Electrical Rule Checking of Integrated Circuits using Satisfiability Modulo Theory

B. Ferres*, O. Oulkaid*§†, L. Henrio*, M. Khosravian G.†, M. Moy*, G. Radanne*, P. Raymond§
*Univ Lyon, EnsL, UCBL, CNRS, Inria, LIP, F-69342, LYON Cedex 07, France.
§Univ. Grenoble Alpes, CNRS, Grenoble INP, VERIMAG, 38000 Grenoble, France
†Aniah, 38000 Grenoble, France
{bruno.ferres, matthieu.moy, ludovic.henrio, gabriel.radanne}@ens-lyon.fr
pascal.raymond@univ-grenoble-alpes.fr, {m.khosravian, o.oulkaid}@aniah.fr

***Abstract*—We consider the verification of electrical properties of circuits to identify potential violations of electrical design rules, also called Electrical Rule Checking (ERC). We present a general approach based on Satisfiability Modulo Theory (SMT) to verify that these errors cannot occur in a given circuit. We claim that our approach is scalable and more precise than existing analyses, like voltage propagation. We applied these techniques to a specific type of errors, the missing level shifters. On an industrial case-study, our technique is able to flag 31% of the warnings raised by the voltage propagation analysis as being false alarms.**

***Index Terms*—Electrical rule checking, Integrated Circuits, SMT solving**

I. Electrical Rule Checking

During hardware design processes, verification of the digital designs is a particularly important task since, unlike software, updates cannot be deployed after manufacturing, meaning that any bug left in the system can induce heavy additional costs. Simulation is a widely used method to verify a hardware design, but it can only prove the presence of bugs, not their absence, and highly depends on the test vectors that are defined by the developers. Formal methods like model-checking can, on the other hand, prove the correctness of a circuit, or of any sub-circuit considered. While theoretically limited by algorithmic complexity or even undecidability, formal methods have successfully been applied in many contexts in practice[1].

In a typical hardware design-flow, verification can happen at multiple stages. Algorithmic properties, such as temporal or logical behavior, can be checked at the highest level of abstraction (RTL, or even on C code in the case of high-level synthesis). However, some properties can only be considered in the final steps of the design flow, where fewer abstractions are used to describe circuits. For example, when a circuit contains multiple power-domains operating at different voltages, design rules state that a specific circuit — a level-shifter — must be used at the interface between power-domains. Level-shifters are not described at RTL level, since power-supplies are not modelled at all at this level of abstraction. They are inserted later in the design-flow, typically using tools based on the Universal Power Format (UPF). It is a mostly automatic step, but uses user-provided configuration files and possibly user-provided sub-circuits. Therefore, this step may introduce bugs in the design. It is important to check the presence of all required level-shifters after this step, hence after the synthesis stage of the flow. More generally, a complete and modern circuit usually contains hand-tuned parts, and it is crucial to check that these parts do comply with the design rules. Such verification, which is usually referred to as Electrical Rule Checking (ERC) [5], typically operates on transistor netlists. Such netlists are either handwritten or can be extracted from the layout of the circuit, and are required for another important verification step, called Layout Versus Schematic analysis. Consequently, ERC approaches generally operate on a transistor netlist, *i.e.* a description of the circuit using transistors (or other hardware components) connected by *nets* (*i.e.* wires).

To verify properties related to power supplies, a typical first step is called *voltage propagation* [3], [5], [6]. It computes, for each net of the circuit, which power-supply is potentially connected to this net, by using a naive approximation of the transistors' behavior (*i.e.* considering that the source and the drain of a transistor are connected unconditionally). When the voltage propagation finds a transistor with a gate connected to a supply S_g and a source connected to a supply S_s with neither S_g nor S_s being the ground, and the voltage of S_g lower than the one of S_s, that transistor is identified as being at the interface between power-domains, and must be protected with a level-shifter. The presence of a level-shifter can be asserted using pattern-matching, checking for the presence of a sub-circuit known to behave as a level-shifter [5], or this transistor can be tagged as potentially problematic and included in some coverage criteria for simulation-based verification. Voltage propagation is relatively simple and identifies all potential problems for several properties, but it is also very imprecise and yields a lot of false alarms.

To perform a more precise analysis, one needs to take into account the fact that the source and drain of a transistor are connected or not, depending on the voltage applied on the gate. In some cases, the gate voltage is known, and this condition can be propagated statically [7]. It is also possible to model the semantics of the transistor with logical formula to verify properties valid for any electrical configuration of the circuit. This was successfully applied to verify the absence of short-circuits [1], but the approach is limited to single-supply circuits, and to the identification of short-circuit conditions.

[1]See *e.g.* the FM conference series: https://fmeurope.org/symposia/

978-3-9819263-7-8/DATE23/© 2023 EDAA

We present a verification approach for transistor netlists, based on the satisfiability of logical formula. This approach is more precise than voltage propagation and supports multiple power supplies, hence allowing the verification of many properties like missing level-shifters, short-circuit or floating gate nets. The analysis considers various possible states for the nets of a circuit, such as short-circuits (at least two supplies connected together) and floating nets (connected to no supply).

