Preface

The last few years have seen a complete renovation of the way the general public communicates. Not that long ago there was a clear separation of the environments for different types of communication: personal computers, laptops, and netbooks are traditionally used as the primary instruments for communication requiring data networks, for example browsing the World Wide Web and using electronic mail and social media. Television sets are considered the *de facto* environment for watching TV shows and movies, and phones for placing voice calls. In terms of telephony, already some time ago, the world has seen the shift from copper-wired household telephones shared by everyone in the family to personal wireless cellular phones that permit the user to make phone calls in practice everywhere and on the move. The development of cellular telephony first made it possible to start sending short messages with the phones and later enabled data connections to wireless mobile devices. As the capabilities of mobile communication improved, mobile telephone users developed a desire for more and more services for their devices. This has lead to the contemporary trends of convergence in the ways we communicate: first, one can see a user need for wireless communication with universal and always-accessible internet availability. Second, there is a desire for having uniform terminal equipment for most, if not all, communication types, and third, users want to have reasonably-sized battery-operated terminals with tolerable standby times. A modern hand-held mobile communication device is now a small embedded computer optimized for the different types of communication the user often engages in. Smartphones are nowadays used as personal calendars, game consoles, remote controls, music players, social media terminals, e-mail and Web clients, and even televisions, in addition to their original historical usage scenario of placing and receiving phone calls. This makes the systems resemble more computers and home entertainment systems than telephones. The convergence is not leading to a situation where a specific type of communication is taken completely away from its traditional environment, but it is leading more to providing an adequate secondary way of using a variety of communication applications in one device. It is still most users' preference to watch television broadcasts and movies on a large TV screen rather than on a small smartphone screen, but the ability to watch the news broadcast on-demand on a smartphone while on the move is a definite added value to traditional telephony. Similarly, a television user may want the ability to make video calls and browse Web pages using the television set while preferring to make video calls using a smart phone and to browse the Web using a computer.

The world is currently seeing the first stages of this paradigm shift in communication in many parts of the world. In the heart of this development is the active research performed in the field of embedded and real time communication systems. The disciplines of computer science, computer engineering, telecommunication, and communication engineering are well established and scientists, researchers, and industry professionals in these disciplines are numerous in all continents. The convergence of these disciplines into embedded and real-time communication systems is a natural development as can be seen for example in the smartphone and tablet industries today. The field is interdisciplinary in scope, binding together research from the mentioned disciplines with focus on how they converge to embedded and real-time systems for the communication application domain.

In this development towards more computer-like mobile communication devices, that is, embedded and real-time communication systems, the system-level design process and the design of hardware and software components of the system are facing brand new challenges: despite the small size and reliance on battery power, the devices need to be able to perform ever more complex operations and tasks while maintaining a low enough pricing scheme to ensure adequate market interest. The manufacturers and their design teams are therefore constantly forced to find a balance between adequate performance of the device (the device may not be too slow but it should not provide excess processing performance beyond its application range), low enough manufacturing and design costs, and a short time-to-market. Inherent to all these development trends is the dramatically increased importance of system security (for example, security of the data stored in the device) and communication security (for example, connections through each available network need to be secure). These trends drive the processes of modeling and designing communication systems towards complex and secure communication-enabled embedded systems at an increasing speed.

Forthcoming research in embedded and real-time communication systems needs to target the challenges in future complex converged wireless systems by adventurous development and technological exploration, and experimentation with novel technologies, systems, and system design methodologies. Key research areas in this respect are embedded system design, communication system design, system-wide security, and hardware/software co-design, producing results that converge into novel technologies usable in future secure embedded communication systems. In Embedded system design the software design process needs to take into account special characteristics of the underlying hardware technology. Typically, the data processing resource and the storage resource are significantly more limited than in desktop computers. These limitations follow from the fact that the design goal in embedded systems is usually a small power-efficient device that is able to perform the planned application just within the given time and storage constraints and does not need to be easily extendable to more complex applications. Likewise, the hardware design process needs as input the assumed complexity of the application software to be run in the system so that the hardware design process would produce a device with adequate processing and storage resources. Clearly the hardware and software design processes depend on each other, and how each of them affects the other one is not trivial. A research discipline within embedded system design called hardware/software co-design emerged a decade ago to deal with the problems of interconnecting the two processes, partitioning the target application into hardware and software, and making the partitioning as late as possible to enable incremental addition of details to the specification of the target system independent of hardware and software design issues.

