A technique for verifying nonlinear analog and mixed-signal circuits with inputs

Chuchu Fan [1], Yu Meng [1], Jürgen Maier [2], Ezio Bartocci [2], Sayan Mitra [1], Ulrich Schmid [2]
[1] {cfan10, yumeng5, mitras}@illinois.edu
University of Illinois at Urbana-Champaign
[2] {jmaier, s}@ecs.tuwien.ac.at, ezio.bartocci@tuwien.ac.at
Technische Universität Wien

©2018. This manuscript version is made available under the CC-BY-NC-ND 4.0 license http://creativecommons.org/licenses/by-nc-nd/4.0/

There has been progress in verification of nonlinear and hybrid systems in the recent years using algorithms that combine simulation data with model-based sensitivity analysis. These approaches only handle closed models, that is, models without inputs. The naïve introduction of models of input signals breaks these approaches, as typical inputs (fast sigmoids, discontinuous functions) for analog and mixed-signal circuits make the system highly sensitive and the number of needed simulations grow rapidly. In this paper, we present a new technique for verifying nonlinear and hybrid circuit models with inputs. A key result in the paper shows that once an input signal is fixed, the sensitivity analysis of the model can be performed much more precisely. Based on this observation, we extend a discrepancy-based verification algorithm and apply it to a suite of nonlinear and hybrid models of CMOS digital circuits under different input signals. The models are low-dimensional but involve highly nonlinear ODEs, with nearly hundreds of logarithmic and exponential terms, and therefore, have challenged existing verification approaches and tools. Our implementation of the new algorithm is able to verify these models; some of our experiments analyze the metastability of bistable circuits, which involve very sensitive ODEs. Our results not only demonstrate the feasibility of our approach but also provided interesting insights like the close connection between metastability recovery time and sensitivity.

I. Introduction

The field of analog and mixed-signal circuits has a synergistic relationship with formal verification of hybrid systems. The former has been a well-spring of hard problem instances and the latter has pushed the envelope of the rigorous analysis of complex properties of these circuits. Examples range from various analog and mixed-signal verification problems [5] to metastability analysis of digital circuits like arbiters [33].

Several important first generation verification algorithms were benchmarked using linear and linear hybrid models of circuits.The key idea behind all these techniques is to simulate dense bundles of trajectories or reachable states that enable exploration of all the possible behaviors of the circuit [20], [21]. Verification tools such as HyTech [23], PHAVer [17], SpaceEx [18], XSpeed [31], Checkmate [22], d/dt [5] and Coho [33] have demonstrated successful applications on different circuit models like tunnel-diode oscillators [26], a fairly complex $\Delta\Sigma$ modulator [22], [5], filtered oscillators [18], [31], and a digital arbiter circuit [33].

While these successes are encouraging, available approaches also have major shortcomings: Realistic circuit models are (highly) non-linear, which makes linear models questionable in general, and causes the linear approximation-based methods to be too conservative. Dealing with nonlinear models was beyond the capabilities of verification tools until recently. However, tools such as Flow* [4], NLTOOLBOX [6], HySAT/iSAT [15], [16], dReach [25], C2E2 [10], [14] and CORA [1], have demonstrated the feasibility of verifying nonlinear and nonlinear hybrid models. Most of these tools are still limited in terms of the complexity of the models, requiring manual tuning of algorithmic parameters, and, last but not least, by the type of external inputs they can handle.

The challenges in verifying circuit models are further exacerbated by the fact that certain circuit verification problems require state exploration in regions, where the model equations are very sensitive. One example is metastability of digital circuits [30]: It is well-known that every bi-stable circuit like a storage element or a flip-flop can be driven into a metastable state. In such a state, the circuit may output signals in the forbidden region between logical 0 and logical 1 (or very high-frequency oscillations) for an arbitrary time, before it resolves to a proper state again. The nature of this phenomenon suggests that the model is very sensitive in the metastable region, which is indeed confirmed by our approach.

In this paper, we take a step in analyzing complex nonlinear models with external inputs, and demonstrate the feasibility of our algorithms by performing bounded model-checking of circuit models with unprecedented complexity (over one hundred logarithmic and exponential terms in differential equations): All our circuits are modeled using either a custom hybrid or a uniform model; the latter integrates the different states of the hybrid model into a single non-linear ODE.

We build-up on previous work that combines numerical simulation data with model-based sensitivity analysis for verification. Consider a nonlinear ODE $\dot{x} = f(x)$, with a set of initial states $\Theta \subseteq \mathbb{R}^n$, a time bound T, and a set of unsafe states $\texttt{unsafe}$. A solution of the system is a function $\xi : \mathbb{R}^n \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$, such that for any initial state $x_0 \in \mathbb{R}^n$ and time t, $\xi(x_0, t)$ satisfies the ODE. For bounded invariant verification, we need to check whether or not the set $\texttt{Reach}(\Theta, [0, T])$ of reachable states from Θ, up to time T, intersects with $\texttt{unsafe}$. In general, $\texttt{Reach}(\Theta, [0, T])$ is hard to compute exactly; the data-driven verification approach over-approximates it using numerical solutions from a finite number of initial states in Θ, and enlarging these simulations by a factor determined by the sensitivity of the solutions to initial states. This notion of sensitivity is formalized as a *discrepancy function* and the resulting algorithms are shown to be sound, complete with respect to robust invariant verification [9], and extensible to nonlinear hybrid models [10], [14].

Highly sensitive circuit models create the orthogonal problem of properly handling external input signals: For a typical digital circuit like a simple inverter, the output trajectory V_{out} depends severely on the input trajectory V_{in}. The naïve approach for handling external inputs, namely, making the system closed by considering input signals as additional state variables, does not work in general: Our discrepancy functions would result in conservative over-approximations of the reachable sets in the case of highly sensitive systems, which often fill up large portion of the state space.

We address the sensitivity issue by introducing a new technique to compute discrepancy which separates input from state variables: As we observed that, with fixed input signals, the sensitivity can be computed much more precisely for state variables, we provide a method for computing the discrepancy function for systems with fixed input signals. This approach is implemented in a new version of the C2E2 tool [14] for verifying circuits with fixed inputs through reach set over-approximations, and it preserves soundness and completeness of the original verification algorithm for nonlinear hybrid models. Owing to this extension, we can stimulate and verify sensitive models with arbitrary input signals.

We demonstrate the feasibility of our approach by verifying a number of digital circuits: We evaluated inverter, NOR-gate and OR-gate using both ramp and sigmoid inputs. The ramp input has constant derivatives for the up and down slope of the pulse, while the sigmoid input smooths the non-differentiable states by using polynomial differential equations to approximate the sigmoid function. As instances of very sensitive models, we also experimented with a storage loop consisting of two inverters in a feedback-loop, and an OR-gate with its output fed back to one of its inputs. The latter allows to memorize a rising transition on its second input, and is an ideal target for demonstrating the capability of C2E2 to even predict metastable behavior correctly. Our results demonstrate that the new C2E2 can indeed be used to verify reachability problems for such circuits: They confirm what is known about metastable behavior, and provide detailed insights into the close connection between sensitivity and metastability recovery (time). Owing to these findings, we are convinced that the tool will open up a promising new avenue for studying such complex, rare and highly transient events in digital circuits.

The rest of the paper is organized as follows: In Section II, we introduce the model and the verification problem. In Section III, we discuss a novel approach to compute the discrepancy function for systems with fixed inputs and in Section IV, we give a brief review of the simulation-driven verification algorithm using discrepancy. In Section V, we discuss modeling of CMOS circuits, and in Section VI we present verification results. Conclusions and directions of future research are provided in Section VII.

II. Background

First we introduce notations used throughout the paper. $\mathbb{R}$ denotes the set of real numbers. For a vector $x \in \mathbb{R}^n$, $\|x\|$ denotes its l^2 norm. For a set $S \subset \mathbb{R}^n$, the diameter of S is the supremum of the distance between any pair of points in S: $dia(S) = \sup_{x,x' \in S}(\|x - x'\|)$. For $\delta \geq 0$, $B_\delta(x)$ is the closed δ-ball around x, that is, the set $\{x' \in \mathbb{R}^n \mid ||x' - x|| \leq \delta\}$. This notion is extended to sets via $B_\delta(S) = \cup_{x \in S} B_\delta(x)$. $S \oplus S'$ means the Minkowski sum of two sets S and S'. For matrix $A \in \mathbb{R}^{n \times n}$, $(A)_{ij}$ denotes the entry on the i^{th} row and j^{th} column. $eig(A)$ is the (largest) eigenvalue of A.

