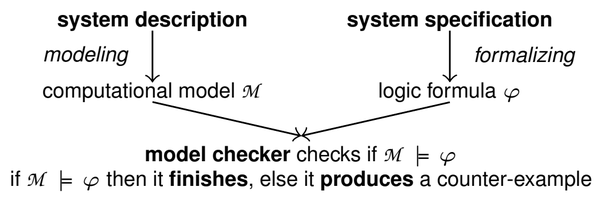
Using Model Checker to Analyze and Test Digital Circuits with Regard to Delay Faults

BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Josef Strnadel (www.fit.vut.cz/∼strnadel, strnadel@fit.vut.cz)

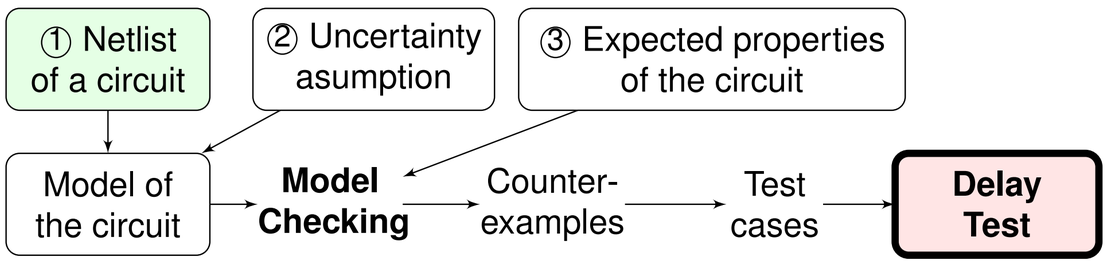
What is this paper about?

- It presents a **model checking approach** aiming to facilitate the solutions of problems with regard to analyzing consequences and testing of delay faults.



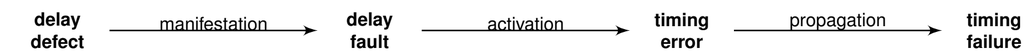
- It expects that a circuit is modeled as a network of **stochastic hybrid timed automata** (SHA) capable to describe the circuit both in the logical and **temporal domains**, including facts such as **uncertainty** and **variations**.
- It expects that one **gathers attributes** and **formalizes expected properties** of a circuit and transform the circuit into the **proposed model**.
- It builds on a **statistical model checker** (SMC) used to check the properties and to produce a **counter-example** for each property being violated.
- It shows that the **counter-examples** can be transformed into **test cases** and finally, into a **delay test** able to check whether the timing requirements are met.

Top-level view at our framework

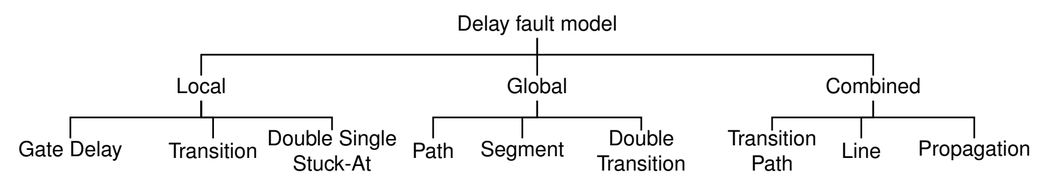


Delay faults

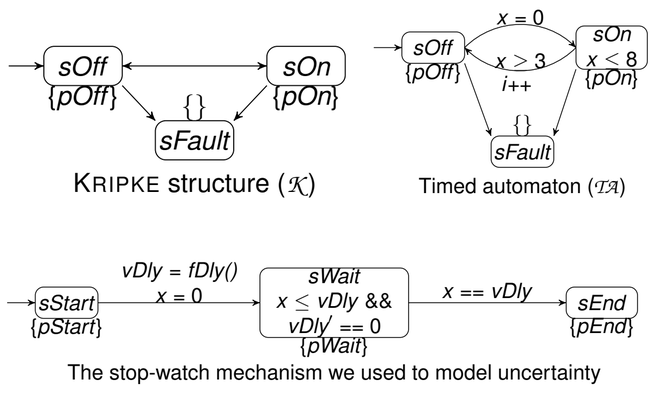
- Narrow timing margins in modern digital circuits result in **delay defects**.

