A Boolean Model for Delay Fault Testing of Emerging Digital Technologies based on Ambipolar Devices

Marcello Dalpasso
DEI - University of Padova, Italy
marcello.dalpasso@unipd.it

Davide Bertozzi
DI - University of Ferrara, Italy
Email: davide.bertozzi@unife.it

Michele Favalli
DI - University of Ferrara, Italy
Email: michele.favalli@unife.it

***Abstract*—Emerging nanotechnonologies such as ambipolar carbon nanotube field effect transistors (CNTFETs) and silicon nanowire FETs (SiNFETs) provide ambipolar devices allowing the design of more complex logic primitives than those found in today's typical CMOS libraries. When switching, such devices show a behavior not seen in simpler CMOS and FinFET cells, making unsuitable the existing delay fault testing approaches. We provide a Boolean model of switching ambipolar devices to support delay fault testing of logic cells based on such devices both in Boolean and Pseudo-Boolean satisfiability engines.**

1. Introduction

As conventional CMOS devices reaches physical limits, new nanoelectronic devices [1], [2] emerges, featuring more complex electrical properties than CMOS ones. For instance, the conduction state of several such devices can be controlled by two terminals instead of a single one, as in [3] or [4] featuring ambipolar behavior to control the dominant carriers.

The challenge in the design, verification and test of logic primitives based on such emerging nanotechnologies requires a proper EDA support. For instance, the in-field controllability of the device polarity enables to design more expressive technology libraries than conventional CMOS ones [6] and this is a pivotal asset to be harnessed via logic synthesis [7]. Besides design, verification and testing strategies are keys to enable the correct operation of any product, but such a complex device functionality, jointly with unreliable manufacturing processes, causes designers not to be familiar enough with the defects of these devices.

Today's CMOS and FinFET fault models cannot manage all defects of ambipolar devices, since they do not cover all their modes of operation [8], and even when dealing with assessed models (e.g., stuck-open or delay faults) it's difficult to select device modeling abstractions for test pattern generation. Although in logic synthesis behavioural models [7] provide a suitable abstraction for device functionality, in testing they end up losing key structural properties. In contrast, as regards test pattern generation it's not clear whether gate-level models can be extended in a straightforward way due to the higher electrical complexity of logic cells in emerging nanotechnologies.

Logic primitives based on ambipolar devices share a static complementary behavior with CMOS and FinFET ones. Conversely, differences arise when considering the dynamic behavior, mainly because ambipolar devices may be controlled by two different nets. Therefore, we focus here on the transient behaviour and on the associated causes for circuit malfunctioning. We claim that standard gate-level models cannot generate high-quality test patterns: our analysis shows that gate-level equivalent test patterns exhibit relevant differences in the faulty delay and ignoring such facts may result in test escapes. This urged the development of a switch-level model that keeps circuit features, thus proving capable of differentiating patterns that a gate-level model would consider as equivalent. A Boolean model is proposed to capture the switching behavior of logics in ambipolar nanodevices with a twofold novelty: a switch-level model accounts for ambipolar behaviour, while the dynamic circuit behaviour is accounted by a multi-frame qualitative time modeling that can trace the transients of complex logics.

2. Signal and transistor model

When two input vectors are applied to a combinational network, we describe the switching characteristics of logic gates with N time frames (the value of N will be discussed in the next sections). For each signal s, let $v(s,i)$, $i = 1..N$ be a Boolean variable denoting its value in the i-th frame: $v(s,1)$ and $v(s,N)$ are the steady-state value of s before the first and after the second input vector is applied, respectively. In a hazard-free context, the s value is:

$$v(s,i) = w(s,i)'v(s,1) + w(s,i)v(s,N) \tag{1}$$

where $w(s,i) \in \{0,1\}$ is 0 if in the i-th frame s has not switched yet, 1 if s switched. The following constraints hold (eq. 3 encodes hazard-free stable or transition cases):

$$w(s,1) = 0,\ w(s,N) = 1 \tag{2}$$

$$w(s,i) \rightarrow w(s,i+1),\ \ i = 2..N-1 \tag{3}$$

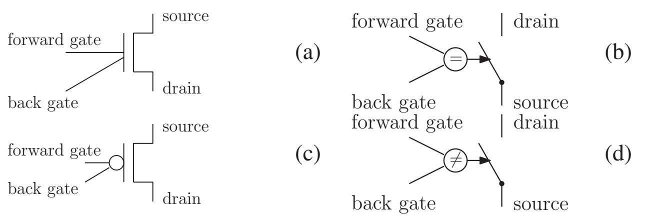

Figure 1: CNTFETs and their logic-level abstractions.

