Heterogeneous MP-SoC –
The Solution to Energy-Efficient Signal Processing

Tim Kogel

CoWare Inc.

tim.kogel@CoWare.com

Heinrich Meyr

CoWare Inc. and

Institute for Integrated Signal Processing Systems
Aachen University of Technology, Germany

heinrich.meyr@iss.rwth-aachen.de

ABSTRACT

To meet conflicting flexibility, performance and cost constraints of demanding signal processing applications, future designs in this domain will contain an increasing number of application specific programmable units combined with complex communication and memory infrastructures. Novel architecture trends like Application Specific Instruction-set Processors (ASIPs) as well as customized buses and Network-on-Chip based communication promise enormous potential for optimization.

However, state-of-the-art tooling and design practice is not in a shape to take advantage of this advances in computer architecture and silicon technology. Currently, EDA industry develops two diverging strategies to cope with the design complexity of such application specific, heterogeneous MP-SoC platforms. First, the *IP-driven* approach emphasizes the composition of MP-SoC platforms from configurable off-the-shelf Intellectual Property blocks. On the other hand, the *design-driven* approach strives to take design efficiency to the required level by use of system level design methodologies and IP generation tools.

In this paper, we discuss technical and economical aspects of both strategies. Based on the analysis of recent trends in computer architecture and system level design, we envision a hand-in-hand approach of signal processing platform architectures and design methodolgy to conquer the complexity crisis in emerging MP-SoC developments.

Categories and Subject Descriptors

C [**Computer Systems Organization**];
I.6 [**Simulation And Modelling**]

General Terms

Design, Performance, Measurement, Economics

Keywords

MP-SoC, Design Space Exploration, Signal Processing, Energy Efficiency, Network-on-Chip

1. INTRODUCTION

Currently, there is a fundamental argument about whether or not EDA industry will continue to exist in its traditional shape or instead refocus and merge with the IP business. This discussion is triggered by the excessive design complexity and financial risk of leading-edge System-on-Chip design, which becomes manifest in the transformation of the ASIC market into an Software centric programmable SoC platform market. As less ASIC design starts translate into less sells of 'RTL-and-below' tools, this trend threatens traditional EDA profitability. In the following, we briefly review technological and economical implications of IP-driven versus design-driven SoC development in the context of signal processing applications.

Technical Considerations

As further elaborated in the following sections, signal processing is characterized by computationally intensive algorithms, demanding bandwidth requirements and stringent cost and power dissipation constraints. Together with high flexibility requirements, the major metric for the implementation of signal processing applications is the *computational efficiency*, i.e. the ratio of performance and cost.

In this context, coarse Common-Of-The-Shelf (COTS) IP blocks are obviously disadvantageous from a technical point of view: Any COTS processor architecture is either fixed or offers limited configurability based on given architecture template, e.g. [37]. The resulting overhead leads to suboptimal computational efficiency.

On the other hand, application specific dedicated logic blocks and application specific instruction-set processors represent optimal solutions in the efficiency/flexibility space. However, current RT-level design efficiency and RT-level reusability prevents from handcrafting SoC platform.

We argue, that this mismatch between design efficiency and efficiency of the design can be resolved by novel architecture generation tools. The design entry for this kind of tools is raised in abstraction to C-level Architecture Description Languages (ADLs). Due to the restriction to a certain class of architectures (like e.g. 'processors' [3] or 'buses'), ADL based architecture generators achieve quality of results in the region of handcrafted RT-level implementations, while preserving the full expressiveness of RTL.

Apart from increasing design efficiency by an order of magnitude, the ADL based approach also fosters design reuse: representing in-house IP blocks by means of a C-level ADL greatly reduces the effort for maintenance and integration into future design projects. So from a technical point of view, this reconciles the discrepancy of IP-driven and design-driven SoC development.

Economical Considerations

Today, signal processing based products like multimedia or wireless communications fiercely compete in consumer mass markets.

The resulting small profit margins in the order of only a few percent make SoC design a highly cost sensitive endeavor.

Despite increasing mask production cost, the design effort still dominates the Non Recurrent Engineering (NRE) cost by at least one order of magnitude. Employing COTS IP offloads the block implementation and often low-level SW development from the overall design effort. However, many other cost factors, like e.g. application development, system integration, or physical design, are not affected by the employment of COTS IP. Hence the significance of IP-driven design cost savings highly depends on the actual project.

On the other hand, today's royalty based business model associated with COTS IP further reduces already narrow profit margins of system houses. Additionally, fixed or template based IP excludes the 'design' factor to achieve *product differentiation* and optimal solutions. As a separate consideration, system houses feel the necessity to own and protect their IP, and hence become increasingly reluctant to out-source design competence.

