1. Record Nr. UNINA9911006781103321

Autore	Roychoudhury Abhik
Titolo	Embedded systems and software validation / / Abhik Roychoudhury
Pubbl/distr/stampa	Amsterdam ; ; Boston, : Morgan Kaufmann Publishers/Elsevier, c2009
ISBN	1-282-25804-4 9786612258046 0-08-092125-6
Descrizione fisica	1 online resource (267 p.)
Collana	The Morgan Kaufmann series in systems on silicon
Disciplina	004.1
Soggetti	Embedded computer systems - Design and construction Embedded computer systems - Testing Computer software - Testing
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Description based upon print version of record.
Nota di bibliografia	Includes bibliographical references (p. 233-239) and index.
Nota di contenuto	Front Cover; Embedded Systems and Software Validation; Copyright Page; Dedication Page; Table of Contents; Acknowledgments; Preface; Chapter 1. Introduction; Chapter 2. Model Validation; 2.1 Platform versus System Behavior; 2.2 Criteria for Design Model; 2.3 Informal Requirements: A Case Study; 2.3.1 The Requirements Document; 2.3.2 Simplification of the Informal Requirements; 2.4 Common Modeling Notations; 2.4.1 Finite-State Machines; 2.4.2 Communicating FSMs; 2.4.3 Message Sequence Chart-Based Models; 2.5 Remarks About Modeling Notations; 2.6 Model Simulations; 2.6.1 FSM Simulations 2.6.2 Simulating MSC-Based System Models2.7 Model-Based Testing; 2.8 Model Checking; 2.8.1 Property Specification; 2.8.2 Checking Procedure; 2.9 The SPIN Validation Tool; 2.10 The SMV Validation Tool; 2.11 Case Study: Air-Traffic Controller; 2.12 References; 2.13 Exercises; Chapter 3. Communication Validation; 3.1 Common Incompatibilities; 3.1.1 Sending/Receiving Signals in Different Order; 3.1.2 Handling a Different Signal Alphabet; 3.1.3 Mismatch in Data Format; 3.1.4 Mismatch in Data Rates; 3.2 Converter Synthesis; 3.2.1 Representing Native Protocols and Converters 3.2.2 Basic Ideas for Converter Synthesis3.2.3 Various Strategies for Protocol Conversion; 3.2.4 Avoiding No-Progress Cycles; 3.2.5

Speculative Transmission to Avoid Deadlocks; 3.3 Changing a Working Design; 3.4 References; 3.5 Exercises; Chapter 4. Performance Validation; 4.1 The Conventional Abstraction of Time; 4.2 Predicting Execution Time of a Program; 4.2.1 WCET Calculation; 4.2.2 Modeling of Microarchitecture; 4.3 Interference within a Processing Element; 4.3.1 Interrupts from Environment; 4.3.2 Contention and Preemption; 4.3.3 Sharing a Processor Cache
4.4 System-Level Communication Analysis4.5 Designing Systems with Predictable Timing; 4.5.1 Scratchpad Memories; 4.5.2 Time-Triggered Communication; 4.6 Emerging Applications; 4.7 References; 4.8 Exercises; Chapter 5. Functionality Validation; 5.1 Dynamic or Trace-Based Checking; 5.1.1 Dynamic Slicing; 5.1.2 Fault Localization; 5.1.3 Directed Testing Methods; 5.2 Formal Verification; 5.2.1 Predicate Abstraction; 5.2.2 Software Checking via Predicate Abstraction; 5.2.3 Combining Formal Verification with Testing; 5.3 References; 5.4 Exercises; Bibliography; Index

Sommario/riassunto

Modern embedded systems require high performance, low cost and low power consumption. Such systems typically consist of a heterogeneous collection of processors, specialized memory subsystems, and partially programmable or fixed-function components. This heterogeneity, coupled with issues such as hardware/software partitioning, mapping, scheduling, etc., leads to a large number of design possibilities, making performance debugging and validation of such systems a difficult problem. Embedded systems are used to control safety critical applications such as flight control, automotive el