Fig. 1a represents an ambipolar CNTFET transistor [4], [5], having two gate terminals: the back gate b, whose value switches the CNTFET between an n-type FET ($v(b,i) = 1$) and a p-type FET ($v(b,i) = 0$), and the forward gate f, whose value determines the state (ON/OFF) of the CNTFET as a

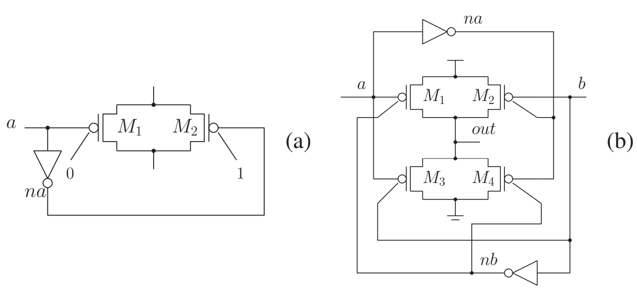

Figure 2: A CNTFET pass transistor (a) and an XNOR gate based on **xor**-type ambipolar devices (b).

function of the type set by the back gate. Fig. 1b shows the logic level abstraction [7] of such **xnor**-type device that is ON when $v(b,i) = v(f,i)$.

Fig. 1c shows the dual case, a CNTFET that behaves like an n-type FET if $v(b,i) = 0$ and as a p-type FET if $v(b,i) = 1$. Such **xor**-type device is ON if $v(b,i) \neq v(f,i)$.

The state of a CNTFET (M) can be described by two variables, $on_n(M,i)$ and $on_p(M,i)$, true when the device is ON as an n-type or a p-type device, respectively. They can be expressed as a function of the values of b and f:

$$\textbf{xnor}\text{-type}\begin{cases} on_n(M,i) &= v(b,i)v(f,i) \\ on_p(M,i) &= v(b,i)'v(f,i)' \end{cases} \tag{4}$$

$$\textbf{xor}\text{-type}\begin{cases} on_n(M,i) &= v(b,i)'v(f,i) \\ on_p(M,i) &= v(b,i)v(f,i)' \end{cases} \tag{5}$$

The ON state of the CNTFET is $on = on_n + on_p$.

The pass-transistor circuit (Fig. 2a) overcomes the limitations of a single CNTFET working as p/n-type FET to transmit a low/high logic value. When the pass-transistor is ON, both devices are ON, one of them working as a p-type device and the other as an n-type device, thus allowing for an efficient transmission of both logic values.

3. Logic primitive model

Ambipolar devices enable the design of compact gates using XOR/XNOR functions, enhancing the implementation of counting and arithmetic functions [6]. Static XOR/XNOR gates could be implemented with two CNTFET: pass-transistor features, however, suggest the design of Fig. 2b for an XNOR gate that always achieves full-swing output. In such a gate the inputs and the inverters' outputs are so described by our multiple time frame model, with $i = 1..N$:

$$\begin{aligned} on_n(M_1,i) &= v(a,i)v(nb,i)', on_p(M_1,i) = v(a,i)'v(nb,i) \\ on_n(M_2,i) &= v(na,i)v(b,i)', on_p(M_2,i) = v(na,i)'v(b,i) \\ on_n(M_3,i) &= v(a,i)v(b,i)', on_p(M_3,i) = v(a,i)'v(b,i) \\ on_n(M_4,i) &= v(na,i)v(nb,i)', on_p(M_4,i) = v(na,i)'v(nb,i) \end{aligned}$$

The gate output value depends on the output connectivity to the power supply and/or ground, that can be either evaluated by a prior analysis of the DC paths or implicitly included in the model, for instance using the switch-level SAT technique proposed in [9]. The first option is used here because it simplifies the model description when the pull-up/-down networks can be easily identified.