We argue that — apart from the individual profile of the company — the decisive factor for whether the IP-driven or the design-driven approach prevails is the availability of right tooling. In the remainder of this paper we further elaborate on architectural and methodical trends to support this thesis.

2. SIGNAL PROCESSING PROPERTIES

We briefly highlight trends in multimedia and wireless communications as two major signal processing application domains.

In the multimedia domain, advances in processing capabilities and multimedia algorithms together with increased user expectations fuels a constant proliferation of new multimedia standards like digital audio decoding (AC3, OGG, MP3), video decoding (MPEG2, MEPEG4, H.263, H.264, DivX, quicktime), and 3D graphic processing (DirectX 9).

Apart from the multitude and dynamics of multimedia standards, a flexible implementation platform is also mandatory to meet demanding cost constraints of converging consumer electronics devices such as the Advanced Set-Top Box (ASTB). Here the processing and communication fabrics have to be shared among the multitude of supported multimedia applications to limit implementation cost.

The wireless communication application domain is characterized by an aggressive use of digital signal processing to maximize bandwidth efficiency. Again, a multitude of ever evolving standards exists, each marking a local optimum in the multi dimensional parameter space spanned by implementation cost, mobility, energy efficiency, performance bandwidth efficiency.

The above considerations of the different embedded application domains with respect to SoC implementation can be summarized into the following set of common trends:

- New features and value added services, together with the heuristic logarithmic law of usefulness [36], lead to *exponentially increasing processing performance and communication requirements*.

- The standards become more dynamic and sophisticated and are introduced more rapidly. This calls for *high flexibility* of the SoC implementation to increase the time-in-market.

- For mobile applications as well as for cost sensitive consumer electronic devices, *energy efficiency becomes the prevailing cost factor*.

As discussed in the following section, heterogeneous Multi-Processor SoC (MP-SoC) platforms are generally believed to optimize the above mentioned conflicting performance, flexibility and energy efficiency requirements of demanding embedded applications. The heterogeneity of future SoC implementations is driven by the heterogeneity of the embedded applications itself, where each part

of the application has an inherent optimal implementation fabric. Hence, in the course of an MP-SoC platform design the *partitioning* of a specific application is a task of major importance.

A first order partitioning into a control dominated domain and a data dominated domain can be applied to any signal processing application. This first order partitioning has major influence on both the target processing and communication elements as well as on the appropriate design methodology.

Control-Plane Processing

Control-Plane Processing is characterized by moderate performance requirements, but on the other hand comprises huge amounts of functionality calling for maximum flexibility. Example control-plane processing tasks in the signal processing domain are, e.g. configuration management, or user applications.

The control plane functionality is usually developed using an architecture agnostic, software centric Integrated Design Environment (IDE) and state-of-the-art software engineering techniques like Object Oriented Programming (OOP) in UML, C++ or Java. To increase the reuse of the control plane Software across multiple MP-SoC platform generations, the Hardware dependant Software (HdS) portions are wrapped into a stack of middleware, RTOS, and driver layers [20, 24].

The huge amount of functionality and little inherent parallelism of control plane processing tasks usually prohibits the explicit specification of Task Level Parallelism (TLP). Thus, in order to gain performance the designer relys on fine grain Instruction Level Parallelism (ILP) to be extracted by a VLIW compiler or by a superscalar processor architecture.

Data-Plane Processing

Data-Plane Processing is characterized by computationally intensive data manipulations, calling for highest processing and communication performance. Additionally, rapidly evolving standards in all application domains impose increasing flexibility constraints. Example data-plane processing tasks in the signal processing domain are, e.g. audio/video de-/encoding, or UMTS/WLAN baseband processing.

The performance requirements of today's and emerging multimedia and wireless communications applications can only be reached by aggressively exploiting the abundant inherent parallelism available in the data-plane processing tasks.

- The functionality can be straight forward partitioned into a set of loosely coupled tasks with well predictable or even cyclo-stationary execution timing.

- A well confined data set is associated with a single activation of an individual task. Additionally, the data sets associated with successive activations of an individual tasks are mostly independent.

These spatial and temporal properties with respect to second order task partitioning and data dependency can already be identified during the algorithm development stage and lead to an identification of coarse grain TLP. This application inherent TLP enables the concurrent and parallel execution on MP-SoC platforms.

3. ARCHITECTURE ELEMENTS

Traditionally, signal processing systems have been largely implemented on dedicated Application Specific Integrated Circuits (ASICs) due to the immense performance requirements. Triggered by the multitude of standards and features in signal processing application domains like wireless communications and multimedia devices, a clear trend towards Software enabled implementations can be observed to provide the required flexibility. However, the demanding energy efficiency constraints of mobile applications prohibit the use of general purpose processors. Instead, the tight

efficiency requirements of versatile signal processing systems lead to application specific heterogeneous multiprocessor architectures [29].