For a pair of matrices $\underline{A}, \bar{A} \in \mathbb{R}^{n \times n}$ with $(\underline{A})_{ij} \leq (\bar{A})_{ij}$ for all $1 \leq i, j \leq n$, we define the *interval matrix* as the set of matrices:

$$[\underline{A}, \bar{A}] \triangleq \{A \in \mathbb{R}^{n \times n} | (\underline{A})_{ij} \leq (A)_{ij} \leq (\bar{A})_{ij}, 1 \leq i, j \leq n\}.$$

A. Dynamic systems with inputs

An n-dimensional *dynamic system* with m-dimensional *input* is described by an ordinary differential equation:

$$\dot{x}(t) = f(x(t), u(t)), \tag{1}$$

where $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a continuously differentiable function, and a compact set $\Theta \subseteq \mathbb{R}^n$ of *initial states*. The input is an integrable function $u : [0, \infty) \to \mathcal{U}$, where $\mathcal{U} \subset \mathbb{R}^m$ is a compact set. Given an input signal u, the *solution* or the *trajectory* of the system is given by a function $\xi_u : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$, such that for any initial state $x_0 \in \Theta$ and at any time $t \in \mathbb{R}_{\geq 0}$, $\xi_u(x_0, t)$ satisfies the differential equation (1). A state $x \in \mathbb{R}^n$ is said to be *reachable* if there exists a state $x_0 \in \Theta$ and a time $t \geq 0$ such that $\xi_u(x_0, t) = x$. The set of all reachable states over an interval of time $[0, t_1]$ with input u is denoted by $\texttt{Reach}_u(\Theta, [0, t_1])$. $\texttt{Reach}_u(\Theta, [t_1, t_1])$ is written as $\texttt{Reach}_u(\Theta, t_1)$ in brief.

Example 1. *Consider a cardiac oscillator model from [10]. The system is described by the time-invariant differential equations: $\dot{x}_1 = -x_1(x_1^2 + 0.9x_1 + 0.9) + 2x_2u + 1; \dot{x}_2 = x_1 - 2x_2$. We consider a smoothed (sigmoidal) pulse input u (see Figure 1 (*left*)) that is described by differential equation: $\dot{u} = u(1.8 - 1.5u) + 0.0015$ for the up ramp and $\dot{u} = -u(1.8 - 1.5u) - 0.0015$ for the down ramp. The corresponding trajectories and (over-approximation of) reach sets projected on $x_1(t), x_2(t)$ are also shown in the Figure.*

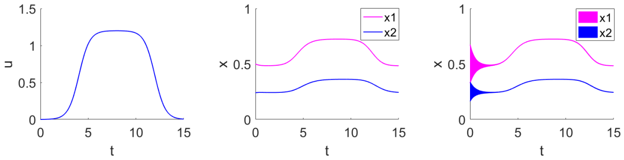

Fig. 1: Input signal $u(t)$ (*left*), corresponding trajectory of x_1, x_2 (*center*), and reach sets from $B_{0.1}([0.5, 0.24]^T)$

B. Safety verification problem

Given an n dimensional dynamic system, an input signal $u(t)$, a compact initial set $\Theta \in \mathbb{R}^n$, an unsafe set $\texttt{unsafe} \subseteq \mathbb{R}^n$, and a time bound $T > 0$, the safety verification problem is to check whether $\texttt{Reach}_u(\Theta, [0, T]) \cap \texttt{unsafe} = \emptyset$.

In general, reach set computation for nonlinear and hybrid models is difficult. *Simulation-driven verification algorithms* combine simulation data with model-based sensitivity analysis to answer the safety verification question with provable guarantees, as long as the problem itself has certain robustness properties [7], [9], [14], [10]. Sensitivity measures the changes in the system's trajectories to changes in the initial state. The simulation-driven algorithm first generates a set of trajectories from a finite set of the initial states Θ. Next, by bloating these simulations by an appropriately large factor using the sensitivity analysis, it computes an over-approximation of the reach set from Θ. Roughly speaking, the higher the sensitivity of the system to initial states, the larger is the number of simulations required to compute over-approximations with certain error bounds. If this over-approximation proves safety or produces a counter-example, then the algorithm decides, otherwise, it draws more samples of initial states and repeats the earlier steps to compute a more precise over-approximation. This approach is implemented in the safety verification tool C2E2 [10], [14] and has been used to successfully verify the safety of power-train control systems [8], parallel landing protocols, and several other complex nonlinear models.

The existing approaches, however, do not support the analysis of systems with inputs. If we explicitly model the input u and include it as a state variable in the closed system, the resulting model often turns out to be extremely sensitive with respect to the initial state. This is because the closed system with input u as state variable can be unstable as the input variable often models unstable behaviors like stiff changes in the pulse. In the case of Example 1, if we treat u as a state variable, the over-approximation reach set of $x_1(t)$ using C2E2 is shown in Figure 2. The blow-up in the over-approximation is due to the unstable input $\dot{u} = u(1.8 - 1.5u) + 0.0015$ that models the rising transition of the smoothed pulse.

This motivates us to separate input from state variables and to come up with methods to measure the sensitivity of the system with fixed inputs as described in the next section.

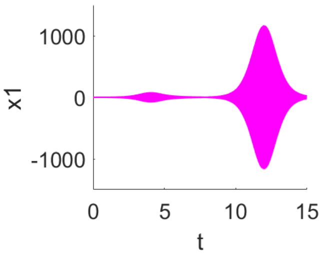

Fig. 2: Over-approximation of $x_1(t)$ without separating input from state variables

III. Sensitivity analysis for systems with fixed input

We formalize sensitivity using *discrepancy functions* as introduced in [9] and discuss methods for computing discrepancies for systems with fixed input described by Equation (1).

A. Discrepancy function for fixed input

Given a fixed input signal $u(t)$ for system (1), a discrepancy function bounds the distance between two neighboring trajectories, as a function of the distance between the initial states and the time. That is, given any two trajectories $\xi_u(x, t)$ and $\xi_u(x', t)$ of the system (1) starting from states x and x', respectively, with input $u(t)$, the discrepancy function β_u is a function of the distance between x and x', and time t. The distance between $\xi_u(x, t)$ and $\xi_u(x', t)$ is upper-bounded by the discrepancy function at t:

Definition 2. *Given an input signal* $u(t)$*, a continuous function* $\beta_u : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ *is a discrepancy function of the system in Equation* (1) *if*

(1) for any pair of states $x, x' \in \mathbb{R}^n$*, and any time* $t \geq 0$*,*

$$\|\xi_u(x, t) - \xi_u(x', t)\| \leq \beta_u(\|x - x'\|, t), \textit{and}$$

(2) for any t*,* $\lim_{\|x - x'\| \to 0^+} \beta_u(\|x - x'\|, t) = 0$*.*

Definition 2 generalizes the discrepancy functions defined in [13], [9]. According to the definition of discrepancy functions, for system (1) with input $u(t)$, at any time t, the ball centered at $\xi_u(x_0, t)$ with radius $\beta_u(\delta, t)$ contains the reach set of (1) starting from $B_\delta(x_0)$. Therefore, by bloating the simulation trajectories using the corresponding discrepancy function, we can obtain reach set over-approximations. As shown in [9], [28], several techniques (contraction metric [27], incremental stability [2], matrix measures [28], etc.) can be used to find discrepancy functions for dynamic systems without inputs. Input-to-state (IS) discrepancy functions as proposed in [24] take into consideration the sensitivity with respect to *different* input signals, and the method for computing IS-discrepancy presented in [13] introduces a multiplicative error factor of $e^{\frac{1}{2}t}$. This makes the over-approximation too conservative to be useful. The technique discussed in the next section will take advantage of the fact that the input signal is fixed for different trajectories starting from different initial states.

B. Computing discrepancy from Jacobian matrices

First, we introduce a basic result rooted in the high-order mean value theorem (Lemma 3) to connect the differential equation with its Jacobian matrices. Then we show that the terms of the Jacobian matrix with respect to the state variables are bounded over compact sets (Lemma 4). Using these two facts, we establish that the distance between neighboring trajectories actually follows a differential equation related to the bound of the Jacobian matrix (Lemma 5). Finally, we prove that the upper bound on the largest eigenvalue of the symmetric part of the Jacobian provides us with a suitable discrepancy function (Lemma 6).