Ideally, the embedded system design process would start from an executable specification written in a compilable programming language. The original executable specification would later serve as a reference model for each stage of the design process. The specification would be refined throughout the design process, ultimately becoming a system model with enough detail for automatic hardware and software generation within given design constraints, producing an HDL hardware description (optionally based on a provided base hardware platform) and a software implementation for the produced hardware description. The resulting hardware/software partitioned model should then be verified by automated comparison to the original reference model. Although hardware/software co-design has been an active research discipline for more than a decade, and there are commercial toolsets for it in existence, the ideal flow described above is not adequately achievable as many of the key transitions from higher design abstractions to lower ones still require significant amounts of manual work for optimal results, and therefore they are often seen to not reduce the designer's workload as much as one would hope. Alternatively, to achieve relaxed transitions from higher abstraction levels to lower ones, the designer would often be forced to use a very limited subset of the modeling language, which can easily be seen to severely limit the innovativeness of novel designs. One possible research aim to deal with these issues would be to develop hardware/software co-design methods and flows for embedded systems further towards the ideal flow outlined above, focusing especially on domain-specific solutions: for example, the embedded communication systems application domain.

In Communication system design, issues like quality of sound, spectrum utilization efficiency, energy used per amount of transmitted information, and digital transmissions have lead to a number of new technologies to be taken into account in the design process. Already a long ago it has been seen that frequency modulation (FM) improves the quality of analog audio transmissions, and it has been used for a number of radio communication systems, like FM radio and VHF radio systems. Currently, several radio systems originally launched as analog services, such as mobile phone and television networks, have been converted into digital transmission systems in many parts of the world. Also, newer services like wireless local and metropolitan area computer networks have been in the digital domain already since their introduction to general audiences. Common to these digital transmission systems is that in each of them, a digitized data stream is sent over a radio link. This digital data stream might correspond to voice audio, video signals, packet data, and so on. Not only is the data stream sent over the network digitally, but the actual radio device modulating the data for transmission over the air is digital nowadays. An increasing part of receivers are designed using pure digital technology, using either Application Specific Integrated Circuits (ASICs), Digital Signal Processors (DSPs), Field Programmable Gate Arrays (FPGAs), or a combination of these and general purpose processors. In these solutions, only the radio frequency parts work using analog technology; the intermediate frequency (IF) is fed into a fast analog-to-digital converter (ADC), and the rest of the processing is done digitally.

In designing small embedded communication devices that support multiple wireless technologies and also provide many other features unrelated to the communication tasks, the traditional solution of using general purpose programmable microprocessors (CPUs) with accompanying application-domain-specific ASICs (application-specific integrated circuits) is neither an efficient nor a flexible enough solution to handle the increasing demands. The lack of flexibility in such systems is mostly due to the often minimal programmability of ASICs; they are designed for a specific application and are either minimally or not at all capable of performing other similar applications. In addition to the lack of flexibility in such systems, devices consisting of general purpose processors and accompanying ASICs may require excess power (depending on the CPU used) or may cause unnecessary enlargement of overall device size due to the area occupied on the circuit board. One suggested solution to this could be to use FPGAs (field programmable gate arrays) that make true hardware reconfigurability possible. Ideally, with FPGAs, it could be possible to store multiple hardware configurations in the device memory or download new hardware configurations when necessary. This way, a new configuration could be taken into use whenever the system needs to switch from one application or wireless networking technology to another (for example, switching from WLAN to one of the fast 3G communication protocols or vice versa). The clear benefit of this technology is that overall device size could be very small since the application or technology specific circuitry (separate ASICs for each technology) would be replaced ideally by just one reconfigurable circuit. However, the "on-the-fly" reconfiguration of FPGAs as of today is not usually seen feasible to facilitate a roaming user switching from one radio network to another. Efficient reconfigurability techniques should be explored and developed in the course of forthcoming research as the reconfigurability speed, and also, its usability and cost can be expected to improve to tolerable levels in the future.