II. Our Verification Approach: ERCtool

Our approach consists in compiling a transistor netlist into a logical formula F, and verifying the satisfiability of the conjunction of the formula with the negation of the error property with a Satisfiability Modulo Theory (SMT) solver, an extension of SAT able to deal with mixed numerical/Boolean formulas, as shown in Figure 1. We currently use Z3 [2] as a solver.

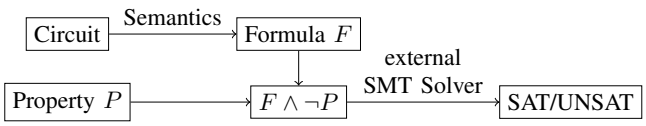

Fig. 1. Overall flow of ERCtool

Formula F describes the possible electrical steady-states of the circuit; it is the semantics of the circuit. When $F \wedge \neg P$ is *UNSAT*isfiable, it means that no electrical state of the circuit can lead to an error, and the property is proved. When it is *SAT*isfiable, the model provided by Z3 gives a concrete configuration where the error occurs, which is helpful for debugging. To ensure the absence of missing level-shifters we must prove the property P stating that "there exists no transistor in the circuit such that the gate and source have different non-zero voltages".

As-is, this approach cannot scale to a complete circuit, but we successfully analyzed sub-circuits of a real-life circuit, which is sufficient in practice even for large circuits. We used the SMT-based approach as a refinement after a voltage propagation pre-analysis that yielded false alarms. For each warning raised by voltage propagation, we extracted the part of the circuit containing the guilty transistor, and ran the SMT-based analysis to partition the warnings into false alarms and potential errors. Potential errors may still be false alarms since we do not consider the context where this circuit is instantiated yet.

We implemented a first way to build formula F, that we called "oriented semantics". For each transistor, a clause of F constrains the electrical status of the drain based on the status of the gate and the source. For each net, we merge the electrical status of drains it is connected to: if all drains force the same voltage value, this value is taken, if multiple values are present, it is a short circuit, and if no drain forces a value, then the net is floating. With this first approach, we successfully analyzed an industrial circuit with 10 distinct power supplies ranging from 0 to 2.5 volts, where 978 warnings were raised after voltage propagation. Among them, 600 could not be analyzed because of limitations of the "oriented semantics" approach, and 3 cases led to timeout. Out of the 378 remaining cases, 308 (*i.e.* 81 % of the analyzed cases, and 31 % of the total warnings) were tagged as false alarms — as many cases that one does not need to manually check anymore.

III. Ongoing Work

We started implementing a second way to build F, called "local semantics", to overcome the limitations of "oriented semantics". Instead of propagating the information one-way only, we now can also propagate the information from drain to source. For each transistor, a clause specifies whether the transistor is passing or not. For each net, a clause enforces that it is either in short-circuit state or in non-short circuit state. In the non-short-circuit case, the net's voltage must be equal to all the nets that it is connected to through a passing transistor. In the short-circuit case, the net must be connected to at least one net with strictly greater voltage and another with strictly lower voltage. This new semantics supports all topologies and should be able to analyze the cases that were not supported by "oriented semantics". The full case study is still work in progress, but we were able to correctly analyze small size circuits successfully.

We plan to work on performance optimization, considering the "local semantics" as a reference. We will propose other semantics that should let the solver perform better, and experiment with different tools like a plain SAT-solver with a pure boolean encoding, or Binary Decision Diagrams. Another research direction is modular verification [4]: the analyzer should be extended to be able to work on a large circuit by replacing some sub-circuits with a contract (assume-guarantee), which could be the key to enable the scaling of our approach.

On overall, we believe that the application of formal methods to ERC is an under exploited area with a great potential to improve the trust designers have in the circuits they design.

Acknowledgement: This work is partially funded by region Auvergne-Rhône-Alpes as part of the EASYTECH program led by MINALOGIC.

References

[1] Joao Afonso and Jose Monteiro. Analysis of short-circuit conditions in logic circuits. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pages 824–829. IEEE, March 2017.

[2] Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient smt solver. In C. R. Ramakrishnan and Jakob Rehof, editors, *TACAS*, pages 337–340, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[3] Matthew Hogan, Sridhar Srinivasan, Dina Medhat, Ziyang Lu, and Mark Hofmann. Using static voltage analysis and voltage-aware DRC to identify EOS and oxide breakdown reliability issues. *2013 35th Electrical Overstress/Electrostatic Discharge Symposium*, page 6, 2013.

[4] Orna Kupferman and Moshe Y. Vardi. Modular model checking. In Willem-Paul de Roever, Hans Langmaack, and Amir Pnueli, editors, *Compositionality: The Significant Difference*, pages 381–401, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[5] Jérôme Lescot, Vincent Bligny, Dina Medhat, Didier Chollat-Namy, Ziyang Lu, Sophie Billy, and Mark Hofmann. Static low power verification at transistor level for SoC design. In *ISLPED'12*, page 129, Redondo Beach, California, USA, 2012. ACM Press.

[6] Ziyang Lu and David Averill Bell. Hierarchical verification of chip-level ESD design rules. page 6, 2010.

[7] Maximilian Neuner and Helmut Graeb. Verification and revision of the power-down mode for hierarchical analog circuits. *Integration*, 73:1–9, July 2020.