The *Jacobian* of f with respect to the state x, $J_x : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^{n \times n}$, is a matrix-valued function of all the first-order partial derivatives of f with respect to the state components. Similarly, the Jacobian of f with respect to the input, $J_u(x, u)$, is a $n \times m$ matrix-valued function:

$$(J_x(x,u))_{ij} = \frac{\partial f_i(x,u)}{\partial x_j}; \quad (J_u(x,u))_{ij} = \frac{\partial f_i(x,u)}{\partial u_j}.$$

The Jacobian matrices for Example 1 are:

$$J_x(x,u) = \begin{bmatrix} -3x_1^2 - 1.8x_1 - 0.9 & 2u \\ 1 & -2 \end{bmatrix}; J_u(x,u) = \begin{bmatrix} 2x_2 \\ 0 \end{bmatrix}.$$

The following lemma relates f with its Jacobian matrices based on the generalized mean value theorem, see [13] for the detailed proof.

Lemma 3. *For any continuously differentiable vector-valued function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $x, r \in \mathbb{R}^n$ and $u, w \in \mathbb{R}^m$,*

$$\begin{aligned} & f(x+r, u+w) - f(x,u) = \\ & \left(\int_0^1 J_x(x+sr, u+w)ds\right) \cdot r + \left(\int_0^1 J_u(x, u+\tau w)d\tau\right) \cdot w, \end{aligned} \tag{2}$$

where the integral is component-wise.

If f is continuously differentiable, all terms in the Jacobian matrix are continuous. Since the input signals are bounded, i.e., $\forall t > 0, u(t) \in \mathcal{U} \subset \mathbb{R}^m$, the Jacobian matrix $J_x(x, u)$ over compact sets is also bounded:

Lemma 4. *If the Jacobian matrix of f in system* (1) *is $J_x(x,u)$, then for any compact sets S, $\mathcal{U}$ there exists an interval matrix $[\underline{A}, \bar{A}]$ such that*

$$\forall x \in S, u \in \mathcal{U}, J_x(x,u) \in [\underline{A}, \bar{A}].$$

The lemma follows from the fact that each term of $J_x(x, u)$ is a continuous function of x, u, and over the compact domains S,$\mathcal{U}$, the function has a maximum and minimum value that defines the matrix pair $[\underline{A}, \bar{A}]$. The bounds of such values can be obtained for a broad class of nonlinear functions using optimization and interval arithmetic solvers.

Lemma 5. *Fix an input signal $u(t)$ and an initial set Θ for system* (1)*. Suppose there exists a compact convex set $S \subseteq \mathbb{R}^n$ and a time interval $[0, t_1]$ such that for any $x \in \Theta$, $\forall t \in [0, t_1]$, $\xi_u(x, t) \in S$. Then for any $x, x' \in \Theta$, for any fixed $t \in [0, t_1]$, the distance $y_u(t) = \xi_u(x', t) - \xi_u(x, t)$ satisfies*

$$\dot{y}_u(t) = A(t)y_u(t),$$

for some $A(t) \in [\underline{A}, \bar{A}]$, where $[\underline{A}, \bar{A}]$ is an interval matrix such that $\forall x \in S, u \in \mathcal{U}, J_x(x,u) \in [\underline{A}, \bar{A}]$.

Lemma 5 can be proved by differentiating $y_u(t)$ using Lemma 3. The detailed proof can be found at Appendix A. Using the differential equation in Lemma 5, we can get a discrepancy function by bounding the eigenvalues of $[\underline{A}, \bar{A}]$:

Lemma 6. *Fix the input signal $u(t)$ for system* (1)*. Suppose the assumptions in Lemma 5 hold, and $\exists \gamma \in \mathbb{R}$ such that $\forall A(t) \in [\underline{A}, \bar{A}]$,*

$$eig(A^T(t) + A(t))/2 \leq \gamma; \tag{3}$$

then for any $x, x' \in \Theta$ and for any $t \in [0, t_1]$,

$$\|\xi_u(x,t) - \xi_u(x',t)\| \leq \|x - x'\|e^{\gamma t}.$$

Proof. Fixing the two initial states $x, x' \in \Theta$, from Lemma 5, we know that $\dot{y}_u(t) = A(t)y_u(t)$, for some $A(t) \in [\underline{A}, \bar{A}]$, where $y_u(t) = \xi_u(x', t) - \xi_u(x, t)$. By differentiating $\|y_u(t)\|^2$, we have that for any fixed $t \in [0, t_1]$,

$$\begin{aligned} \frac{d\|y_u(t)\|^2}{dt} &= \dot{y}_u^T(t)y_u(t) + y_u^T(t)\dot{y}_u(t) \\ &= y_u^T(t)\left(A(t)^T + A(t)\right)y_u(t). \end{aligned} \tag{4}$$

If $eig(A^T(t) + A(t))/2 \leq \gamma$, then the eigenvalues of $B = A^T(t) + A(t) - 2\gamma I$, where I is the identity matrix, are all less or equal to zero, so B is negative semi-definit. Therefore,

$$y_u^T(t)\left(A(t)^T + A(t) - 2\gamma I\right)y_u(t) \leq 0.$$

Hence, (4) becomes $\frac{d\|y_u(t)\|^2}{dt} \leq 2\gamma\|y_u(t)\|^2$.

After applying Grönwall's inequality, we have $\|y_u(t)\| \leq \|y_u(0)\|e^{\gamma t}, \forall t \in [0, t_1]$. □

Lemma 6 obviously provides a discrepancy function

$$\beta_u(\|x - x'\|, t) = \|x - x'\|e^{\gamma t},$$

and Algorithm 2 in [11] provides a method to compute such upper bound γ of the eigenvalues for an interval matrix $[\underline{A}, \bar{A}]$ using $O(n^2)$ time.

Note that the discrepancy is defined using the Euclidean (l^2) norm in Definition 2. In [12], a less conservative discrepancy function can be achieved by computing the optimal coordinate transformation for the Euclidean norm, which minimizes the upper bound γ for the transformed matrix. However, it takes extremely long time to solve such optimization problems for the complicated circuit models in this paper. We thus implemented a simple coordination transformation based on Jordan form decomposition method proposed in [13, Sec. 4.3] instead. Empirical results show that the latter method suffices to provide tight reach set over-approximations.

Example 7. *For Example 1, restrict x_1 to be within the range $[0.4, 0.6]$ and u to be within the range $[0.1, 0.2]$, then $J_x \in [\underline{A}, \bar{A}]$ where $\underline{A} = \begin{bmatrix} -3.06 & 0.2 \\ 1 & -2 \end{bmatrix}$ and $\bar{A} = \begin{bmatrix} -2.1 & 0.4 \\ 1 & -2 \end{bmatrix}$. Using Algorithm 2 in [11], we get that $\gamma = -1.05$ satisfies Equation* (3)*. Therefore, $\beta_u(\|x - x'\|, t) = \|x - x'\|e^{-1.05t}$ is a discrepancy function for this system with fixed input $u(t)$.*

IV. Verifying system with fixed inputs

In this section, we give a more detailed review of the simulation-driven verification algorithm. When implementing the above concepts, the representation of the trajectories are simulations, and the representation of the reach sets are reachtubes as defined below.

A. From simulations to reachable sets using discrepancy

Definition 8. *(Simulation) Fix the input $u(t)$, for any initial state $x \in \Theta$, for any $\epsilon, \tau, T > 0$, a (ϵ, τ, T)-simulation of the trajectory $\xi_u(x,t)$ of the system* (1) *is a sequence of time-stamped hyper-rectangles $\{(R_i, t_i)_{i=0}^k\}$ such that $\forall i = 0, 1, \ldots k, \forall j = 1, \ldots, k$: 1) $dia(R_i) \leq \epsilon$. 2) $0 < t_j - t_{j-1} \leq \tau$, $\xi_u(x, t_i) \in R_i$, and $\forall t \in (t_{j-1}, t_j)$, $\xi_u(x,t) \in \mathtt{hull}(R_{j-1}, R_j)$.*

A reachtube is also a sequence of time-stamped hyper-rectangles, this time containing all (and infinitely many) trajectories starting from the initial set Θ, however.

Definition 9. *(Reachtube) Fix the input $u(t)$, for any initial set $\Theta \subseteq \mathbb{R}^n$ and time bound $T > 0$, a (Θ, T)-reachtube is a sequence of time-stamped hyper-rectangles $\{(O_i, t_i)_{i=1}^k\}$, such that each $\mathtt{Reach}_u(\Theta, [t_{i-1}, t_i]) \subseteq O_i$.*

Lemma 5 assumes the existence of a compact set S that contains all the trajectories from initial set Θ up to time t_1. To get such coarse over-approximation S for system with input, we employ Proposition 4.1 of [24] based on the Lipschitz constant. S is often too conservative for being used directly as the over-approximation reachtube. Instead Lemma 6 refines the over-approximation by providing much tighter bounds on the distance between neighboring trajectories, even when starting out from a coarse over-approximation S.