Software Defined Radio is the term for software and hardware technologies that enable reconfigurable radio systems. SDR-enabled devices can be dynamically programmed to reconfigure the characteristics of the equipment. An even wider concept is Open Wireless Architecture (OWA), which by definition is an elastic, extensible, and open broadband platform that can support diverse communication standards and can integrate multiple wireless networks. To achieve this flexibility, OWA focuses on all aspects of a communication system such as the RF section, baseband processing, and application domain. Software Defined Radio (SDR) is one of the building blocks of OWA. The goal is that many systems could use a common hardware platform that could be configured in software to function in a desired way in multiple applications. Provided a well-defined base hardware platform, new radio systems could be implemented in a robust way utilizing the ideal principles introduced in SDR and OWA. A well-defined platform should both provide the functional blocks that the radio system requires (as a reconfigurable on-chip communication system) and enable data transports between the functional blocks. Adventurous research to find new technologies to design such SDR and OWA enabled platforms for future converged wireless platforms is an important research area for future embedded and real-time communication systems.

System and communication security is in the heart of the trends of having the ability to connect a single device to multiple networks that differ significantly from each other and on the other hand having small hand-held devices that automatically try to establish network connections whenever they detect a network service is available. In an operating environment like this, the importance of guaranteed data security, data and user authenticity, and data transmission security is becoming more and more crucial every day. Even if much of the information moving in the networks is public, there is always information that for varying reasons must be made unavailable to unauthorized actors on the network. The traditional wired networks could in principle be made totally secure by ensuring that non-authorized actors never have physical access to the network. However, this is seldom the case, and in a wireless environment, basically anyone near enough can be eavesdropping on the communication. Hence, methods are needed for making it impossible, or very difficult, for eavesdroppers to actually use the data being sent over the communications link. Also, as the embedded communication devices nowadays hold an increasing amount of personal information of the owner of the device that could be taken advantage of in identity thefts and financial frauds, the importance of protecting the data stored into the device from misuse for example in cases of device theft or device loss is of extremely high importance.

Traditionally, cryptographic systems relied on a simple algorithm and a lengthy encryption key, like for example in the “one-time-pad”-type encryption, where the algorithm is a simple bitwise XOR operation but the key is as long as the data to be encrypted. Modern cryptosystems have switched to algorithmic complexity and shorter, for example 256, 512, or 1024 bit keys, the length of which is independent from the length of the data to be encrypted. The goal is to have more manageable keys and a software implementation of the encryption algorithm. However, currently the speed of computing systems is forcing the keys to be made longer again to fight brute force attacks on the encryption algorithms. For example, in wireless local area networks, this problem has been seen already some time ago: the WEP encryption of such a network can be broken by brute force in a few minutes by means of cryptanalysis (although one must recall that the WEP encryption method is also flawed by design, resulting in its breaking being easier than without the design flaws). As encryption keys are made longer, and solid and safe encryption algorithms are used, brute force attacks will take more and more time and potentially become unfeasible again. However, longer keys also generate considerably higher processing requirements for the encrypting and decrypting devices that more and more often today are battery-powered handheld devices. Clearly, one could argue that the history is repeating itself: a pure software implementation on a general purpose microprocessor might no longer be feasible in a hand-held system, and therefore, effort should be placed in researching encryption application-domain-optimized ASIP-type programmable processors that would move the most calculation-intensive parts of the encryption algorithm into domain-optimized hardware execution units of the processor.

The integration of state-of-the-art and future encryption features into embedded communication systems as outlined above is a very important and contemporary research area requiring focus from scientists and engineers in the discipline. Also, the development of novel cryptosystems is a well-established research discipline in the mathematical sciences, and analyzing the feasibility of novel and future cryptosystems for use in embedded communication systems, developing encryption application domain aware hardware platforms and technologies, and based on this work, integrating cryptographic features into embedded communication systems, are research areas requiring attention to facilitate secure embedded and real-time communication system design in coming years.