Especially recent architectural advances with respect to both communication and computation offer a huge design space with enormous potential for optimization.

Processing Element Trends

Known as the *Energy-Flexibility tradeoff*, programmable processing elements achieve significant gains with respect to performance and energy efficiency by tailoring instruction set and micro architecture to the respective set of tasks [16, 32]. Examples are innovative architectures based on the VLIW paradigm to exploit Instruction Level Parallelism (ILP) without sacrificing energy efficiency [15] as well as SIMD based extraction of Data Level Parallelism (DLP).

Despite the increased computational performance, the effective performance is often confined by the communication architecture, since memory accesses latency does not keep pace with the processing power. General purpose processors resolve the memory access bottleneck by using sophisticated cache and memory hierarchies. Unfortunately this approach is often not applicable for embedded applications due to the poor memory locality of stream driven and packet based data processing.

Instead, processer architectures are increasingly equipped with hardware supported Multi-Threading (HW-MT) [38] to perform task switches with virtually no performance overhead. By that, the application inherent TLP is exploited with the purpose of hiding memory latency, which effectively leads to a significant increase in the processor utilization. This technique is already widely employed in the network processor domain [30] but recently finds its way into advanced multimedia [25] and signal processing platforms [9].

Besides the immediate benefit of increased utilization, HW-MT can be considered as a lean operating system implemented in hardware to efficiently share the processing resources among multiple concurrent tasks. In analogy with today's software operating systems (SW-OS), the HW-MT concept bears the potential to bring a disciplined management of processing resources to the data processing domain. From the perspective of the functional tasks, this 'processing management' introduces a virtualization of the computational resources.

Communication Architecture Trends

Today's predominant shared bus paradigm as inherited from the PCB era constitutes the major power and performance bottleneck. In response to this problem, the global communication is envisioned to be handled by full-scale Network-on-Chip (NoC) architectures [21]. Dedicated on-chip networks enable the use of physically optimized transmission channels to address power, reliability and performance issues [19, 1].

Apart from resolving the physical issues, Network-on-Chip architectures also address the *functional* aspects of on-chip communication. So far, the dynamic priority based arbitration scheme of shared busses creates a mutual dependency between all components connected to the bus. Due to this lack of traffic management capabilities every change in the traffic requirements of the application requires a re-design of the bus architecture. Instead, NoC architectures take advantage of sophisticated networking algorithms to provide elaborated traffic-management capabilities. By that, the ad-hoc communication mapping is replaced with a disciplined allocation of the required communication services and the on-chip network takes care to provide the required resources.

The system architecture perspective, this *virtualization* of the communication resources into a set of offered communication services decouples the mapping problem for communication and com-putation. The price for the physical and functional benefits of NoC based communication is a significant penalty in terms of chip area as well as transfer latency. In the light of the latency issue caused by NoC architectures, the importance of memory hiding in the processing elements described above is likely to increase in the future.

Taking the above considerations together, future SoCs can be considered as NoC enabled multi-processor architectures. The on-chip communication backbone connects a large number of heterogeneous processing clusters and storage elements. Individual processing clusters consist of one or few application specific programmable kernels together with tightly coupled instruction and data memories as well as local peripherals.

4. MP-SOC DESIGN STRATEGIES

According to the *multi-level SoC design* approach proposed by Magarshak and Paulin [28], the overall MP-SoC design process can be separated into four (largely) orthogonal phases. Except for the low-level semiconductor technology phase, we examine the methodology and tooling related aspects of the individual phases.

Functional Phase

The functional phase is performed by application specialists and is completely agnostic to architectural considerations. This phase includes the embedded SW development of the control-plane portion of the application as well as data-plane algorithm development.

The functional phase employs a multitude of different 'vertical' Models of Computation, each of which address the abstract modeling of a particular application domain. In particular the signal processing domain relies on Data-Flow models, which allow for efficient capture and analysis of signal processing algorithms. Corresponding tools exploit the formal properties data-flow models for the purpose of highly efficient simulation engines as well as automatic model sanity checks.

High-Level IP Creation Phase

The IP creation phase deals with the design of processing elements (RISC, DSP, MCU, ASIPs), on-chip interconnect technologies (buses, NoC), domain specific standard I/O (PCI-variants, SPIx variants, HyperTransport, I2C, FireWire, QDR, etc.), as well as the creation of well defined ASIC IP blocks e.g. an MPEG4 video codec).

In the signal processing domain, high level IP creation is not orthogonal to the considered application, but requires to tailor the IP block towards the respective portion of application. Especially the development of an Application Specific Instruction-set Processor (ASIP) requires a joint consideration of the computer architecture and the corresponding task.