Computing a single coarse over-approximation S for the entire time horizon $[0, T]$ is usually also too conservative. To mitigate this problem, we use the discrepancy function provided by Lemma 6 in a piece-wise fashion. We divide the time interval $[0, T]$ into smaller consecutive time intervals $[0, t_1], [t_1, t_2], \ldots, [t_{k-1}, T]$ and compute a piece-wise discrepancy function, where each piece is relevant for a smaller portion of the state and input space and time interval. That is, from initial set Θ at time 0, we first compute the reach set for the time interval $[0, t_1]$, $\mathtt{Reach}_u(\Theta, [0, t_1])$ using the discrepancy function from Lemma 6, then use the reach set at time t_1, $\mathtt{Reach}_u(\Theta, t_1)$ as the initial set for the time interval $[t_1, t_2]$ and so on. We refer readers to Algorithm 2 in [13] for the concrete treatment of constructing a reachtube from initial set Θ for the entire time horizon using the piece-wise discrepancy function.

B. Simulation-driven verification

Recall that our safety verification problem requires to check whether $\mathtt{Reach}_u(\Theta, [0, T]) \cap \mathtt{unsafe} = \emptyset$. If there exists some $\epsilon > 0$ such that $B_\epsilon(\mathtt{Reach}_u(\Theta, [0, T])) \cap \mathtt{unsafe} = \emptyset$, we say the system is *robustly safe* . That is, the system is robustly safe if all states in some envelope around the system behaviors are safe. If there exists some $\epsilon > 0, x \in \Theta$, such that $B_\epsilon(R_i) \subseteq \mathtt{unsafe}$ for some R_i in the simulation from x, $\{(R_i, t_i)\}_{i=0}^k$, we say the system is *robustly unsafe* .

Given the computed discrepancy function β_u for the system (1), the safety verification algorithm for (1) is provided as Algorithm 1 in Appendix B. It returns SAFE if the reach set $\mathtt{Reach}_u(\Theta, [0, T])$ has no intersection with the unsafe set, along with a robustly safe reachtube STB, or returns UNSAFE upon finding a counter-example, i.e., the simulation ψ that has some part fully contained in the unsafe region.

The detailed description of Algorithm 1 can also be found in Appendix B. It incorporates a function *Bloat*(ψ, δ, ϵ) (Line 6) that uses the discrepancy function β_u from Lemma 6 to expand the simulation ψ from x, and gives an over-approximation of $\mathtt{Reach}_u(B_\delta(x), [0, T])$. The function guarantees that the union of all the bloated simulations STB is an over-approximation of $\mathtt{Reach}_u(\Theta, [0, T])$, which leads to the soundness of the algorithm. According to Lemma 6, if δ gets finer (i.e., smaller), the value of the discrepancy function β_u becomes smaller (i.e., the reachtube is arbitrary close to the simulation), which guarantees that the algorithm always terminates. To sum up, the discrepancy function computed using Lemma 6 gives us two key properties of the algorithm [9]:

Theorem 10. *(Soundness and relative completeness). Given initial set Θ, unsafe set* $\mathtt{unsafe}$, *time bound T and fixed input $u(t)$ for system as described in* (1), *if Algorithm 1 using the discussed discrepancy function as in Lemma 6 returns safe or unsafe, then the system* (1) *is safe or unsafe, respectively. The Algorithm will always terminate whenever the system is either robustly safe or robustly unsafe.*

The same idea can be made working for switched and hybrid models (see [10] for details). The main complication is that, because of the over-approximations in the computed reach sets, we have to keep track of spurious mode changes. This is what is implemented in the new version of C2E2 that is used in Section VI for experiments.

V. Modeling of CMOS Circuits

Most digital circuits today are still manufactured using *complementary metal-oxide-semiconductor* (CMOS) technology, with transistor sizes down to below 20 nanometers. Modern digital simulation tools like Modelsim or NC-Verilog allow fast functional and timing analysis of complex circuits consisting of millions of transistors. Whereas the used delay models provide sufficient accuracy for many applications, critical parts of a circuit require a more careful analysis using analog simulations. A prominent tool for this purpose is *Spice*, as it handles very detailed transistor models, configured by hundreds of manufacturer-provided parameters. However, these models quickly reach their limits in terms of simulation complexity for circuits consisting of more than a few tens of transistors and/or signal traces beyond milliseconds in real-time.

In order to decrease simulation times, simplified models of CMOS transistors have been introduced [3]. Their size is significantly smaller compared to *Spice* models (at most six equations are required), which makes it possible to evaluate them using general purpose tools like *MATLAB*. Despite their reduced complexity these models can capture subtle phenomena like channel length modulation (CLM) and carrier velocity saturation.

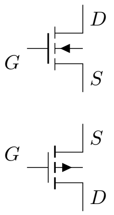

Fig. 3: NMOS (top) and PMOS (bottom) transistor

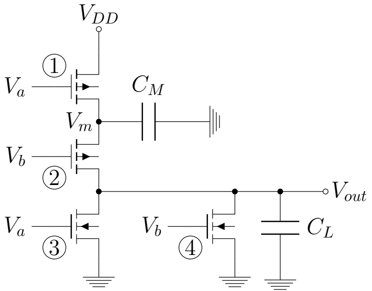

Fig. 4: Internal structure of NOR gate

In CMOS circuits, there are two different transistor types: NMOS and PMOS (see Figure 3), which differ in physical and, hence, electrical properties. Essentially, both deliver current based on the voltages applied to their gate (G), drain (D) and source (S) contact. In our model [29], every transistor can either operate in the sub-threshold (ST) region, where very little current is delivered, in the ohmic region (OHM), where the current scales linearly, and in the saturation region (SAT), where the current only changes moderately. The actual behavior within every region is described by a set of equations, which involve several fitting parameters. These parameters, which differ for NMOS and PMOS transistors, are either inferred directly from *Spice* model variables or fitted to *Spice* simulations.

A. Hybrid inverter model

The simplest CMOS gate is an inverter (shown in Appendix D as Figure 9), which consists of a PMOS transistor stacked above an NMOS one. Its output voltage V_{out} is the inverse of the input voltage V_{in}, ideally $V_{out} = V_{DD} - V_{in}$ where V_{DD} denotes the supply voltage. In reality, V_{out} is determined by

$$\dot{V}_{out} = \frac{1}{C_L} I_{out} \tag{5}$$

where C_L is the external load capacitance seen by the output. The output current I_{out} is the difference between the current delivered by the PMOS and the current consumed by the NMOS, which both depend upon V_{in} and V_{out}. As each of the two transistors can operate in three different regions, our basic hybrid inverter model has nine modes. As two of those modes are unreachable in reality, we finally arrive at the hybrid model shown in Appendix D Figure 10 (called InvHy in the sequel).

B. Uniform model

Given that the number of modes increases exponentially with the number of transistors in a circuit, it is natural to consider ways of avoiding multiple modes already in the transistor models: If the behavior of a transistor could be described by a single, possibly more complex equation that is valid for all operation regions, the need for a hybrid model vanishes altogether.

Our *uniform model* InvUni [29] accomplishes this by smoothening the boundaries between different regions by means of suitably chosen continuous functions. This results in a single non-linear equation, which describes the current through the transistor over the whole operation range. In conjunction with equation (5), this finally leads to a non-linear ODE that describes the behavior of V_{out} depending on V_{in} very accurately.

Apart from dramatically reduced model complexity, a key feature of our uniform model is the straightforward development of models for multi-transistor circuits like a NOR gate shown in Figure 4. In a hybrid model, it would blow up to a system of $3^4 = 81$ states; here, we end up with a system of two non-linear ODEs only:

$$\dot{V}_m = \tfrac{1}{C_M}(I_1 - I_2); \dot{V}_{out} = \tfrac{1}{C_L}(I_2 - I_3 - I_4)$$

Here, I_X represents the current through transistor ⓧ. The derivative of V_m, i.e., the current flowing to C_M divided by C_M, compare eq. (5), is just the difference between the current flowing through the PMOS transistors ① and ②. Note that C_M represents the capacitances of the transistor contacts only, and is thus several orders of magnitude smaller than C_L. The derivative of V_{out} is finally determined by the current passing through the lower PMOS ② minus the currents consumed by the NMOS transistors ③ and ④.

VI. Experiments and Results

For our experiments, we utilized the new version of C2E2, which implements the earlier described simulation and verification procedures, to analyze basic CMOS circuits[1]. Apart from analog waveform simulations, we used C2E2's key asset, the reach tubes, for verification purposes, which even included metastable behavior.

A. Input, simulation and verification

As external inputs we use ramp (Ramp) and sigmoidal signals (Sig), which are generated using two separate hybrid automata (see Appendix C); a 4 state one for Ramp and 2 state one for Sig. Typical trajectories of these generators for a supply voltage of $V_{DD} = 1.2$ V are shown in Figure 8 in Appendix C.