Convergence of the technologies described in the previous pages is the future of communication devices and their design. The emerging variety of wireless systems and the increase in features that are not directly related to the main communication task are a vital challenge for developers designing novel embedded communication devices. In terms of the communication part, the trend seems to be to adopt SDR and OWA methodologies, whereas the parts unrelated to communication are developed more like computer programs. Both parts of the system require more and more research effort also in the area of data and data transmission security to ensure safe storage of information and safe communication between communicating parties.

This book is a summation volume of articles published in the *International Journal of Embedded and Real-Time Communication Systems* (IJERTCS) in its first volume year, 2010. The journal has an interdisciplinary scope, binding together research from different disciplines, with focus on how the disciplines converge to embedded and real-time systems for the communication application domain. The subject coverage of the journal is broad, which enables a clear presentation of how the research results presented in the journal benefit the convergence of embedded systems, real-time systems, and communication systems. The journal is aimed to benefit scientists, researchers, industry professionals, educators, and junior researchers like PhD students in the embedded systems and communication systems sector. An important aim is to provide the target audience with a forum to disseminate and obtain information on the most recent advances in embedded and real-time communication systems research: to give the readers the opportunity to take advantage of the research presented in the journal in their scientific, industrial, or educational purposes.

As a journal in the focal point of disciplines such as computer science, computer engineering, telecommunication, and communication engineering, the *International Journal of Embedded and Real-Time Communication Systems* is positioned well to provide its readership with interesting and well-focused articles based on recent high-quality research. The journal's coverage in topics from embedded systems, real-time systems, to communications system engineering, and especially how these disciplines interact in the field of embedded and real-time systems for communication, offers its readership both theoretical and practical research results facilitating the convergence of embedded systems, real-time computing, and communication system technologies and paradigms.

This book begins with an introductory chapter by Jouni Isoaho, *et al.* In this chapter, titled "Current Challenges in Embedded Communication Systems," the technological challenges brought by future embedded communication systems are investigated based on current research and trends in the domain. The authors define the key challenges and proceed to proposing potential solutions to deal with the challenges through an analysis of the development trends and historical evolution of embedded systems. The proposed solutions are gathered around three key research areas and challenges for future embedded communication systems: technologies for self-aware autonomous systems, embedded security, and device interoperability. Solutions to the challenges are suggested to be found in the areas of verification and modeling methods, system implementation platforms, and security-enabled designs flows.

After the introductory chapter, the book is organized into four thematic sections. The section on **Technologies for Embedded Communication Systems** focuses on novel technological advances for embedded communication systems. Many different aspects of technological advances are covered: for example, programming techniques, energy efficiency, cache technology, and memory interfaces for embedded communication systems. The selection of topics clearly highlights the multidisciplinary nature of this research area; embedded and real-time communication systems are positioned in the focal points of research of several different Information Technology disciplines.

The second thematic section of this book is **Mobile Communication Applications**. As discussed in the previous pages, it is not only the physical device that constitutes the embedded communication system, but actually it is the applications and the application domain that define the features and functionality of the system. In this thematic section, a selection of important application areas of embedded communication systems are studied, most notably the Long-Term Evolution (LTE) high speed wireless network protocol processing in mobile devices, signal processing algorithms for air interfaces and networks built into devices to interconnect the internal modules and transport application data between the internal modules the device is made of, that is, embedded networks.

The third thematic section is **On-Chip Communication**, where focus is on the research problems regarding reliable communication inside a chip; for example, between the different on-chip cores that together form a Multi-Processor System-on-Chip (MPSoC). An on-chip communication implementation itself can be seen as an embedded and real-time communication system, but very often today, MPSoCs are an essential building part of some larger embedded communication system. The efficient and reliable implementation of on-chip communication is essential for ensuring excellence in device performance and variety in its operating capabilities, and is thus in the heart of the research areas needing scholarly focus for future communication systems and environments.

The last thematic section of the book is **Formalisms and Methodologies for Embedded Communication Systems**. This thematic section focuses on putting the technologies and applications together and making it possible to do high abstraction level design work and verify the correctness of the designs. A very important issue to take into account is the decision making at the methodological level and what kind of effects these decisions have in the lower levels of abstraction in the design. One could see the methodological aspects as the key to successful system integration of all the different technologies needed in a complex embedded communication system. In a modern design process, it is essential to have a well-documented high abstraction level design methodology accompanied with a reliable method for verifying the correctness of the design.

