High-level ATPG: a real topic or an academic amusement?

M. Sonza Reorda
Politecnico di Torino – Dip. di Automatica e Informatica
Torino, Italy
http://www.polito.it/~sonza

In the past years, researchers have published tens of papers whose title was including the magic words “high-level ATPG” or something similar. However, high-level ATPG tools are hardly available on the market, and industries still demand for effective solutions in this area. Why? What would be necessary in order to change this situation?

To answer, we should try to compare what has been proposed and what is (or is going to be) required by industries. When speaking about high-level ATPG, they are normally pushed by four main wishes: being able to perform some testability analysis before the synthesis step (no matter whether it is performed in a manual or automatic manner), generating test vectors for a high-level description when the corresponding gate-level netlist is not available (e.g., with µP cores), reducing the complexity (or increase the efficiency) of the ATPG step performed later at the gate-level, obtaining vectors to be effectively used for design validation.

On the other side, a long sequence of techniques have been proposed, differing among them for several important aspects.

First of all, the adopted fault model: the final effectiveness of each technique is still measured in terms of the Fault Coverage of stuck-at faults reached by the generated vectors on the corresponding gate-level description, but all high-level techniques require a goal to orient their work at the adopted level (RT or behavioral). The fault model can be borrowed from software testing (e.g., statement or path coverage), include some observability issue, or emulate the stack-at fault at a higher level. Depending on the adopted fault model, the complexity of the work to be done changes very much, and so does the relationship between the high-level figures and the corresponding gate-level fault coverage.

The proposed techniques also strongly differ on the kind of input descriptions they accept: apart from the level (behavioral or RT), some of them focus on descriptions which can be synthesized from automatic synthesis tools (thus imposing several constraints especially in terms of allowed constructs and clocking properties), others on special architectures such as pipelined processors; some addresses control-dominated architectures, other focus on data-dominated ones. As it often happens, generality is paid in terms of efficiency, but the restrictions on the input descriptions are not always so clearly stated.

Finally, the proposed techniques adopt very different computational models to solve their problem: popular approaches are the ones based on symbolic computation, classical branch and bound search guided by heuristics, evolutionary strategies. The choice on the model strongly influences the completeness of the reachable solution, the ability to deal with more general descriptions, the maximum size and complexity of the accepted descriptions.

A rich variety of techniques thus exist in the literature: what is still missing for transforming the most effective ones into real products usable by industries? I believe that the answer is two-fold, and is mainly a matter of method to be followed by the research community in supporting activities in this field.

First of all, it is necessary to more clearly define (and bound) the problem which is of interest: high-level ATPG is definitely a too vague definition, and a common agreement on its meaning is still missing, thus making more difficult for good ideas to clearly emerge. Splitting the whole problem into several more specific ones (such as “Automatic Test Pattern Generation for RT-level synthesizable descriptions”, or “Automatic Generation of sequences for Design Validation of behavioral descriptions”) would be a major step forward.

Second, even after a categorization has been performed, it is anyway difficult to seriously evaluate and compare what is being proposed, because we still lack even a basic set of widely accepted and available benchmarks: the effort recently done by Scott Davidson is starting to fill this gap. It is my belief that similar efforts, if specifically devoted to high-level ATPG, and jointly supported by industries and academia, will have a key importance in both defining the problems to be addressed, and identifying the techniques which best solve these problems. Otherwise, high-level ATPG is likely to become only an amusement for academicians.