We evaluated InvHy, InvUni, NOR-gate and OR-gate by verification (parameters see Table I) and simulation. The first one uses the hybrid model presented in Section V-A, so we end up with $7 \times 4 = 28$ states in the Ramp case and $7 \times 2 = 14$ in the Sig case. All other circuits are based on

[1] The the tool and model files can be found at https://publish.illinois.edu/c2e2-tool/gate/.

TABLE I: Verification parameter settings

Verification parameters	Setting
Unsafe Set	$V_{out} > 1.32V$
Time Horizon	$6.4s$

the uniform model. The OR gate is easily derived from the NOR shown in Figure 4 in Section V-B, by appending an inverter. All the circuit models based on the uniform model have hundreds of logarithmic and exponential terms in their ODEs, and thousands of other arithmetic operations. We show the ODEs, verification parameters and reachtube of InvHy, InvUni, NOR and OR gates with Ramp and Sig inputs in Appendix D, E, F.

In addition to the above circuits, we also investigated a two inverter loop, where the input of one inverter is connected to the output of the other one, as shown in Appendix G. This circuit implements a simple state-holding device. In contrast to the other circuits used in our experiments, however, it does not have an external input. Consequently, we just set the output voltages to some initial values and let the circuit run.

The simulations behave as expected and show smooth output transitions even when activated by a ramp at its input. Verification shows that, despite initial state uncertainty, the traces quickly converge to a deterministic signal trace. Further results are summarized in Table II.

TABLE II: Verification of InvHy, InvUni, NOR-gate and OR-gate with Ramp (top) and Sig (bottom) input and InvLoop without input on a laptop with standard configuration (8G RAM, Intel Core i5 CPU). All verification results for the settings in Table I are safe.

Model	Verification parameters		Timing split [s]			time [s]
	Steps	Initial Set	Sim.	Discr.	I/O	
InvHy	128k	$V_{out} \in [1.15, 1.2]$	111	33	79	223
InvUni	64k	$V_{out} \in [1.15, 1.2]$	58	124	29	211
NOR	320k	$V_{out} \in [1.15, 1.2]$	396	1750	179	2325
OR	320k	$V_{nor} \in [1.199, 1.201]$ $V_{out} \in [0, 0.002]$	943	1722	148	2813
InvHy	128k	$V_{out} \in [1.15, 1.2]$	118	39	78	235
InvUni	64k	$V_{out} \in [1.15, 1.2]$	30	127	20	177
NOR	320k	$V_{out} \in [1.15, 1.2]$	168	1698	101	1967
OR	320k	$V_{nor} \in [1.199, 1.201]$ $V_{out} \in [0, 0.002]$	443	1778	89	2310
InvLoop	64k	$V_1 \in [1.0, 1.2]$ $V_2 \in [0.5, 0.6]$	27	224	5	256

B. Metastability

It is well known that any bistable digital circuit, ranging from a simple storage loop over a flip-flop to complex circuits with internal feedback, can be driven into a *metastable* state [30]. A circuit in such a state may output voltage values in the forbidden region between 0 and 1 or very high-frequency oscillation for an arbitrary time, before it resolves to a proper digital state again.

In order to demonstrate the capability of C2E2 to predict metastable behavior correctly, we use an OR-gate with its output fed back to one of its inputs, as shown in Appendix H. This circuit implements a storage loop, which is capable of memorizing a rising transition on its second input. It has been shown in [19] that it can be driven into a metastable state, namely, by an input pulse that is shorter than the delay of the feedback loop.

Figure 5 shows input (top), simulation traces (middle) and reachtube (bottom) of this circuit. The latter has been computed for the simulation trace that resolves latest. One can see that the reachtube blows up to several thousand Volts, which is physically impossible but indicates the very high sensitivity of the underlying system of ODEs in the metastable region: Even the slightest disturbances of the initial state results in very different trajectories, in particular, in very different metastability resolution times, after which V_{out} resolves to 0 or 1. Albeit this is completely in accordance with what is known about metastability, it is a unique feature of our approach to reveal the close relation to sensitivity and to even assess it quantitatively via the reachtube.

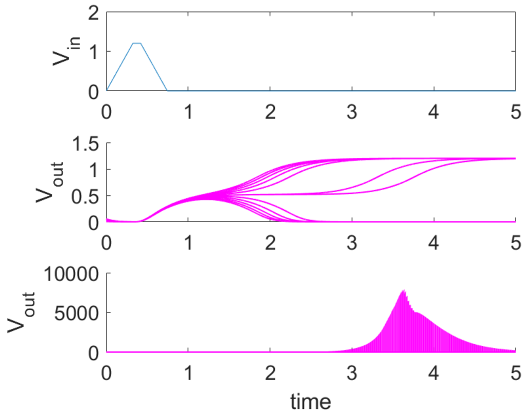

Fig. 5: Metastability analysis of fed back OR gate

Overall, our results confirm that the new C2E2 with support for discrepancy computation for systems with fixed inputs can indeed be used for simulation and verification of complex non-linear circuit models, and in fact opens up new avenues in challenging research areas like metastability analysis.

VII. Conclusions and Future Work

In this paper, we demonstrated that the new C2E2 provides a viable way to add arbitrary external inputs in the verification of highly sensitive non-linear ODEs, as it avoids the blow-up of the reachtubes in discrepancy-based reachability analysis of systems with inputs considered as additional state variables: We verified several basic CMOS circuits like inverter and NOR gate, based on both a hybrid and a complex uniform non-linear transistor model. Moreover, we also succeeded to verify the metastable behavior of a memory element, which demonstrates the ability of our approach to handle highly sensitive ODEs.

We did not yet perform a fair comparison to existing tools that can handle non-linear ODEs, like Flow*[2] and CORA, as our attempts to use these tools on our models

[2] In the case of Flow*, we were not able to even parse our circuit models

were not successful. In any case, however, these tools seem to have only limited ways to deal with external inputs. Besides the naïve way of incorporating inputs as additional state variables, CORA also allows inputs that take some value in a given set of values. Interestingly, in [33], Yan and Greenstreet proposed an alternative to the new C2E2 for handling external inputs in their verification of an arbiter circuit (that can become metastable): They allow to specify valid input signals via a Brockett annulus, which is the allowed region in the phase space $(V_{in}, \dot{V_{in}})$. Rather than a single input trajectory, this effectively specifies a whole set of allowed input trajectories. Since the used tool (Coho) uses linear differential inclusion and linear programming for computing reach sets, such inputs with uncertainty were relatively easy to incorporate. On the other hand, not surprisingly, the authors report excessive over-approximations in the case of stiff ODEs. By contrast, our approach considers input signals specified as functions only. However, extending our approach to inputs with bounded uncertainty would be a very interesting but non-trivial extension in the context of our discrepancy-based approach, and is hence a topic of future research.

Future work will also be devoted on reducing the execution time of C2E2, which turned out to be very long already for verifying the NOR gate. There are two main reasons for this: First, the ODEs of these systems are very complicated (around four hundreds logarithmic and exponential terms), costing the ODE solver and bloating algorithm much time on evaluating the function expressions. Moreover, the ODEs are inherently sensitive, forcing the time step to be very small. Apart from run time issue, some circuits (e.g., a NAND gate) even caused math range errors when computing the local discrepancy.

Such improvements will allow us to verify more complex circuits, as well as circuits based on more elaborate transistor models. Particular promising applications are envisioned in the area of advanced digital circuit analysis, where C2E2 could be used for verifying the metastable behavior, e.g. of Schmitt-Trigger circuit [32].

References

[1] M. Althoff and D. Grebenyuk. Implementation of interval arithmetic in CORA 2016. In *ARCH Workshop*, pages 91–105, 2016.

[2] D. Angeli. A Lyapunov approach to incremental stability properties. *Automatic Control, IEEE Transactions on*, 47(3):410–421, 2002.

[3] N. Arora. *MOSFET models for VLSI circuit simulation; theory and practice*. Computational microelectronics. Springer, Wien [u.a.], 1993.

[4] X. Chen, E. Ábrahám, and S. Sankaranarayanan. Flow*: An analyzer for non-linear hybrid systems. In *CAV*, pages 258–263. Springer, 2013.

[5] T. Dang, A. Donzé, and O. Maler. Verification of analog and mixed-signal circuits using hybrid system techniques. In A. J. Hu and A. K. Martin, editors, *FMCAD*, pages 21–36, 2004.

[6] T. Dang, C. Le Guernic, and O. Maler. Computing reachable states for nonlinear biological models. In *CMSB*, volume 5688 of *LNCS*, pages 126–141. Springer, 2009.