The first section of this book, Technologies for Embedded Communication Systems, comprises of four chapters (chapters 2-5). Chapter 2, by Dake Liu, *et al.*, titled "*Parallel Programming and its Architectures Based on Data Access Separated Algorithm Kernels*," presents a novel master-multi-SIMD architecture and its kernel based parallel programming flow as a parallel signal processing platform, *embedded Parallel DSP processor architecture with Unique Memory Access* (ePUMA). In this approach, data accessing kernels are separated from arithmetic computing kernels in an attempt to minimize the run-time cost of data access. The technique relies on running data access in parallel with algorithm computing. The memory subsystem architecture based on SIMD improves significantly the total computing performance in the presented approach. The hardware system and the programming flow specified for it are primarily targeted to high-performance embedded parallel computing in environments requiring low power consumption and silicon cost. The performed benchmarking shows promising results for the architecture, and its scalability is successfully tested with a configuration of one master and eight SIMD cores.

Chapter 3, "*Towards Sustainable Development of Energy-Aware Systems*," by Luigia Petre and Kaisa Sere, presents an energy-aware modeling and refinement framework for systems with mobile software and hardware resources. Nowadays, more and more effort in system design must be placed on energy consumption: device power consumption needs to be minimized to reduce energy costs and to conserve resources used to produce the energy, and on the other hand, to maximize standby times and recharging intervals. The authors propose two development approaches that can be used to develop energy aware systems in a sustainable manner.

In Chapter 4, written by Arnaldo Azevedo and Ben Juurlink, and titled "*A Multidimensional Software Cache for Scratchpad-Based Systems*," an extension of software caches is discussed. Software caches are used especially in embedded systems where the hardware-controlled caches are too inflexible, for example, regarding the replacement algorithms, prefetching policy, or cache partitioning. In many streaming applications, the working set is well predictable and can be managed using software. In scratchpad-based systems it is important to prefetch data before it is needed and to overlap data transfers with computation in order to hide the memory latency. The authors introduce Multidimensional Software Cache (MDSC) with 1-4 dimensions to mimic such data structures. The software cache overhead can be reduced by using information of the application's access characteristics to reduce the number of cache accesses.

Section 1 of this book ends with Chapter 5, by David Kammler, *et al.* The chapter is titled "*Automatic Generation of Memory Interfaces for ASIPs*." In Systems-on-Chip, the design and implementation of memory interfaces that results from high-level design decisions may be cumbersome and error-prone. In the worst case the implementations have to be manually adapted to the interfaces of dedicated memories or memory controllers, slowing down the design-space exploration regarding the memory architecture of Multi-Processor System-on-Chip (MPSoC) devices. By automating the generation of the memory interfaces required, the authors streamlined the design flow and increased the reliability of the resulting interface logic. The authors introduce a new abstract and versatile description format for memory interfaces and their timing protocols. The feasibility of their approach is proven by real-life case studies including a design space exploration for a banked memory system.

Section 2 of this book is titled Mobile Communication Applications, and it consists of four chapters (6-9). Chapter 6, written by Di Wu, *et al.,* begins this section and is titled "*System Architecture for 3GPP-LTE Modem Using a Programmable Baseband Processor*." This chapter, originally published as a research article in IJERTCS vol. 1 no. 3, was one of the receivers of the highly esteemed 2010 volume year Excellence in Research Journal Awards honoring innovative research and outstanding scholarship. The new 3GPP Long-Term Evolution (LTE) standard is a hot topic among the researchers and engineers in the embedded and real-time communication systems domain. It provides an upgrade path for high performance mobile data communications, but by doing that, it poses new challenges on the implementation side. The authors address the LTE modem physical layer implementation by using a parallel processing architecture. Their architecture is an application-specific processor adopting the novel Single Instruction Multiple Tasking (SIMT) approach. In this work, the Multiple-Input Multiple-Output (MIMO) symbol detector and a parallel Turbo decoder have been implemented as configurable ASIC accelerator blocks, and the rest of the modem on the programmable processor. The authors conclude that the chosen architecture provides a feasible solution for the LTE CAT4 modem implementation.

