Hardware Verification: Techniques, Methodology and Solutions

Sharad Malik

Dept. of Electrical Engineering
Princeton University
Princeton, NJ 08544, USA
sharad@princeton.edu

Abstract. Hardware verification has been one of the biggest drivers of formal verification research, and has seen the greatest practical impact of its results. The use of formal techniques has not been uniformly successful here — with equivalence checking widely used, assertion-based verification seeing increased adoption, and general property checking and theorem proving seeing only limited use. I will first examine the reasons for this varied success and show that for efficient techniques to translate to solutions they must be part of an efficient methodology and be scalable. Next I will describe specific efforts addressing each of these critical requirements for the verification of emerging computing systems.

A significant barrier in enabling efficient techniques to flow into efficient methodology is the need for human intervention in this process. I argue that in large part this is due to the gap between functional design specification, which is still largely in natural language, and structural design description at the register-transfer level (RTL). This gap is largely filled by humans, leading to a methodology which is error-prone, incomplete and inefficient. To overcome this, we need formal functional specification and a way to bridge the gap from this specification to structural RTL. In this direction I will present a modeling framework with design models at two levels — architectural and microarchitectural. The architectural model provides for a functional specification, and the microarchitectural model connects this to a physical implementation. I will illustrate how this enables greater automation in verification.

A major challenge in verification techniques providing scalable solutions is the inherent intractability of the problem. This is only getting worse with increasing complexity and is reflected in the increasing number of bug escapes into silicon. I argue that existing verification solutions need to be augmented with runtime validation techniques, through online error-checking and recovery in hardware. I will illustrate this with examples from emerging multi-core architectures. I will also discuss the complementary roles of formal techniques and runtime validation in a cooperative methodology.

C.R. Ramakrishnan and J. Rehof (Eds.): TACAS 2008, LNCS 4963, p. 1, 2008.