For a given network $x \in \{up, down\}$, $on_n(x)$ ($on_p(x)$) is true if at least one DC path of ON $n(p)$-type-only transistors connects the gate output to the x voltage source, while $on(x)$ is true if at least one path of ON transistors exists with no regard to transistor type, i.e., $on_n(x) \rightarrow on(x)$ as well as $on_p(x) \rightarrow on(x)$.

	signal				transistor			
frame	a	b	na	nb	M_1	M_2	M_3	M_4
1	1	0	0	1	$\emptyset$	$\emptyset$	n	p
2	1	1	0	1	$\emptyset$	p	$\emptyset$	p
3	0	1	0	1	p	p	p	p
4	0	1	1	1	p	$\emptyset$	p	$\emptyset$
5	0	1	1	0	$\emptyset$	$\emptyset$	p	n

TABLE 1: Signal waveforms and corresponding evolution of transistors' state in an XNOR gate during an event featuring all transistors ON in frame 3. The symbol n/p denotes an ON CNTFET in the n/p mode; $\emptyset$ denotes an OFF CNTFET.

Path analysis defines such variables. For Fig. 2b:

$$\begin{aligned} on_n(up,i) &= on_n(M_1,i) + on_n(M_2,i) \\ on_p(up,i) &= on_p(M_1,i) + on_p(M_2,i) \\ on_n(down,i) &= on_n(M_3,i) + on_n(M_4,i) \\ on_p(down,i) &= on_p(M_3,i) + on_p(M_4,i) \end{aligned}$$

and, losing some electrical-level information: $on(up,i) = on_n(up,i) + on_p(up,i)$ and $on(down,i) = on_n(down,i) + on_p(down,i)$. Finally, inverters' constraints are provided. As an example, the inverter driving na in Fig. 2b is modeled by Eq. 1 and $v(na,1) = v(a,1)'$, $v(na,N) = v(a,N)'$.

The proposed model consists of the logical conjunction of all signal, transistor and network equations, and can be translated in the form required by Boolean handling packages and solvers: 1) conjunctive normal form for Boolean and Pseudo-Boolean (PB) satisfiability solvers [10]; 2) Reduced Ordered Binary Decision Diagram (ROBDD); 3) Constraint Satisfaction Problem (CSP), here described by the Minizinc format [11] that, in perspective, can be used to add quantitative constraints requiring the use of integer variables.

Example. As regards the circuit of Fig. 2b with hazard-free transitions at inputs, cases could be evaluated by hand, so it's a good choice for illustrating the model. To account for all possible event orderings, $N = 5$ is enough; then, we add constraints on variables $w(s,i)$ to prevent any simultaneous event on a, b, na and nb.

For instance, let's look at static power dissipation during transients: we are interested in finding whether an event ordering exists that makes all transistors temporarily ON during a transient, thus resulting in a worst case condition. In the Boolean model this additional constraint is given by:

$$\exists i \mid (on_n(M_1,i)+on_p(M_1,i))(on_n(M_2,i)+on_p(M_2,i)) \\ (on_n(M_3,i)+on_p(M_3,i))(on_n(M_4,i)+on_p(M_4,i)) = 1$$

One of the eight possible solutions provided by any Boolean solver engine is shown in Tab. 1.

In any solution the gate output initial and final value are the same, thus meaning that the imposed condition occurs only during a static hazard when the input transitions are close enough to make both inputs switch before the inverters. The actual waveform of the gate output depends on the duration of each time frame and, therefore, on signal skews and inverter delays: our method is not intended to replace circuit-level simulation but to provide a qualitative framework to drive such analysis. Conversely, the gate output is in the high impedance state (i.e., all transistors are OFF) during an output hazard in anyone of the eight different conditions.

The proposed model has been extended to approximate the pull-up/-down conductances of gates during transients, currently limited to series/parallel networks whose conductance is mapped by Boolean conditions on gate signals for all frames.

4. Delay fault testing in ambipolar logic cells

We only consider gross delays due to devices that feature impaired current driving because of defects affecting either their DC contacts or their control gate terminals, but the proposed approach can be (easily) extended to different cases.