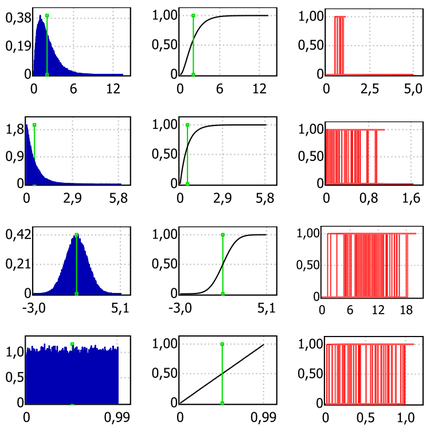


- The probability that such a defect occurs increases with factors such as **shrinking feature sizes**, **increasing process variations** or **operating frequencies**, **aging/stress**.
- Traditionally, **timing** is considered in connection with the logic design, physical design and layout, and delay testing phases of the circuit development process and builds on principles of **delay characterization**, **fault models** and **timing analysis**.

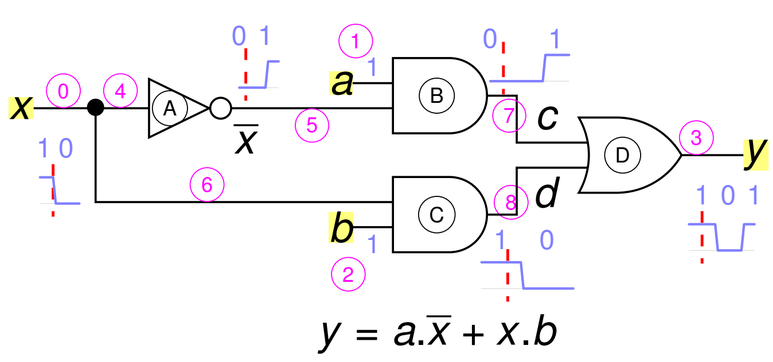


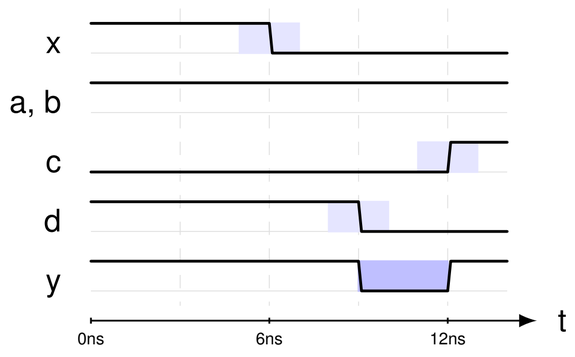
Timed automata





Example circuit: schema, timing of signals, representation





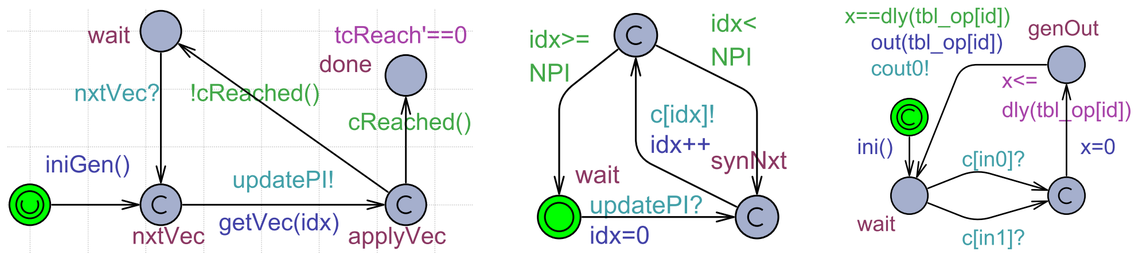
Listing 1. An illustration to 3 PI + 1 PO setup of our framework

```
1 const int NPI = 3;                    // # PI bits
2 const int NPO = 1;                    // # PO bits
3 const int PI[NPI] = {0,1,2};          // PIs
4 const int PO[NPO] = {3};              // POs
```

