FUNCTIONAL VERIFICATION: APPROACHES AND CHALLENGES

A. MOLINA† and O. CADENAS‡

† Computer Architecture Department, Universitat Politècnica de Catalunya, Barcelona, Spain
amolina@ac.upc.edu
‡ School of System Engineering, University of Reading, Reading RG6 6AY, UK
o.cadenas@reading.ac.uk

Abstract—It's a fact that functional verification (FV) is paramount within the hardware’s design cycle. With so many new techniques available today to help with FV, which techniques should we really use? The answer is not straightforward and is often confusing and costly. The tools and techniques to be used in a project have to be decided upon early in the design cycle to get the best value for these new verification methods. This paper gives a quick survey in the form of an overview on FV, establishes the difference between verification and validation, describes the bottlenecks that appear in the verification process, examines the challenges in FV and exposes the current FV technologies and trends.

Keywords—Functional verification, Simulation-based verification.

I. INTRODUCTION

Functional verification (FV) is a necessary step in the development of today’s complex digital designs. Hardware complexity growth continues to follow Moore’s Law (Moore, 1965), but verification complexity is even more challenging. In fact, it theoretically rises exponentially with hardware complexity doubling exponentially with time (Dempster and Stuart, 2001). FV is widely acknowledged as a major bottleneck in design methodology: up to 70% of the design development time and resources are spent on FV (Fine and Ziv, 2003). Recent study highlights the challenges of FV (Mishra and Dutt, 2005): Figure 1 shows the statistics of the SOC designs in terms of design complexity (logic gates), design time (engineer years), and verification complexity (simulation vectors) (Spirakis, 2004). Recently all major EDA companies in the electronic sector are aggressively targeting the verification process with new and better EDA tools and a substantial number of seminars and workshops. This paper offers a quick review on what is being agreed is this rapid evolving area by examining recent references generated both by industry and the academia. The paper is organized as follows. Section 2 establishes the difference between verification and validation. Section 3 describes the bottlenecks that appear in the verification process. Section 4 examines the challenges in FV followed by current FV technologies and trends in Section 5. Finally, Section 6 gives some remarks as conclusions.

II. VERIFICATION VERSUS VALIDATION

Kropf (1997) defines “validation” as the “process of gaining confidence in the specification by examining the behavior of the implementation.” Recently there was discussion on the subject of “verification versus validation”. Many views were presented regarding the difference. One view was that “validation ensures it’s the right design; while verification ensures that the design is right” (Verification Guild Website, 2006). Another view was “verification means pre-silicon testing (Verilog/VHDL simulations) while validation is post-silicon testing (testing silicon on boards in the laboratory)”.

Whether it is validation or verification, two things need to happen to ensure that the silicon meets the specification: (1) The chip specification is interpreted correctly (typically through documentation and sometimes modeling). (2) The interpretation is captured and implemented correctly (typically through HDL) and synthesized into silicon and packaged as a chip. For the purposes of this article we will consider the second step as verification, and the first step as validation.

III. BOTTLENECKS

A. Design bottleneck

Design time is a function of silicon complexity. This gives rise to system complexity, which affects time to market, as shown in Fig. 2.

Fig. 1. The study highlights the tremendous complexity faced by simulation-based validation of complex SOCs: it estimates that by 2007, a complex SOC will need 2000 engineer years to write 25 million lines of register-transfer level (RTL) code and one trillion simulation vectors for functional verification.