Chapter 7 continues in the Long-Term Evolution (LTE) mobile communication domain. The chapter is titled "*Joint Uplink and Downlink Performance Profiling of LTE Protocol Processing on a Mobile Platform*" and is written by David Szczesny, *et al.* The authors address in particular the LTE protocol processing. They measure the Layer 2 protocol processing execution times and maximum data rates on a virtual ARM-based mobile phone platform. The profiling is done for uplink (UL) and downlink (DL) direction separately as well as in a joint UL and DL scenario. As a result, they identify time critical algorithms in the protocol stack and check to what extent state-of-the-art hardware platforms with a single-core processor and traditional hardware acceleration concepts are still applicable for protocol processing in LTE and beyond LTE mobile devices. Their conclusion is that such traditional platforms are incapable to achieve the highest data rates specified for LTE-Advanced downlink transmission. More sophisticated hardware accelerators for the L2 processing will be needed to supply enough computational power required in LTE and especially in next generation mobile devices.

In Chapter 8, Sergey Balandin and Michel Gillet focus on the current state-of-the-art in embedded networks research with the title "*Embedded Networks in Mobile Devices*." Mobile phones are no longer targeted solely for placing calls, but have become complex devices with a rich set of features for personal information management, multimedia processing, and multiple standard compliant communication and computer network connectivity. From the user perspective the development is intriguing as the smartphones of the near future will be able to perform many of the tasks currently executed with laptop computers; however, in terms of the device design and integration, vast challenges exist. One favored approach is to make designs as modular as possible to clarify integration and ease product differentiation. This development has lead to the need for designing embedded networks, network architectures, and specifications for interconnecting the modules of which the devices is comprised. The embedded operating environment poses challenges in the areas of power management and power consumption optimization, protocol design, quality of service, and data transport security.

Chapter 9, by Fabio Garzia, *et al*., concludes the second section of this book. In the chapter titled "*Implementation of FFT on General-Purpose Architectures for FPGA*," the authors evaluate Fast Fourier Transform (FFT) implementations on programmable and reconfigurable architectures. FFT and inverse FFT are intensively applied in modern communication systems based on Orthogonal Frequency Division Multiplexing (OFDM) as their air interface. One challenge is to implement the transforms as flexibly as possible to enable the Software-Defined Radio approach, which can be used to efficiently implement several communication receiver standards on a single device. The authors have characterized and compared such flexible implementations on multi-core and coarse-grain reconfigurable architectures implemented on a programmable logic device. Both of the parallel platforms show a considerable speed-up in comparison with the single-processor reference architecture. The speed-up is higher in the reconfigurable solution, but the MPSoC provides an easier programming interface that is completely based on C language. The authors conclude that their approach of implementing a programmable architecture on FPGA and then programming it using a high-level software language is a viable alternative to designing a dedicated hardware block with a hardware description language (HDL) and mapping it onto FPGA.

The third section of this book is On-Chip Communication, and it consists of four chapters (10-13). The section starts with chapter 10, written by Faiz-ul-Hassan, *et al.* The title of the chapter is "*Performance Analysis of On-Chip Communication Structures under Device Variability*." The increasing device variability in fine linewidth integrated circuits will affect not only the capabilities of the active computing subsystems but also on-chip communication characteristics. The authors have studied the impact of device variability on that issue concerning 13-25 nm processing technologies. They have critically examined the effect of device variability due to random dopant fluctuation (RDF) on the performance of the basic elements of on-chip communication, such as tapered buffer drivers with different tapering factor, repeaters of different sizes, and data storage registers. As a design methodology, scaling up of circuits in the critical paths can be employed to minimize the effects of device variability. However, the study results show that Network-on-Chip (NoC) link failure probability increases significantly with the increase of device variability, and it is a limiting factor in the maximum operating frequency of a synchronous link.

