Modular Analysis for Formal Verification of Integrated Circuits at Transistor Level

2023–2024

Keywords: SAT Solving; SMT Solving; Logical Formulas; Integrated Circuits; Optimization; `OCaml`

Context

Aniah is a start-up that offers tools for analyzing integrated circuits at an industrial scale[1]. Aniah has introduced algorithms that significantly push the boundaries of the size of analyzable circuits, from a few hundred thousand elements to several trillion. Aniah is working in collaboration with the Laboratoire de l'Informatique du Parallélisme (LIP) and the Verimag laboratory to consolidate and generalize its approach by supplementing its practical results with a theoretical backbone. We already carried-out one post-doc and started one Ph.D on the topic. One of the objectives of this study is to explore the applicability of state-of-the-art model-checking techniques to the problem of circuit electric verification.

Model-checking [14] consists in exploring all the reachable states of a system, typically to check the unreachability of a set of error states. It is a well-established set of techniques, and has successfully been applied both to software [1, 8, 7] and hardware [2, 4]. It is usually applied to check properties on the *behavior* of a system. For example, hardware model-checking usually considers boolean values (0 and 1, possibly extended with X and Z to model short-circuits and disconnected signals), but abstracts away the physical details (typically, voltage values are not modeled). Model-checking can be either enumerative (reachable states are explored one by one), or symbolic. Symbolic model-checking consists in representing a possibly very large set of states using a symbolic formula, that can be exponentially more efficient in terms of memory footprint and execution time. Common tools for symbolic model-checking are Binary Decision Diagrams (BDD) [5] and SAT-solvers [3] that allow manipulating boolean logical formulas. Satisfiability Modulo Theory (SMT) solvers extend SAT-solvers with non-boolean variables (e.g. rational numbers, integers, or other data structures). Among other work, these approaches have successfully been applied by the supervisors of this internship for Lustre program verification [11] and SystemC program verification [10].

Aniah proposed a graph-based algorithm to detect electrical errors in a hierarchical design circuit. In this regard, the algorithm first assigns a finite set of values to the input variables of the circuit. Then, by analyzing the behavior of each net within the circuit, the algorithm detects electrical errors. One of the main issues in this analysis is the complexity, both in space and time, that is exponential with respect to the size of input variables. While the existing algorithm is usually fast enough in practice thanks to the good properties of the circuit topology, we are working on using symbolic model checking tools (BDD, SAT- and SMT-solvers) to speed up verification even more, as has been done in previous works [13, 12]. We currently have a prototype tool [6] that compiles a circuit description into a logical formula comprising both numerical variables (representing voltage values) and booleans, that we solve using the Z3 [9] SMT solver.

However, while this approach can be used to analyze circuits with up-to thousands of transistors, it is not able to scale up to billions — the size of industrial circuits that Aniah can analyze. We hence consider improving our prototype with modular analysis, making it possible to analyze circuits in a hierarchical manner, from (simple) sub circuits up to the top of the hierarchy — in an efficient way.

[1] `https://www.aniah.fr/`

Objectives of the internship

The objective of the internship is to implement several heuristics that can be used for modular analysis of circuits — *i.e.* abstracting analysis results on sub-circuits to reason about the whole hierarchy. It is then expected that the candidate will:

- develop a theoretical basis to reason about sub-circuit in the existing SMT-based semantics
- implement the derived abstraction of sub-circuit in the `OCaml` prototype of the project, to enable modular analysis
- benchmark the performances of the approach
- propose various optimization to improve the abstraction

Context of the Collaboration and Physical Location

The internship is proposed as part of the collaboration between LIP laboratory (Lyon), Verimag laboratory (Grenoble), and Aniah company (Grenoble). A CIFRE Ph.D (Oussama Oulkaid) student is already working on a related subject, along with one of the supervisor (Bruno Ferres), who developed the original prototype during his post-doc. The student recruited for this internship will interact closely with them. A continuation on a Ph.D on a related subject is possible if the student is motivated.

The internship is proposed by LIP and Verimag, in collaboration with Aniah. The physical location of the internship is to be discussed with applicants. The student will visit other sites and meetings with all co-supervisors will be organised frequently.