Listing 2. An illustration to instantiation/linking of components

```
1 w04(0,PI[0],4,c[PI[0]],c[4]);
2 w06(1,PI[0],6,c[PI[0]],c[6]);
3 gA=gNot(2,4,5,c[4],c[5]);
4 gB=gAnd(3,PI[1],5,6,c[PI[1]],c[5],c[7]);
5 gC=gAnd(4,PI[2],6,8,c[PI[2]],c[6],c[8]);
6 gD=gOr(5,7,8,PO[0],c[7],c[8],c[PO[0]]);
```

Delay faults: Modeling and analysis



vector generator signal propagator logic

- $\Pr[t \leq tbound](|val_{act}(x,t) - val_{exp}(x,t)| > \Delta x)$:
probability that the value at a node/line x deviates from the expected one by more than Δx within the specified time limit (*tbound*)
- $applicationOf(v) \rightsquigarrow_{\leq t} val_{act}(x) \neq val_{exp}(x)$:
ability of an input vector v (or, a pair of vectors) to check whether the value at node/line x deviates from the expected one for no more than t units of time

Algorithm 1: ΔTGEN generates a delay test

Input: The stochastic model $\mathcal{M}$ of a circuit, set $\mathcal{S}$ of expected properties (φ) of the circuit (see ①–③ in Fig. 5), probability uncertainty ε, desired fault coverage fc

Output: A test for checking timing of $\mathcal{M}$ given by $\mathcal{S}$

1 $counter\_examples_{\mathcal{S}} \leftarrow test\_cases_{\mathcal{S}} \leftarrow test_{\mathcal{S}} \leftarrow \{\}$
2 **foreach** $\varphi \in \mathcal{S}$ **do**
3 $\quad stats_{\varphi} \leftarrow getStats(SMC(\varepsilon, \mathcal{M}, \varphi))$
4 $\quad counter\_examples_{\varphi} \leftarrow getCntrEx(\varepsilon, \mathcal{M}, \varphi)$
5 **end**
6 **foreach** $\varphi \in \mathcal{S}$ **do**
7 $\quad test\_cases_{\mathcal{S}} \leftarrow getTCases(counter\_examples_{\varphi})$
8 **end**
9 $test\_vectors_{\mathcal{S}} \leftarrow getInputs(test\_cases_{\mathcal{S}})$
10 $test\_responses_{\mathcal{S}} \leftarrow getOutputs(test\_cases_{\mathcal{S}})$
11 $test_{\mathcal{S}} \leftarrow optimize(test\_vectors_{\mathcal{S}}, fc, stats_{\varphi})$
12 **return** $test_{\mathcal{S}}$

References

[1] C. Baier and J.-P. Katoen, *Principles of Model Checking*, ser. Representation and Mind. MIT Press, 2008.

[2] E. M. Clarke, T. A. Henzinger, H. Veith, and R. Bloem, *Handbook of Model Checking*, 1st ed. Cham: Springer International Publishing, 2018, DOI 10.1007/978-3-319-10575-8.

[3] J. Mahmod, S. Millican, U. Guin, and V. Agrawal, "Special Session: Delay Fault Testing - Present and Future," in *2019 IEEE 37th VLSI Test Symposium (VTS)*, 2019, pp. 1–10, DOI 10.1109/VTS.2019.8758662.

[4] A. David, K. Larsen, A. Legay, M. Mikučionis, and D. Poulsen, "Uppaal SMC Tutorial," *International Journal on Software Tools for Technology Transfer*, vol. 17, no. 4, pp. 397–415, 2015, DOI 10.1007/s10009-014-0361-y.

[5] G. Agha and K. Palmskog, "A Survey of Statistical Model Checking," *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, vol. 28, no. 1, pp. 6:1–6:39, Jan. 2018, DOI 10.1145/3158668.

strnadel@fit.vut.cz
www.fit.vut.cz/∼strnadel

DDECS 2021
DESIGN & DIAGNOSTICS of ELECTRONIC CIRCUITS & SYSTEMS
Vienna, Austria April 7-9

This work was supported by the **Czech Science Foundation (GACR) project No. 19-10137S** "Designing and Exploiting Libraries of Approximate Circuits". D:20210330173114+02'00'