[7] A. Donzé. Breach, a toolbox for verification and parameter synthesis of hybrid systems. In *CAV*, pages 167–170. Springer, 2010.

[8] P. S. Duggirala, C. Fan, S. Mitra, and M. Viswanathan. Meeting a powertrain verification challenge. In *CAV*, pages 536–543. Springer, 2015.

[9] P. S. Duggirala, S. Mitra, and M. Viswanathan. Verification of annotated models from executions. In *EMSOFT*, page 26. IEEE Press, 2013.

[10] P. S. Duggirala, S. Mitra, M. Viswanathan, and M. Potok. C2E2: A verification tool for stateflow models. In *TACAS*, pages 68–82. Springer, 2015.

[11] C. Fan. Automatic simulation-driven reachability using matrix measures. *Master Thesis at University of Illinois at Urbana-Champaign*, 2016.

[12] C. Fan, J. Kapinski, X. Jin, and S. Mitra. Locally optimal reach set over-approximation for nonlinear systems. In *CAV*, page 6. ACM, 2016.

[13] C. Fan and S. Mitra. Bounded verification with on-the-fly discrepancy computation. In *ATVA*. Springer, 2015.

[14] C. Fan, B. Qi, S. Mitra, M. Viswanathan, and P. S. Duggirala. Automatic reachability analysis for nonlinear hybrid models with C2E2. In *CAV*, pages 531–538. Springer, 2016.

[15] M. Fränzle and C. Herde. Hysat: An efficient proof engine for bounded model checking of hybrid systems. *Formal Methods in System Design*, 30(3):179–198, 2007.

[16] M. Fränzle, C. Herde, T. Teige, S. Ratschan, and T. Schubert. Efficient solving of large non-linear arithmetic constraint systems with complex boolean structure. *JSAT*, 1(3-4):209–236, 2007.

[17] G. Frehse. Phaver: algorithmic verification of hybrid systems past hytech. *International Journal on Software Tools for Technology Transfer*, 10(3):263–279, 2008.

[18] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler. Spaceex: Scalable verification of hybrid systems. In *CAV*, pages 379–395. Springer, 2011.

[19] M. Függer, R. Najvirt, T. Nowak, and U. Schmid. Towards binary circuit models that faithfully capture physical solvability. In *DATE*, pages 1455–1460, San Jose, CA, USA, 2015. EDA Consortium.

[20] M. R. Greenstreet. Verifying safety properties of differential equations. In *CAV*, volume 1102 of *LNCS*, pages 277–287. Springer, 1996.

[21] M. R. Greenstreet and I. Mitchell. Reachability analysis using polygonal projections. In *HSCC*, volume 1569 of *LNCS*, pages 103–116. Springer, 1999.

[22] S. Gupta, B. H. Krogh, and R. A. Rutenbar. Towards formal verification of analog designs. In *ICCAD*, pages 210–217, Washington, DC, USA, 2004. IEEE Computer Society.

[23] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Hytech: A model checker for hybrid systems. In *CAV*, pages 460–463. Springer, 1997.

[24] Z. Huang and S. Mitra. Proofs from simulations and modular annotations. In *HSCC*, Berlin, Germany. ACM press.

[25] S. Kong, S. Gao, W. Chen, and E. Clarke. dReach: δ-reachability analysis for hybrid systems. In *TACAS*, pages 200–205. Springer, 2015.

[26] K. Lata and H. S. Jamadagni. Formal verification of tunnel diode oscillator with temperature variations. In *ASPDAC*, pages 217–222. IEEE Press, 2010.

[27] W. Lohmiller and J.-J. E. Slotine. On contraction analysis for nonlinear systems. *Automatica*, 34(6):683–696, 1998.

[28] J. Maidens and M. Arcak. Reachability analysis of nonlinear systems using matrix measures. *Automatic Control, IEEE Transactions on*, 60(1):265–270, 2015.

[29] J. Maier. Modeling the cmos inverter using hybrid systems. Technical Report TUW-259633, E182 - Institut für Technische Informatik; Technische Universität Wien, 2017.

[30] L. R. Marino. General theory of metastable operation. *IEEE ToC*, 30(2):107–115, 1981.

[31] R. Ray, A. Gurung, B. Das, E. Bartocci, S. Bogomolov, and R. Grosu. Xspeed: Accelerating reachability analysis on multi-core processors. In *HVC*, volume 9434 of *LNCS*, pages 3–18. Springer, 2015.

[32] A. Steininger, J. Maier, and R. Najvirt. The metastable behavior of a schmitt-trigger. In *ASYNC*, pages 57–64, May 2016.

[33] C. Yan and M. R. Greenstreet. Verifying an arbiter circuit. In *FMCAD*, pages 7:1–7:9, Piscataway, NJ, USA, 2008. IEEE Press.

Appendix

A. *Proof of Lemma 5*

Lemma 5. Fix an input signal $u(t)$ and an initial set Θ for system (1). Suppose there exists a compact convex set $S \subseteq \mathbb{R}^n$ and a time interval $[0, t_1]$ such that for any $x \in \Theta$, $\forall t \in [0, t_1]$, $\xi_u(x, t) \in S$. Then for any $x, x' \in \Theta$, for any fixed $t \in [0, t_1]$, the distance $y_u(t) = \xi_u(x', t) - \xi_u(x, t)$ satisfies

$$\dot{y}_u(t) = A(t) y_u(t),$$

for some $A(t) \in [\underline{A}, \bar{A}]$, where $[\underline{A}, \bar{A}]$ is an interval matrix such that $\forall x \in S, u \in \mathcal{U}, J_x(x, u) \in [\underline{A}, \bar{A}]$.

Proof. Using Lemma 3 we have the following:

$$\begin{aligned}
\dot{y}_u(t) &= \dot{\xi}_u(x', t) - \dot{\xi}_u(x, t) \\
&= f(\xi_u(x', t), u(t)) - f(\xi_u(x, t), u(t)) \\
&= \left(\int_0^1 J_x(\xi_u(x, t) + s y_u(t), u(t)) ds \right) \cdot y_u(t) \\
&+ \left(\int_0^1 J_u(\xi_u(x, t), u(t)) ds \right) \cdot (u(t) - u(t)) \\
&= \left(\int_0^1 J_x(\xi_u(x, t) + s y_u(t), u(t)) ds \right) \cdot y_u(t) \quad (6)
\end{aligned}$$

Given a compact convex set S and bounded set $\mathcal{U}$, the interval matrix $[\underline{A}, \bar{A}]$ satisfies the conditions in Lemma 4. For any fixed t, $\int_0^1 J_x(\xi_u(x, t) + s y_u(t), u(t)) ds$ is a constant matrix. Because $\xi_u(x, t), \xi_u(x', t)$ are contained in the convex set S, according to the convexity assumption of S, $\forall s \in [0, 1], \xi_u(x, t) + s y_u(t)$ is also contained in S. Thus, at t, $J_x(\xi_u(x, t) + s y_u(t), u(t)) \in [\underline{A}, \bar{A}]$. Since the integration is from 0 to 1, it is straightforward to check that also

$$\int_0^1 J_x(\xi_u(x, t) + s y_u(t), u(t)) ds \in [\underline{A}, \bar{A}].$$

We rewrite (6) as

$$\dot{y}_u(t) = A(t) y_u(t), A(t) \in [\underline{A}, \bar{A}], \tag{7}$$

which means that at any fixed time $t \in [0, t_1]$, we always have $\dot{y}_u(t) = A(t) y_u(t)$, where $A(t)$ is unknown but $A(t) \in [\underline{A}, \bar{A}]$. □

B. *Simulation-driven verification algorithm*

Function $Cover()$ returns a set of triples $\{\langle x, \delta, \epsilon \rangle\}$, where x's are sample states, the union of $B_\delta(x)$ covers Θ completely, and ϵ is the precision of the simulation. Function $Bloat()$ expands the simulation trace ψ by β_u to get the reachtube $\mathcal{R} = \{(O_i, t_i)\}_{i=1}^k$. That is, for each $i = 1, \ldots, k, O_i \leftarrow \mathtt{hull}(R_{i-1}, R_i) \oplus \max_{t \in [t_{i-1}, t_i]} \beta_u((\delta + \epsilon), t)$. From Lemma 6 it follows that $Bloat(\psi, \delta, \epsilon)$ contains $\mathtt{Reach}_u(B_\delta(x), [0, T])$. There are two important data structures used in Algorithm 1: $\mathcal{C}$ is a collection of the triples returned by $Cover()$, which represents the subset of Θ that has not yet been proved safe, and STB stores the bounded time reachtube.