Chapter 11 is titled "*Schedulability Analysis for Real Time On-Chip Communication with Wormhole Switching*," written by Zheng Shi, *et al*. Network-on-Chip (NoC) architectures need to provide different levels of communication service to meet the needs of each component of the applications that are executed by the platform. For real-time application components, a scheduling strategy and an analysis approach for predicting whether all real-time packets are able to meet their timing bounds are required. The authors present an analytical method to predict an upper bound for the packet network latency in NoCs under any practical network traffic condition based on evaluating diverse inter-relationships and service attributes among traffic flows. A comparison to simulation results verifies that the communication latencies predicted using the authors' real-time analysis approach are safe and give results very close to ones obtained from simulations.

Chapter 12, written by Leonidas Tsiopoulos, et al., and titled "*Modeling Communication in Multi-Processor Systems-on-Chip using Modular Connectors*," presents an approach to formally model and verify the functionality of an asynchronous on-chip communication platform in terms of elementary communication components. The authors propose a new approach for modeling generic and hierarchical connectors for handling the complexity of on-chip communication and data flow. The communication infrastructure consists of a distribution of different channels composed into connectors, and further distribution of connectors. Their goals are to avoid overloaded bus-based architectures and to reach a distributed framework. The authors conclude that the approach is useful in modeling complex MPSoCs, and the designer can take advantage of the reusability of formally proven modular components.

The third section of this book ends with Chapter 13, written by Peter Sørensen and Jan Madsen. The chapter, titled "*Generating Process Network Communication Infrastructure for Custom Multi-Core Platforms*," focuses on the problem of automatically generating the communication infrastructure for applications modeled as process networks, targeted to custom execution platforms. The communication approach is based on abstraction layers implementing the channels and does not depend on any assumptions regarding the actual underlying platform. The approach utilizes a generic implementation that is adapted to the target custom platform built of reusable modules. The work includes an analysis method analyzing the low-level services available in the platform and for gathering the information required to synthesize access to the detected services. For the analysis, the platform is described in the service relation model formalism presented in this chapter. The approach also includes a procedure for analyzing user-provided process network mappings for feasibility, and in cases where feasibility cannot be ensured, the procedure attempts to extend the network and mapping so that it becomes feasible. The extension is carried out by incorporating additional processes and channels into the network. The approach is tested with a case study demonstrating its capabilities.

The fourth and final section of this book is titled Formalisms and Methodologies for Embedded Communication Systems, and it consists of three chapters (14-16). The section starts with Chapter 14, written by Sanna Määttä, *et al.,* and titled "*Joint Validation of Application Models and Multi-Abstraction Network-on-Chip Platforms*." The authors propose an application modeling formalism for joint validation of application and platform models during the design of multiprocessor systems-on-chip (MPSoCs). Presently, there are difficulties in capturing application constraints and utilizing them in the process of platform design space exploration. The proposed approach assists designers in doing trade-off analyses between accuracy, observability, and validation speed, and is able to carry out successive refinement of platform models at multiple abstraction levels. The executable model based approach uses a back-annotation strategy for increased accuracy of application execution during the joint validation of application and platform models. The applicability of the approach is tested and demonstrated with a case study of a single application successively mapped onto three different platform models.

Chapter 15 is titled "*Hierarchical Agent Monitored Parallel On-Chip System: A Novel Design Paradigm and Its Formal Specification*" and is written by Liang Guang, *et al.* The authors present a formal framework to model the characteristics of reconfigurable agent based systems and the monitoring operations flowing between agents at different levels of hierarchy. The authors call their approach HAMSOC, hierarchical agent monitored SoC. HAMSOC is expected to offer resilience and adaptivity to each structural level of a parallel system of any size, this way achieving dependable performance, power efficiency, and all required system features. The approach seems to be quite promising for online monitoring services in massively parallel SoCs.

The fourth section ends with the last chapter of the book, chapter 16, written by Linas Laibinis, *et al.,* and titled "*Service-Oriented Development of Fault Tolerant Communicating Systems: Refinement Approach*." The authors present the formalization of a top-down service based development method supporting correctness preserving refinement and fault tolerance management of complex communication systems. Communication systems are required to have high availability, and to achieve this, system correctness must be ensured. In this chapter the authors focus on software correctness assurance. The authors formalize and extend a top-down service-oriented method for development of communication systems called Lyra. Lyra is based on models expressed in UML2. The authors state that the proposed development approach has a high degree of automation.

Seppo Virtanen
University of Turku, Finland