11 BEHAVIORAL FAULT SIMULATION

Jean-François Santucci
Paul Bisgambiglia
Dominique Federici

11.1 Introduction

Due to the ever-increasing complexity of VLSI circuits, the use of VHDL [Vhd87] behavioral descriptions in the fields of test generation and fault simulation becomes advised. In order to understand this evolution and the context in which it is efficient, the increasing complexity of VLSI circuits must be considered. To better cope with the complexity of existing and future systems, higher abstraction levels must be taken into account. Furthermore, the only knowledge about the device being tested may come from data sheets or signal measurements. In this case the only way to generate test patterns is behavioral testing. Today, such a task is done manually and one of the main interests in Behavioral Test Pattern Generation (BTPG) is to automate it. Another point to consider is that the test generation process is an integral part of the design process and its implementation must begin during the behavioral design phase.

Motivated by these facts, various BTPG methods have been proposed in the recent past [Bar87,One90,Nor89,Hum91,Cou92,Stra93]. Most of these BTPG methods are based on a deterministic generation process using a formal fault model: each sequence is constructed in order to detect a given fault belonging to a Fault List determined from a Behavioral Fault Model.

Despite the use of heuristic criteria [Che91,San92,San93], to speed up the search for test sequences, these methods remain very time-consuming because of the
deterministic nature of the BTPG processes and the lack of Behavioral Fault Simulation (BFS) methods.

In order to facilitate the definition of a powerful BFS method, it is essential to explicitly represent the concepts involved in behavioral descriptions. We have therefore defined an internal model which highlights on the one hand the sequential and concurrent aspects and on the other hand the separation and interaction between the control flow and the data flow.

The list of considered behavioral fault hypothesis is derived using two steps: (i) definition of an exhaustive fault model which collects all possible fault hypothesis in terms of the incorrect «functioning» of modeling items; (ii) selection of a sub-set of conventional faults [Gho91].

Our approach for defining a BFS method leans on the resolution of the three following sub-problems in order to deal with high-level behavioral descriptions:

- 1- determination of the high level basic elements of the internal model which will have a key role in BFS.
- 2- definition of how lists of behavioral faults are propagated through the previous basic elements. The simulation of a basic element requires deducing the fault list at the basic element outputs from the input faults lists.
- 3- development of an overall algorithm allowing to propagate fault lists through the complex data and control structures involved in the internal model.

The BFS method has been implemented in C++ language and first results have been obtained.

In the next section of the chapter we deal with the main concepts involved in behavioral test pattern generation. Section 3 presents the motivation in behavioral testing. The interest of using of a behavioral fault simulator coupling with a deterministic and a pseudo-random behavioral test pattern generator is detailed in section 4. Our approach for performing BFS is given in section 5. Section 6 deals with implementation and results. Perspectives are overviewed in a concluding part.

### 11.2. What is Behavioral Testing

#### 11.2.1 Concepts of Behavioral Testing

Testing is a major activity in CAD (Computer Aided Design) systems for digital circuits. For that reason, CAD systems include ATPG (Automatic Test Pattern Generation) tools. ATPG techniques can be divided into two families depending on the model type of the circuit under test: structural or behavioral.

The structural view or "white box" represents a potential structure of the studied system as an interconnection of basic elements. This structure is qualified as potential because there is not necessarily a complete mapping between the model