HIERARCHICAL TEST GENERATION
FOR DIGITAL SYSTEMS

Marina Brik, Gert Jervan, Antti Markus,
Jaan Raik and Raimund Ubar

Technical University of Tallinn
Ehitajate tee 5, EE0026, Tallinn
ESTONIA
Fax: (+372) 620 2253, e-mail: raiub@pld.ttu.ee

ABSTRACT

A hierarchical test generator for digital systems described on register-transfer (RTL) and
gate levels is presented. The system is supposed to consist of control and data parts
coupled with global feedback. The generator implements a novel test generation approach
based on using multiple abstraction levels of alternative graph (AG) models. The uniform
AG representation allows application of common modelling methods and procedures on
all abstraction levels. Experimental results showing the efficiency of the approach are
provided.

1. INTRODUCTION

Test generation for real life digital circuits on the gate-level is extremely complex. It has
been shown that test generation for combinational circuits is an NP-complete problem [1].
Gate-level test generation for sequential circuits is even more complex and remains still
an unsolved problem in practice. During recent times, as a possible solution, hierarchical
test generation methods have evolved [2-5] which take advantage of higher abstraction
levels (behavioural or register-transfer (RT) levels) information while generating tests for
gate-level faults. The system is considered at different levels, and tests are created on these
levels by separate tools. Both, top-down and bottom-up strategies are known. In the
bottom-up approach, tests generated at the lower level will be later assembled at the higher
level. Current paper considers the top-down approach, where constraints extracted at
higher level [6] are considered when deriving tests for the lower level. In the approach
discussed below, different design abstraction levels of the system are represented by
alternative graph (AG) models [7,8]. This feature provides for a uniform model
representation and an application of common procedures throughout the levels. As the
result, the complexity of the problem can be reduced and the efficiency of the ATPG will
be increased.

The paper is organized as follows. Section 2 explains the concept of AGs for representing
digital systems at different abstraction levels, Section 3 describes the structure of the test
generator, and in Section 4, experimental results are given.
2. THE MODEL

In this paper an approach based on AGs (or decision diagrams) for test generation is used. AGs serve as a mathematical basis for solving a wide spectrum of test tasks, resulting in a uniform fault model and a restricted set of standardized procedures. AGs were proposed the first time for test generation in [9]. Unlike the analogical binary decision diagrams (BDD) [10] introduced for representing Boolean functions, AGs describe both the functions and the structural features of a circuit or a system.

AGs can be regarded as a way of representing programs (procedures, algorithms), or as a data structure to be manipulated by programs, or as a way to concisely represent test knowledge that we have about the system. This universality of AGs makes it possible to transform different information we have about the system easily and directly to the form which is most suitable for solving test design or diagnosis tasks. AGs describe digital systems on mixed logical and functional levels, which can include random logic, traditionally treated at the gate level, as well as digital systems like microprocessors, controllers etc., traditionally described at the procedural or RT levels. The fault model developed for AGs covers in a uniform way a wide class of faults represented at different levels like stuck-at faults, opens, shorts, functional faults [11], faults for VHDL descriptions [12] etc. The fault model defined on AGs can be regarded as a generalization of the classical gate-level stuck-at fault model [7].

Alternative graph is defined as a non-cyclic directed graph whose nodes are labelled by variables, constants or algebraic expressions. For each combination of values for node variables there exists always a corresponding activated path from the starting node to some terminal node. This relationship describes a mapping from a Cartesian product of the sets of values of all node variables to the joint set of values of labels in the terminal nodes. Therefore, by AGs it is possible to represent arbitrary digital functions \( Y = F(X) \), where \( Y \) is the variable whose value will be calculated on the AG and \( X \) is the vector of all variables which belong to the labels of the nodes in the AG.

When using AGs to describe complex digital systems, we have, at the first step, to represent the system by a suitable set of interconnected components (combinational or sequential ones). At the second step, we have to describe these components by their corresponding functions which can be represented by AGs. AGs which describe digital systems at different levels may have special interpretations, properties and characteristics, however, the same formalism and the same algorithms for test and diagnosis purposes can be used, which is the main advantage of AGs.

![AG Representation of a Data Path](image-url)