Abstract

In recent times, most leading chip design companies are seriously investigating the possibility of integrating *formal property verification*(FPV) into their pre-silicon validation flows. FPV allows the designer to express the key correctness requirements of a design in terms of formal properties and automatically verify them exhaustively over a given implementation. It is unrealistic to obtain a similar guarantee through simulation driven approaches, in view of the number of intricate directed tests that would be required to achieve full coverage of potential behaviors. This is the main attraction of FPV technology. Unfortunately, existing FPV technology has two major limitations. These are:

1. *Capacity.* Current model checking techniques for FPV do not scale beyond small circuit modules. The state space of circuits typically grow exponentially with the size of the circuits, resulting in *state explosion*. In order to verify a property over a given circuit, a model checking algorithm needs to exhaustively search this space for a counter-example and hence it suffers from the state explosion.

2. *Completeness* FPV guarantees that the given properties are exhaustively checked on the given implementation. It *does not* guarantee that the set of properties are complete, that is, whether these properties cover the design intent. The question – *Have I written enough properties?* – does not yet have an adequate formal answer.

The above limitations of existing FPV technology poses some interesting and potentially revolutionary challenges. One of the most notable among these is the problem of validating a design's architectural intent. At the architectural level, the micro-architects decide on several key architectural features, such as the nature of the instruction and data pipelining, caching and arbitration policies, interrupt modes, etc. It is widely accepted that the major focus of design optimization today is at the architectural level, therefore it is crucial that the main architectural decisions are appropriately interpreted, implemented and validated within the design flow. Most chip design companies specify the architectural intent in a text document (say, English), and also admit that some of the most difficult bugs enter the design due to misinterpretation of the architectural intent.

There are two main challenges in formal design intent verification, namely, (a) expressing the design intent formally, and (b) formally verifying the RTL with respect to the design

intent. For the first, there exists an arsenal of formal property specification languages, including Forspec (of Intel), SystemVerilog Assertions (of Accelera) and Sugar / PSL (of IBM / Accelera), and their underlying temporal logics, namely Linear Temporal Logic (LTL) and Computation Tree Logic (CTL). While it is infeasible to express the complete functionality of a design in terms of formal properties, common experience shows that the important architectural properties can indeed be expressed with these languages.

The thesis addresses quite a few problems keeping the above two challenges as the focus. The problems addressed in this work are as follows:

1. *Checking consistency of reactive system specification.* We have provided an sound algorithm to check realizability for reactive systems. Our algorithm is an approximate algorithm which uses a satisfiability checker as a black box to determine realizability of a specification.

2. *Measuring the completeness of a design specification at the highest level in absence of an implementation.* Our prototype tool *CovAnalyzer* takes properties and the interface signals of the design as input and determines the coverage of faults in the interface lines with respect to the specifications of the design.

3. Providing a method for specification refinement and hence enabling checking system level properties over large designs feasible. This technology will serve as the driving factor for moving formal verification beyond the block level design verification.

4. Designing a framework for verifying formal properties by emulation and also enabling the design of robust safety critical systems by synthesizing formal property checkers directly into the chip.