Before analyzing the detection of such faults in logic primitives based on ambipolar devices, let us first briefly recall some concept regarding their detection in conventional CMOS gates. The fault is detected when a test pair is applied that propagates a transition from the pseudo-primary inputs (PPIs) to the pseudo-primary outputs (PPOs) of a combinational module through the faulty gate. If the transition excites the fault, an additional delay will affect some PPOs, maybe resulting in the sampling of a logic error; the fault is excited when the faulty device is critical in providing the current to switch the gate output. If all the current flows through the faulty device, the faulty gate output will reach its final value with an additional delay that depends on current capabilities of the faulty device: if the current is indefinitely lowered, the gate output will never switch.

Several testing approaches have been proposed for such defects: transition delay fault testing [12] supposes that the additional gate delay is always large enough to result in the sampling of a logic error at the PPOs. Conversely, gate delay fault testing accounts for the defect size and the delays of the paths propagating the transition [13]. In standard-cell based technologies, these effects are typically modeled at the gate level. Switch-level fault simulation for delay faults [14] and switch-level delay test robustness conditions [15] have been proposed for more complex full-custom circuits.

In the case of logic primitives based on ambipolar devices these considerations partially hold: the pass-transistor configuration of devices is intrinsically redundant and the logic delay of the gate cannot indefinitely grow when lowering the faulty driving capabilities. This focuses the attention on gate delay fault testing because the defect size is expected to be upper bounded.

In the practice, a single faulty device cannot change the ON state of the (pull-up/pull-down) network including it and the complex structure of such primitives makes the use of a gate-level model questionable. In the next section, we will use the proposed model to investigate the technology-specific issues regarding the detection of gross delay defects. At first we will characterize the fault-free switching modes of a complex cell. Then, we will introduce fault detection specific constraints and we will show test generation results.

For a given cell and fault, our model is used to generate a local test pattern exciting the fault. Then, a global test that justifies this pattern at the PPIs of the combinational module containing the faulty cell and propagates the output transition to its PPOs can be computed using a SAT-based delay fault test generation framework.

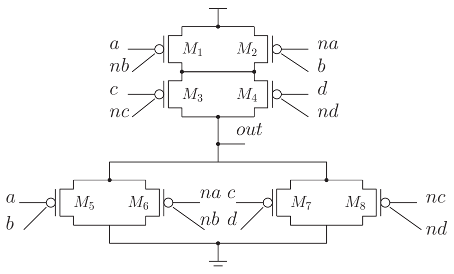

Figure 3: Example of ambipolar CNTFET complex logic primitive implementing the function $out = (a \oplus b)'(c \oplus d)'$ with a (not shown) single inverter for each input.

i	n_{sw}	$\#cfg$	n_{cc}	n_{up}	(g_{up}, g_{down})				g
1	1	8	1	3	0,1.8	1.0,0.8	1.8,0		2.0
2	1	8	1	3	0,1.8	1.2,1.0	1.8,0		2.0
3	2	24	1	2	0,1.8	0,1.8	1.0,0.8	1.8,0	0.2
4	2	24	1	2	0,1.8	0,1.8	1.2,1.0	1.8,0	0.2
5	2	8	2	3	0,3.6	1.0,2.0	1.2,1.0	1.8,0	1.0
6	2	16	2	3	0,3.6	0.8,1.6	1.0,0.8	1.8,0	1.2
7	2	9	2	3	0,3.6	0.8,1.8	1.2,1.0	1.8,0	1.0

TABLE 2: For the gate in Fig. 3, here are different model solutions. Left-to-right: index (i) of a group of transitions showing a similar behavior over time frames; # of switching inputs (n_{sw}); # of solutions featuring a similar behavior while differing in either input values or signal transition order ($\#cfg$); # of frames featuring short circuit current (n_{cc}); # of frames with ON pull-up (n_{up}). Rightmost columns commented in the text.

4.1. Characterization of fault-free switching

Referring to Fig. 3, we explore the gate behavior with single and multiple (simultaneous) hazard-free input transitions switching the output from 0 to 1. Tab. 2 reports all the solutions of our model provided by a solver under such constraints. For space reasons the table shows only single and double input transitions with $N = 4$ to accomodate for any events ordering. The table shows information about the state of the gate's switching networks as well as some approximate result on their conductance, achieved by a CSP formulation and a finite domain solver that directly accounts for conductance value. Such qualitative data show that any rising transition presents static current dissipation due to inverters' delays and the behavior is so complex that can hardly be fully accounted by a gate delay model.

