as the normalized total deviation area of the digital predictions, relative to SPICE for the standard inverter chain (top) and high/low threshold inverter chain (bottom). Lower bars indicate better results.

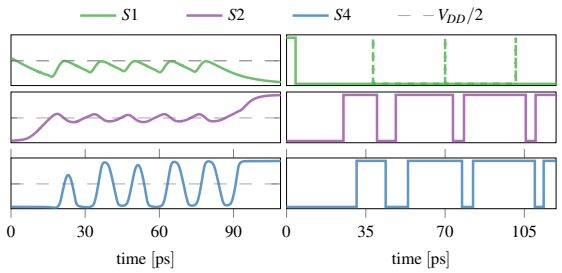

Figure 8: Recovering sub-threshold waveforms in an inverter chain using the CIDM.

trace obtained with a digital delay model. Summing up the area (without considering the sign), we obtain a metric that can be used to compare the similarity of two traces. Since the absolute values of the area are inexpressive, we normalize the results and use the inertial delay model as baseline.

For short pulses, IDM*, IDM+ and CIDM perform similarly. We conjecture that this is a consequence of the narrow range for V_{th}^{out*} and V_{th}^{in*} ([0.39156, 0.4] V), and therefore the induced error due to non-perfect matchings in IDM+ is negligible. For broader pulses, we observe a reduced accuracy of IDM* and IDM+, which is primarily an artifact of the imperfect approximation of the real delay function by the ones supported by the Involution Tool. We even observed settings, where CIDM does not even beat the inertial delay model, which can also be traced to this cause.

For our custom inverter chain [Fig. 7 (bottom)], CIDM outperforms, as expected, the other models considerably, whereas IDM+ occasionally delivers poor results, even compared to inertial delays. This is a consequence of the non-matching threshold values and the accumulating error. IDM* achieves much better predictions, but still falls short compared to CIDM. For broader pulses, CIDM and the inertial delay model perform similar, since they use the same maximum delay $\delta_\uparrow(\infty)$ and $\delta_\downarrow(\infty)$. The degradation of IDM* is once again a result of the imperfect delay function approximations.

Finally, analog simulations shown in Fig. 8 reveal, that sub-threshold pulses at some stage $S1$ can recover at the subsequent stage ($S2$) and even later in the chain ($S4$). Note that such a behavior is only faithfully modeled by CIDM, since IDM cannot propagate canceled transitions on signal $S1$ (dashed lines).

To summarize the results of our experiments, we highlight that the characterization procedure for IDM either requires high effort (IDM*) or may lead to modeling inaccuracies (IDM+). The CIDM clearly outperforms all other models w.r.t. modeling accuracy for our custom inverter chain, and is also the only model that can faithfully predict the "de-cancellation" of sub-threshold pulses.

8 CONCLUSIONS

We presented the Composable Involution Delay Model (CIDM), a generalization of the Involution Delay Model (IDM) that retains its faithful glitch-propagation properties. Its distinguishing properties are wider applicability, composability, easier characterization of the delay functions, and exposure of canceled pulse trains at interconnecting wires. Despite this considerable step forward towards a faithful delay model, there is still some room for improvement, in particular, for accurately modeling the delay of multi-input gates.

REFERENCES

[1] A. Asenov. 1998. Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 mu MOSFET's: A 3-D "atomistic" simulation study. *IEEE Transactions on Electron Devices* 45, 12 (1998), 2505–2513. https://doi.org/10.1109/16.735728

[2] Manuel J. Bellido-Díaz, Jorge Juan-Chico, and Manuel Valencia. 2006. *Logic-Timing Simulation and the Degradation Delay Model.* Imperial College Press, London.

[3] Cadence Design Systems 2015. *Effective Current Source Model (ECSM) Timing and Power Specification.* Cadence Design Systems. Version 2.1.2.

[4] Matthias Függer, Thomas Nowak, and Ulrich Schmid. 2016. Unfaithful Glitch Propagation in Existing Binary Circuit Models. *IEEE Trans. Comput.* 65, 3 (March 2016), 964–978. https://doi.org/10.1109/TC.2015.2435791

[5] M. Függer, R. Najvirt, T. Nowak, and U. Schmid. 2020. A Faithful Binary Circuit Model. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 10 (2020), 2784–2797. https://doi.org/10.1109/TCAD.2019.2937748

[6] Himanshu Gupta and Bahniman Ghosh. 2014. Transistor size optimization in digital circuits using ant colony optimization for continuous domain. *International Journal of Circuit Theory and Applications* 42, 6 (2014), 642–658. https://doi.org/10.1002/cta.1879 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cta.1879

[7] W. Ibrahim and V. Beiu. 2009. Reliability of NAND-2 CMOS gates from threshold voltage variations. In *2009 International Conference on Innovations in Information Technology (IIT).* 135–139. https://doi.org/10.1109/IIT.2009.5413631

[8] Jürgen Maier. 2017. *Modeling the CMOS Inverter using Hybrid Systems.* Technical Report TUW-259633. E182 - Institut für Technische Informatik; Technische Universität Wien.

[9] Jürgen Maier, Daniel Öhlinger, Ulrich Schmid, Matthias Függer, and Thomas Nowak. 2021. A Composable Glitch-Aware Delay Model. arXiv:2104.10966 [cs.OH]

[10] Mayler Martins, Jody Maick Matos, Renato P. Ribas, André Reis, Guilherme Schlinker, Lucio Rech, and Jens Michelsen. 2015. Open Cell Library in 15Nm FreePDK Technology. In *Proceedings of the 2015 Symposium on International Symposium on Physical Design* (Monterey, California, USA) *(ISPD '15).* ACM, New York, NY, USA, 171–178. https://doi.org/10.1145/2717764.2717783

[11] Tooraj Nikoubin, Poona Bahrebar, Sara Pouri, Keivan Navi, and Vaez Iravani. 2010. Simple Exact Algorithm for Transistor Sizing of Low-Power High-Speed Arithmetic Circuits. *VLSI Design* 2010, Article 3 (Jan. 2010), 7 pages. https://doi.org/10.1155/2010/264390

[12] J. Segura, J. L. Rossello, J. Morra, and H. Sigg. 1998. A variable threshold voltage inverter for CMOS programmable logic circuits. *IEEE Journal of Solid-State Circuits* 33, 8 (1998), 1262–1265. https://doi.org/10.1109/4.705367

[13] Synopsis Inc. 2016. *CCS Timing Library Characterization Guidelines.* Synopsis Inc. Version 3.4.

[14] Stephen H. Unger. 1971. Asynchronous Sequential Switching Circuits with Unrestricted Input Changes. *IEEE Transaction on Computers* 20, 12 (1971), 1437–1444.

[15] Daniel Öhlinger, Jürgen Maier, Matthias Függer, and Ulrich Schmid. 2021. The Involution Tool for Accurate Digital Timing and Power Analysis. *Integration* 76 (2021), 87 – 98. https://doi.org/10.1016/j.vlsi.2020.09.007