- Laboratoire de l'Informatique du Parallélisme (LIP) – École Normale Supérieure de Lyon.
- Laboratoire Verimag, Grenoble.
- Aniah, Grenoble.

Required profile

The candidate should be familiar with algorithm design, understand the basics of Boole's algebra and logic. Good programming skills are required for the experimental validation of the approach. Since the software prototype is implemented in OCaml, prior knowledge of OCaml is appreciated, but the student can learn OCaml's basics during the internship. While the application domain is electronics, no knowledge of electronics is required to perform this internship.

Continuation with a Ph.D

A Ph.D is possible on the same topic, in partnership with the Aniah company (a CIFRE Ph.D considered, and a more academic Ph.D is also possible).

How to apply

Send an email to matthieu.moy@univ-lyon1.fr, pascal.raymond@univ-grenoble-alpes.fr and bruno.ferres@univ-grenoble-alpes.fr with your CV, a short text describing your motivation, and any document that can support your application.

Advisors

- Matthieu Moy, Maître de Conférences UCBL/LIP, `https://matthieu-moy.fr/`
- Pascal Raymond, Chargé de Recherche CNRS/Verimag, `http://www-verimag.imag.fr/~raymond/`
- Bruno Ferres, Maître de Conférences UGA/Verimag, `https://ferres.me/`.

References

[1] Thomas Ball, Vladimir Levin, and Sriram K Rajamani. A Decade of Software Model Checking with SLAM. *Communications of the ACM*, 54(7):68–76, 2011.

[2] Ilan Beer, Shoham Ben-David, Cindy Eisner, and Avner Landver. RuleBase: An Industry-Oriented Formal Verification Tool. In *33rd Design Automation Conference Proceedings, 1996*, pages 655–660. IEEE, 1996.

[3] Armin Biere, Alessandro Cimatti, Edmund Clarke, and Yunshan Zhu. Symbolic Model Checking without BDDs. In *International conference on tools and algorithms for the construction and analysis of systems*, pages 193–207. Springer, 1999.

[4] Aaron R Bradley. SAT-based Model Checking without Unrolling. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*, pages 70–87. Springer, 2011.

[5] Jerry R Burch, Edmund M Clarke, Kenneth L McMillan, David L Dill, and Lain-Jinn Hwang. Symbolic Model Checking: 1020 States and Beyond. *Information and computation*, 98(2):142–170, 1992.

[6] Bruno Ferres, Oussama Oulkaid, Ludovic Henrio, Matthieu Moy, Gabriel Radanne, Pascal Raymond, and Mehdi Khosravian. Electrical Rule Checking of Integrated Circuits using Satisfiability Modulo Theory. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–2. IEEE, 2023.

[7] Patrice Godefroid. Software Model Checking: The VeriSoft Approach. *Formal Methods in System Design*, 26(2):77–101, 2005.

[8] Daniel Kroening and Michael Tautschnig. CBMC–C Bounded Model Checker. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 389–391. Springer, 2014.

[9] Leonardo de Moura and Nikolaj Bjørner. Z3: An Efficient SMT Solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer, 2008.

[10] Matthieu Moy, Florence Maraninchi, and Laurent Maillet-Contoz. LusSy: an Open Tool for the Analysis of Systems-on-a-Chip at the Transaction Level. *Design Automation for Embedded Systems*, 10(2):73–104, 2005.

[11] Pascal Raymond. Synchronous Program Verification with Lustre/Lesar. *Modeling and Verification of Real-Time Systems*, page 7, 2008.

[12] S Rodriguez-Chavez, AA Palma-Rodriguez, E Tlelo-Cuautle, and SX-D Tan. Graph-based Symbolic and Symbolic Sensitivity Analysis of Analog Integrated Circuits. In *Analog/RF and Mixed-Signal Circuit Systematic Design*, pages 101–122. Springer, 2013.

[13] Guoyong Shi. A Survey on Binary Decision Diagram Approaches to Symbolic Analysis of Analog Integrated Circuits. *Analog Integrated Circuits and Signal Processing*, 74(2):331–343, 2013.

[14] Wikipedia contributors. Model checking — Wikipedia, the free encyclopedia, 2021. [Online; accessed 21-September-2021].