Initially, $\mathcal{C}$ contains a singleton $\langle x_0, \delta_0 = dia(\Theta), \epsilon_0 \rangle$, where $\Theta \subseteq B_{\delta_0}(x_0)$ and ϵ_0 is a small positive constant. For each triple $\langle x, \delta, \epsilon \rangle \in \mathcal{C}$, the **while**-loop from Line 3 checks the safety of the reachtube from $B_\delta(x)$, which is computed in Line 5-6. ψ is a (ϵ, τ, T)-simulation from x with input $u(t)$, which is a sequence of time-stamped rectangles $\{(R_i, t_i)\}_{i=0}^k$ and is guaranteed to contain the trajectory $\xi(x, T)$ by Definition 8. Bloating the simulation result ψ by the discrepancy function β_u to get $\mathcal{R}$, a $(B_\delta(x), T)$-reachtube with input $u(t)$. If $\mathcal{R}$ is disjoint from $\mathtt{unsafe}$, then the reachtube from $B_\delta(x)$ is safe and the corresponding triple can be safely removed from $\mathcal{C}$. If for some j, R_j (one rectangle of the simulation) is completely contained in the unsafe set, then we can get a counterexample of a trajectory that violates the safety property. Otherwise, the safety of $\mathtt{Reach}_u(B_\delta(x), [0, T])$ is inconclusive and a refinement of $B_\delta(x)$ needs to be made with some smaller δ and smaller ϵ, τ.

Algorithm 1: Simulation-driven verification of systems with input

```
   input: $\Theta, u(t), T, \mathtt{unsafe}, \epsilon_0, \tau_0$
 1 $\delta \leftarrow dia(\Theta); \epsilon \leftarrow \epsilon_0; \tau \leftarrow \tau_0;$ STB $\leftarrow \emptyset$;
 2 $\mathcal{C} \leftarrow Cover(\Theta, \delta, \epsilon)$;
 3 while $\mathcal{C} \neq \emptyset$ do
 4     for $\langle x, \delta, \epsilon \rangle \in \mathcal{C}$ do
 5         $\psi = \{(R_i, t_i)\}_{i=0}^k \leftarrow Simulate(x, u, T, \epsilon, \tau)$;
 6         $\mathcal{R} \leftarrow Bloat(\psi, \delta, \epsilon)$;
 7         if $\mathcal{R} \cap \mathtt{unsafe} = \emptyset$ then
 8             $\mathcal{C} \leftarrow \mathcal{C} \backslash \{\langle x, \delta, \epsilon \rangle\}$ ;
 9             STB $\leftarrow$ STB $\cup \mathcal{R}$ ;
10         else if $\exists j, R_j \subseteq \mathtt{unsafe}$ then
11             return (UNSAFE, $\psi$)
12         else
13             $\mathcal{C} \leftarrow \mathcal{C} \cup Cover(B_\delta(x), \frac{\delta}{2}, \frac{\epsilon}{2}), \tau \leftarrow \frac{\tau}{2}$ ;
14         end
15     end
16 end
17 return (SAFE, STB);
```

C. *Hybrid automaton of input generators*

To generate the input signals we used hybrid automaton. The 4-state implementation generating the Ramp trace is shown in Figure 6. For Sig inputs it is possible to reduce the state count to 2, shown in Figure 7. Please note that the that the additive / subtractive term 0.005 is required to (1) achieve $\dot{V}_{in} \neq 0$ for $V_{in} = 0$ and (2) to reach the final value faster. Figure 8 shows the resulting analog waveforms.

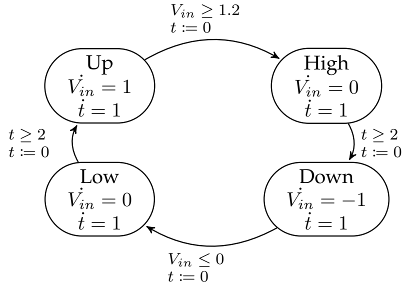

Fig. 6: Ramp input generator

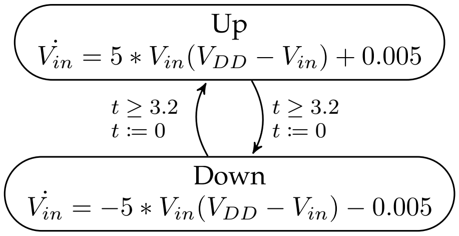
Up
$\dot{V_{in}} = 5 * V_{in}(V_{DD} - V_{in}) + 0.005$
$t \geq 3.2$
$t := 0$
$t \geq 3.2$
$t := 0$
Down
$\dot{V_{in}} = -5 * V_{in}(V_{DD} - V_{in}) - 0.005$

Fig. 7: Sig input generator

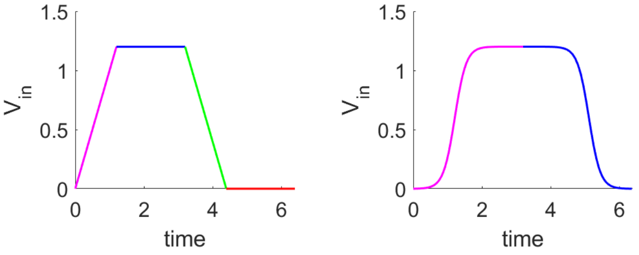

Fig. 8: Ramp (left) and Sig (right) input traces

D. *Inverter*

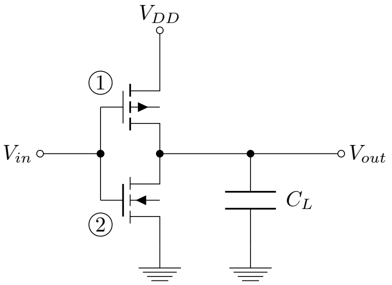

Fig. 9: Internal structure of CMOS inverter

Hybrid model automata:

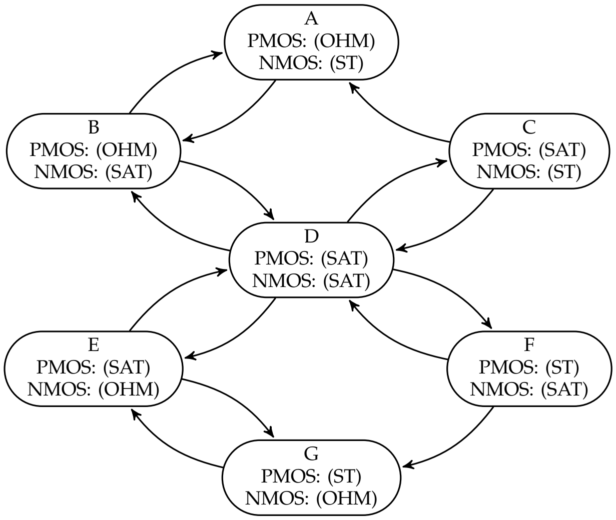

Fig. 10: Hybrid model automata of a CMOS inverter. The nodes represent the different modes, each involving specific transition guards and invariants. The label shows the operation regions of the transistors, the arcs indicate the possible transitions.

Uniform model:

$$\dot{V_{out}} = \frac{1}{C_L}(I_1 - I_2)$$

The results for InvHy and InvUni using (1) Ramp and (2) Sig inputs are shown in Figure 11 and 12. Colors indicate the different modes, where especially for InvHy many modes are traversed in quick succession. The reach tubes in both cases quickly converge as can be seen in the zoomed parts.

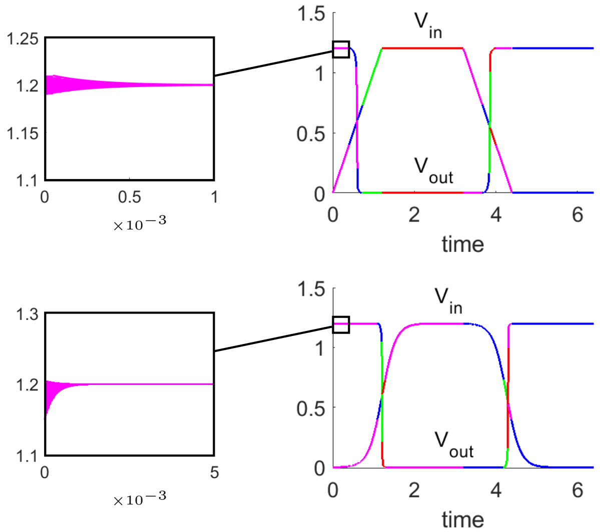

Fig. 11: InvHy output voltage over-approximation set for V_{in} = Ramp (top) and V_{in} = Sig (bottom)