A deeper insight in switching networks is needed to qualitatively reason about delays, as summarized in the wide column 6 that, for each time frame, shows the approximated pull-up (g_{up}) and pull-down (g_{down}) conductance evaluated with symmetric worst-case delay for rising and falling transitions and 20%-reduced conductance for CNTFETs acting as $n(p)$-type devices when propagating high (low) logic values. The number of frames grows with the number of switching inputs to account for more signal transitions.

The rightmost column reports the difference (g) between the sum of pull-up and pull-down conductances over all time frames but the first one: it is simply an indicator of the current driving efficiency of the gate during the transient and does not directly represent delays.

The large number of possible different behaviors and values of current driving capabilities clearly indicates the need for specialized EDA tools accounting for the new features possibly introduced by this kind of technologies.

i	cell	$abcd$	n_{cc}	n_{up}	(g_{up}, g_{down})			g
1	fault-free	0f11	1	2	0,1.8	1.0,0.8	1.8,0	2.0
	faulty		1	2	0,1.8	1.0,0.8	1.0,0	1.2
2	fault-free	f011	1	2	0,1.8	1.2,1.0	1.8,0	2.0
	faulty		0	2	0,1.8	0,1.0	1.0,0	0.0
3	fault-free	00r1	1	2	0,1.8	1.2,1.0	1.8,0	2.0
	faulty		0	2	0,1.8	0,1.0	1.0,0	0.0
4	fault-free	001r	1	2	0,1.8	1.2,1.0	1.8,0	2.0
	faulty		1	2	0,1.8	0.8,1.0	1.0,0	0.8
5	fault-free	f000	1	2	0,1.8	1.2,1.0	1.8,0	2.0
	faulty		0	2	0,1.8	0,1.0	1.0,0	0.0
6	fault-free	0f00	1	2	0,1.8	1.0,1.0	1.8,0	1.8
	faulty		1	2	0,1.8	1.0,1.0	1.0,0	1.0
7	fault-free	00f0	1	2	0,1.8	1.2,1.0	1.8,0	1.8
	faulty		1	2	0,1.8	0.8,1.0	1.0,0	0.8
8	fault-free	000f	1	2	0,1.8	1.0,0.8	1.8,0	2.0
	faulty		1	2	0,1.8	0.8,0.8	1.0,0	1.0

TABLE 3: Test pairs produced by the proposed approach for a fault affecting transistor M_1 in the gate of Fig. 3. Left-to-right: an index (i); gate input values (r and f for rising and falling transitions, respectively); # of frames with both pull-up/-down ON (n_{cc}); # of frames with ON pull-up (n_{up}); initial conductances of the pull-up/-down; achieved conductances when the switching inverter has not switched yet; final conductances; g defined in Sect. 4.1.

4.2. Fault detection results

Consider the circuit of Fig. 3 with a not-fully-conducting transistor M_1. To generate a test pair exciting such a fault, we use a symbolic testbench that sets a difference between the states of the faulty and fault-free circuits. The fault is injected by forcing M_1 OFF in all frames and additional constraints defining the symbolic comparator simply require that $on(up^{ff}, N) = 1$ and $on(up^{f}, N) = 0$, where ff and f denote the fault-free and the faulty circuit, respectively.

In a typical CMOS gate and ignoring robustness issues, such a comparator would generate a test pair, while in this gate it would lead to an unsatisfiable instance. The reason is self-evident: turning ON the fault-free (M_1') counterpart of the faulty device, M_2 is turned ON, too, and the pull-up of the faulty cell is ON in spite of the fault. During the transient, M_2 can still drive a non negligible current.

We can exploit the information regarding $on_n(up, i)$ and $on_p(up, i)$. The single devices, in fact, are not redundant with respect to the connectivity of the output to the power supplies when only transistors in the n or p mode are considered. In particular, we can formulate a new constraint as:

$$on(up^{ff},N)on(up^{f},N)' + on_p(up^{ff},N)on_p(up^{f},N)' = 1 \quad (7)$$

If the solver is unable to satisfy the first logic product in Eq. 7, it will satisfy the second one, thus resulting in an ON pull-up whose currrent paths, however, always include at least a device in the n-mode that does not efficiently propagate the final high logic value. When optimization is possible (PB and CSP cases), Eq. 7 can be reformulated as a maximization target function given by $2 \cdot on(up^{ff}, N)on(up^{f}, N)' + on_p(up^{ff}, N)on_p(up^{f}, N)'$: the solver provides the better solution even when the faulty device is not redundant since it is used in the n or p mode only.

