1. Record Nr. UNINA9910808577703321

Autore	Oshana Robert
Titolo	Software engineering for embedded systems : methods, practical techniques, and applications / / Robert Oshana, Mark Kraeling
Pubbl/distr/stampa	Amsterdam ; ; Boston, : Elsevier/Newnes, 2013 Waltham, MA : , : Newnes, , 2013
ISBN	1-299-45228-0 0-12-415941-9
Edizione	[1st ed.]
Descrizione fisica	1 online resource (xlix, 1150 pages) : illustrations (some color)
Collana	Expert guide
Disciplina	006.22
Soggetti	Software engineering Embedded computer systems
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	"Expert guide"--Page 1 of cover.
Nota di bibliografia	Includes bibliographical references and index.
Nota di contenuto	Front Cover; Software Engineering for Embedded Systems; Copyright Page; Contents; Software Engineering for Embedded Systems: A Roadmap; Foreword to Software Engineering for Embedded Systems; Acknowledgments; About the Editors; About the Authors; 1 Software Engineering of Embedded and Real-Time Systems; Software engineering; Embedded systems; Embedded systems are reactive systems; Real-time systems; Types of real-time systems - soft and hard; Differences between real-time and time-shared systems; Examples of hard real-time Based on signal sample, time to perform actions before next sample arrivesHard real-time systems; Real-time event characteristics; Real-time event categories; Efficient execution and the execution environment; Efficiency overview; Resource management; Challenges in real-time system design; Response time; Recovering from failures; The embedded system software build process; Distributed and multi-processor architectures; Software for embedded systems; Super loop architecture; Power-save super loop; Window lift embedded design; Hardware abstraction layers (HAL) for embedded systems; Summary 2 Embedded Systems Hardware/Software Co-DevelopmentToday's embedded systems - an example; HW/SW prototyping users; HW/SW prototyping options; Prototyping decision criteria; Choosing the right

prototype; Industry design chain; The need to change the design flow; Different types of virtual prototypes; A brief history of virtual prototypes; The limits of proprietary offerings; What makes virtual prototypes fast; Standardization: the era of SystemC TLM-2.0; SystemC TLM-2 abstraction levels; Architecture virtual prototypes; Software virtual prototypes
Summary - the growing importance of virtualization3 Software Modeling for Embedded Systems; When and why should you model your embedded system?; Modeling; What is a modeling language?; Examples of modeling languages; The V diagram promise; So, why would you want to model your embedded system?; When should you model your embedded system?; Mission- and safety-critical applications; Highly complex applications and systems; Operational complexity; Cost of defect versus when detected; Large development teams require modeling; Modeling is often the only choice
So - modeling is great, but aren't all models wrong?You have your prototype - now what?; Conclusion; Next steps - try it!; Closed-loop control with a DC motor; Learn more about prototyping with a downloadable kit; Designing applications with the NI Statechart Module; Design and simulate a brushed dc motor h-bridge circuit; Multi-domain physical modeling with open-source Modelica models; References; 4 Software Design Architecture and Patterns for Embedded Systems; Overview of architecture and design; Architecture is about system-wide optimization; Three levels of design
What are design patterns?

Sommario/riassunto

Software Engineering for Embedded Systems clearly explains the software engineering tools and techniques needed to optimally design and implement embedded systems in contexts sure as networking, storage, and automotive applications. Written by experts with a solutions focus, this encyclopedic reference is a useful aid to tackling typical problems and issues, including: Architecture and design patternsHardware interfacesEmbedded operating systems, including Linux and AndroidMemory, performance, and power optimizationUser interface consi