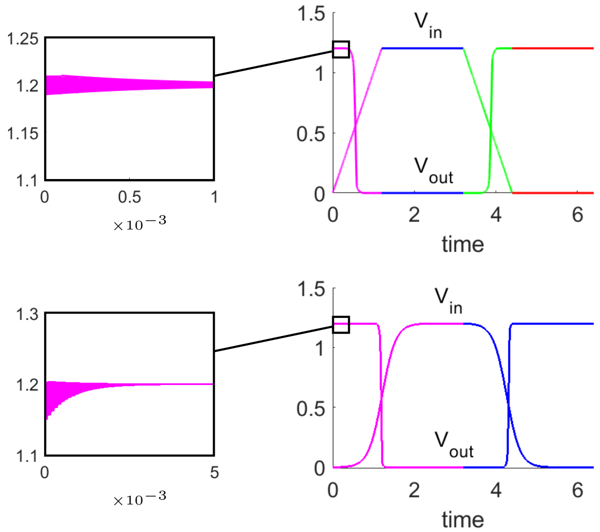

Fig. 12: InvUni output voltage over-approximation set for V_{in} = Ramp (top) and V_{in} = Sig (bottom)

E. NOR *Gate*

The circuit and uniform model implementation of this gate have already been presented in Section V-B. For our experiments we bound V_a to ground ($V_a = 0$) and connected $V_b = V_{in}$ to an input generator. In this configuration the output voltage should behave similarly to that of an inverter when applying the trace of V_{in}. Figure 13 shows that the results match our expectations with again quickly converging reach tubes.

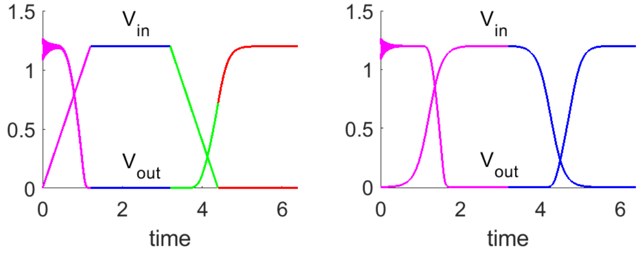

Fig. 13: NOR gate output voltage over-approximation set for V_{in} = Ramp (left) and V_{in} = Sig (right)

F. OR *gate*

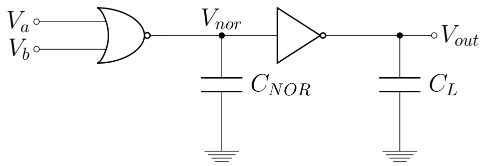

Fig. 14: Internal structure of OR gate

Uniform model:

For the equations we used the uniform models of NOR gate and inverter described earlier.

$$\dot{V}_m = \frac{1}{C_M}(I_{1,NOR} - I_{2,NOR})$$

$$\dot{V}_{nor} = \frac{1}{C_{NOR}}(I_{2,NOR} - I_{3,NOR} - I_{4,NOR})$$

$$\dot{V}_{out} = \frac{1}{C_L}(I_{1,INV} - I_{2,INV})$$

Just as what we did with the NOR gate, we bound one input to ground ($V_a = 0$) and applied the input signal to the other one. Since the difference between NOR and OR is just an additional inverter, we expect a buffer like behavior, i.e., that the output has the same shape as the input. The results shown in Figure 16 satisfy our expectations.

G. *Inverter loop*

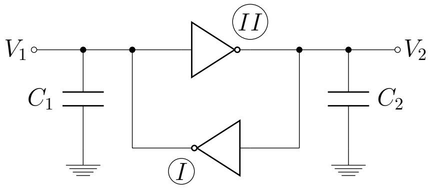

Fig. 15: Internal structure of an inverter loop

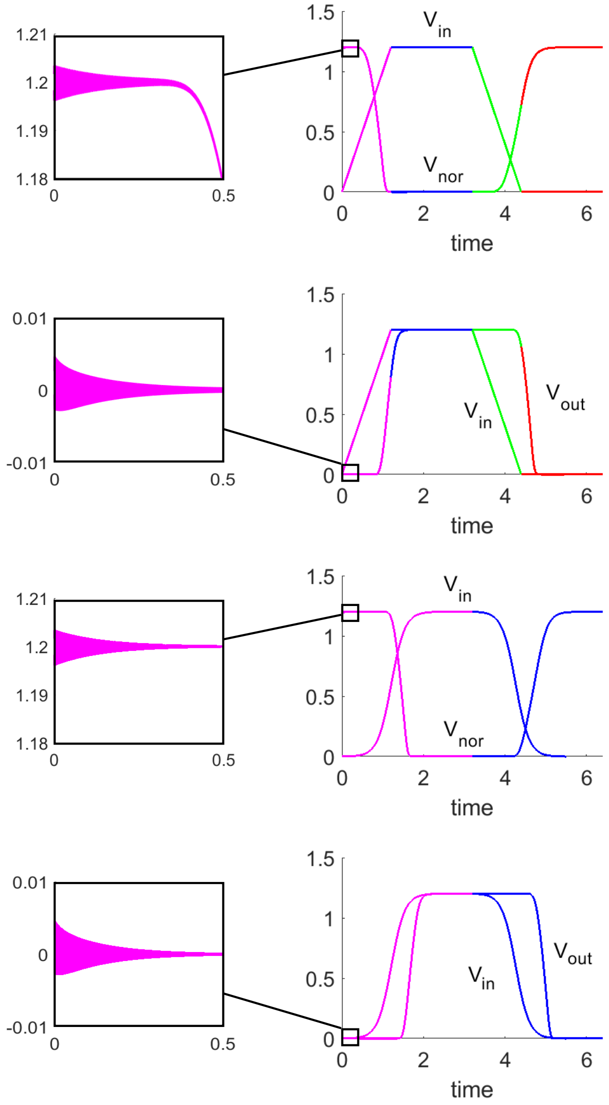

Fig. 16: OR gate V_{nor} and V_{out} over-approximation set for V_{in} = Ramp (top half) and V_{in} = Sig (bottom half)

Uniform model:

$$\dot{V}_1 = \frac{1}{C_1}(I_{1,I} - I_{2,I})$$

$$\dot{V}_2 = \frac{1}{C_2}(I_{1,II} - I_{2,II})$$

For an inverter loop we solely need to specify the initial values of voltages V_1 and V_2 as no input exists. Since this storage loop has two stable configurations $(V_1, V_2) = (0, V_{DD})$ and $(V_1, V_2) = (V_{DD}, 0)$ we expect to see a resolution to one of these configurations.

Verification results shown in Figure 17 meet our expectation. Due to the fact that initially $V_1 > V_2$ the stable configuration $(V_1, V_2) = (V_{DD}, 0)$ is rapidly approached and stored from there onwards.

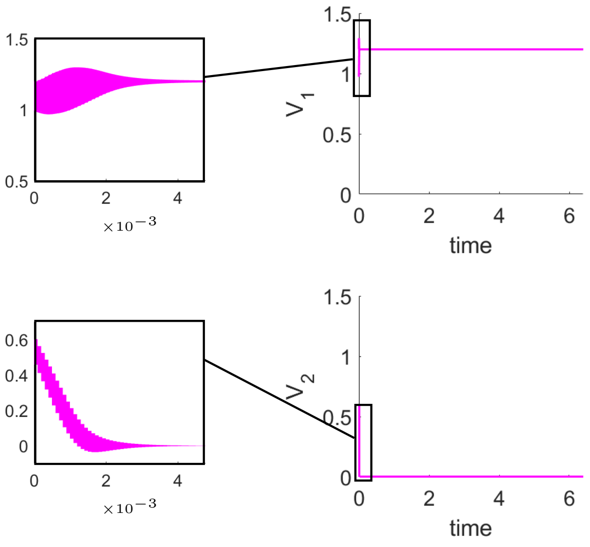

Fig. 17: Inverter loop output voltages over-approximation set for V_1 (top) and V_2 (bottom)

H. *feed back* OR *gate*

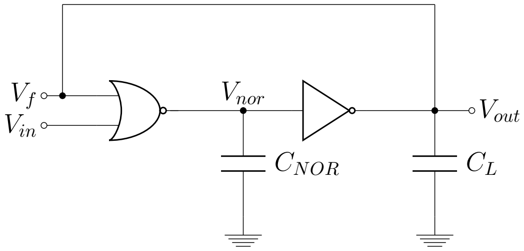

Fig. 18: Internal structure of OR gate loop

Uniform model:

The uniform model for this circuit is exactly the same as for the regular OR gate with the difference, that $I_{1,NOR}$ and $I_{3,NOR}$ now depend on V_{out}, not on an independent input. This feedback path is the main reason for the storing capabilities of this loop.

This circuit has been used to investigate metastability effect. The corresponding results have been shown in Section VI.