We built the test generation model with the detection constraint of Eq. 7 and an additional constraint forcing a single input transition ($n_{sw} = 1$). The solutions provided by the solver are illustrated in Tab. 3 where we accounted for approximated conductances, too.

It is worth mentioning that, while the behavior of the fault-free circuit is rather independent of the applied input vector ($g \in \{1.8, 2.0\}$), the faulty behavior spans over a wider range ($g \in \{0.0, 1.2\}$). Test vectors featuring faulty low values of g are likely to produce a larger transition delay. Therefore, these tests are preferable to others, thus showing the utility of using Pseudo-Boolean or CSP solvers that allow for optimization.

5. Conclusions

A switch-level multi-frame model for logic primitives based on ambipolar devices available in emerging technologies has been proposed and instantiated for ambipolar CNTFET circuits. The model can be used by different Boolean engines to analyze the behavior of these circuits. Here, it has been specifically applied to the testing of gross delay faults due to transistor defects. Results show that the proposed approach accounts for complex behaviors of these circuits that cannot be managed by existing gate-level approaches.

References

[1] Y.-B. Kim, “Challenges for Nanoscale MOSFETs and Emerging Nanoelectronics,” in *Trans. on Electrical and Electronic Materials*, vol. 11, no. 3, pp. 93–105, 2010.

[2] D. Nikonov and I. Young, “Benchmarking of Beyond-CMOS Exploratory Devices for Logic Integrated Circuits,” *IEEE J. on Exploratory Solid-State Comput. Dev. and Circ.*, vol. 1, pp. 3–11, 2015.

[3] M. De Marchi, “Polarity control in double-gate, gate-all-around vertically stacked silicon nanowire FETs,” IEDM 2012.

[4] Y.M. Lin, “High-performance carbon nanotube field-effect transistor with tunable polarities,” *IEEE Trans. Nanotechnologies*, vol. 4, no.5, pp. 481–489, 2005.

[5] J. Deng et al., “Carbon Nanotube Transistor Circuits: Circuit-Level Performance Benchmarking and Design Options for Living with Imperfections,” ISSCC 2007.

[6] M. H. Ben-Jamaa et al., “An Efficient Gate Library for Ambipolar CNTFET Logic,” *IEEE Trans. on Computer-Aided Design of Integ. Circ. and Sys.*, vol. 30, no. 2, pp. 242–255, 2011.

[7] L. Amaru et al., “New Logic Synthesis as Nanotechnology Enabler,” *Proceedings of the IEEE*, vol. 103, no. 11, pp. 2168–2195, 2015.

[8] H. G. Mohammadi et al., “Fault modeling in controllable polarity silicon nanowire circuits,” DATE 2015.

[9] M. Favalli and M. Dalpasso, “Applications of Boolean Satisfiability to Verification and Testing of Switch-Level Circuits,” *J. of Electronic Testing*, vol. 30, no. 1, pp. 41–55, 2014.

[10] N. Een and N. Sorensson, “Translating Pseudo-Boolean Constraints into SAT,” *J. on Satisfiability, Boolean Modeling and Comput.*, vol. 2, no. 14, pp. 1-26, 2006.

[11] N. Nethercote et al., “Minizinc: Towards a standard CP modelling language,” CP 2007.

[12] J. A. Waicukauski et al., “Transition Fault Simulation,” *IEEE Design & Test of Computers*, vol. 4, no. 2, pp. 32–38, 1987.

[13] V. S. Iyengar et al., “On computing the sizes of detected delay faults,” *IEEE Trans. on Computer-Aided Design of Integ. Circ. and Sys.*, vol. 9, no. 3, pp. 299–312, 1990.

[14] S. Bose et al., “Algorithms for switch level delay fault simulation,” ITC 1997.

[15] S. Natarajan et al., “Switch-level delay test,” ITC 1999.