

Preface

The growing complexity of modern processor designs and their shrinking production schedules cause an increasing number of errors to escape into released products. Many of these escaped bugs can have dramatic effects on the security and stability of consumer systems, undermine the image of the manufacturing company and cause substantial financial grief. Moreover, recent trends towards multi-core processor chips, with complex memory subsystems and sometimes non-deterministic communication delays, further exacerbate the problem with more subtle, yet more devastating, escaped bugs. This worsening situation calls for high-efficiency and high-coverage verification methodologies for systems under development, a goal that is unachievable with today's pre-silicon simulation and formal validation solutions. In light of this, functional post-silicon validation and runtime verification are becoming vitally important components of a modern microprocessor development process. Post-silicon validation leverages orders of magnitude performance improvements over pre-silicon simulation while providing very high coverage. Runtime verification solutions augment the hardware with on-chip monitors and checking modules that can detect erroneous executions in systems deployed in the field and recover from them dynamically.

The purpose of this book is to present and discuss the state of the art in post-silicon and runtime verification techniques: two very recent and fast growing trends in the world of microprocessor design and verification. The first part of this book begins with a high-level overview of the various verification activities that a processor is subjected to as it moves through its life-cycle, from architectural conception to silicon deployment. When a chip is being designed, and before early hardware prototypes are manufactured, the verification landscape is dominated by two main groups of techniques: simulation-based validation and formal verification. Simulation solutions leverage a model of the design's structure, often written in specialized hardware programming languages, and validate a design by providing input stimuli to the model and evaluating its responses to those stimuli. Formal techniques, on the other hand, treat a design as a mathematical description of its functionality and focus on proving a wide range of properties of its functional behavior. Unfortunately, these two categories of validation methods are becoming increasingly inadequate

in coping with the complexity of modern multi-core systems. This is exactly where post-silicon and runtime validation techniques, the primary scope of this book, can lend a much needed hand.

Throughout the book we present a range of recent solutions in these two domains, designed specifically to identify functional bugs located in different components of a modern processor, from individual computational cores to the memory subsystem and on-chip fabrics for inter-core communication. We transition into the second part of the book by presenting mainstream post-silicon validation and test activities that are currently being deployed in industrial development environments and outline important performance bottlenecks of these techniques. We then present *Reversi*, our proposed methodology to alleviate these bottlenecks in processor cores. Basic principles of inter-core communication through shared memory are overviewed in the following chapter, which also details new approaches to validation of communication invariants in silicon prototypes. We conclude the discussion of functional post-silicon validation with a novel technique, targeted specifically to modern multi-cores, called *Dacota*.

The recently proposed approaches to validation that we collected in part two of this book have an enormous potential to improve verification performance and coverage; however, there still is a chance that complex and subtle errors evade them and escape into end-user silicon systems. Runtime solutions, the focus of the third part of this work, are designed to address these situations and to guarantee that a processor performs correctly even in presence of escaped design bugs without degrading user experience. To better analyze these techniques we investigate the taxonomy of escaped bugs reported for some of the processor designs available today, and we also classify runtime approaches into two major groups: *checker-* and *patching-based*. In the remainder of part three we detail several runtime verification methods within both categories, first relating to individual cores and then to multi-core systems. We conclude the book with a glance towards the future, discussing modern trends in processor architecture and silicon technology, and their potential impacts on the verification of upcoming designs.



<http://www.springer.com/978-1-4419-8033-5>

Post-Silicon and Runtime Verification for Modern
Processors

Wagner, I.; Bertacco, V.

2011, XVII, 224 p., Hardcover

ISBN: 978-1-4419-8033-5