As already discussed in the introduction, the high-level IP creation phase bears an enormous potential to significantly increase design productivity by using ADL driven architecture generators. This also demonstrates, that there is no one-to-one correspondence between 'design phase' and 'abstraction level'. For example, the IP creation phases covers both ADL-level as well as RT-level abstraction layers.

MP-SoC Platform Phase

The MP-SoC platform phase covers the system-architecture specification by integration of high level IP blocks, but also the spatial and temporal mapping of the application to the MP-SoC platform. Hence, this phase is concerned with the full functional and architectural complexity of MP-SoC platforms.

The MP-SoC platform phase is concerned with the system architecture specification as well as the application mapping. Therefore, abstraction concepts on this level have to support the joint consideration of the application and architecture. On the other hand, the

high level of detail inherent to Register Transfer Level (RTL) implementation models prohibits the investigation and optimization across heterogeneous communication and processing elements.

Transaction Level Modeling (TLM) [33, 12] is generally considered as the emerging abstraction level to cope with the requirements of the MP-SoC platform phase. The characteristic property of TLM is that the pin-level communication interface of RTL models is replaced by a set interface methods. In theory, this communication mechanism is provided by all actor-oriented specification languages like SystemC [31], SpecC [13] or the Metropolis Meta Model [14].

The well established cycle-level TLM based design methodology promotes the early creation of an executable prototype of the MP-SoC platform for hardware dependant software development as well as tuning the communication and memory architecture to the application requirements. The underlaying HW/SW Co-simulation technology combines flexible just-in-time compiled processor simulation [4] with cycle accurate TLM bus and HW models [5]. This approach finds great demand in the SoC design community and is well supported by SystemC based standardized IP models [2, 10], and ESL tools like CoWare ConvergenSC [11]. This kind of tools also provide capabilities for hybrid simulation of cycle accurate TLM models together with RTL implementation models executed by an HDL simulator.

Despite the undisputed success of cycle-level TLM based co-simulation technology for early platform prototyping [26], this approach does not solve all the challenges of the MP-SoC design phase: first, the effort to create a cycle-accurate prototype of the complete platform is still too high to allow for the investigation of a large number of architecture and application mapping alternatives. Second, the simulation speed in the order of 100k cycles per second is neither sufficient for embedded SW development nor for sensitivity analysis of large sets of design parameters.

Evolution of HW/SW Co-Design

In the context of heterogeneous signal processing platforms, the classical vertical partitioning approach to HW/SW Co-design, where the performance critical parts are implemented as dedicated HW blocks and the rest is executed in SW, is no longer applicable. Instead HW/SW Co-design can be seen as a multi-dimensional horizontal mapping problem of an application running on a heterogeneous multiprocessor platform. The partitioning of the application into a set of loosely coupled functional blocks and the extraction of task level parallelism (TLP) is mostly straight forward for typical signal processing applications and can be immediately derived from the algorithmic block diagram.

However, the spatial and temporal mapping of the functional tasks to processing elements as well as the mapping of the inter-task data exchange to a communication architecture while meeting performance and cost requirements is an unresolved challenge.

As described in section 3, the key concept to cope with the resulting design complexity is to achieve a virtualization of the architectural elements, such that they can be *allocated* by the system architect in a deterministic way [7]. As discussed above, this virtualization is provided by the novel NoC approach for the communication part as well as by SW and HW operating systems for the control and data processing respectively. This divide-and-conquer oriented design paradigm enables individual optimization of the architectural elements to take full advantage of recent developments in computer architecture and NoC enabled communication. The price for this benefits with respect to both design efficiency and architectural efficiency is merely a penalty in terms of chip area, which is generally considered to be of constantly decreasing importance.

In this context HW/SW-Co-design of a given embedded application is defined to a) architect a heterogeneous MP-SoC platform and b) allocate the architectural resources for the execution of the application. Note, that architecture virtualization resolves the mutual dependencies in the mapping process, but the *trade-offs* in the design space still require a joint consideration of application and architecture as well as communication and communication. For example the latency of a more complex on-chip network can be compensated by either introducing memory hierarchy or employing hardware multi-threaded processor kernels. Obviously, the resulting design space is virtually infinite and the architecting and the mapping phase cannot be considered independently without sacrificing quality of results.

5. MP-SOC EXPLORATION FRAMEWORK

We argue, that the virtualization of architectural resources for the first time really enables architectural considerations on top of cycle-level accuracy. Application mapping and architecture exploration can be reduced to the allocation of timing budgets on processing and communication resources.

We envision a new Intermediate Representation (IR) in the design of application specific MP-SoC platforms to perform 'large-scale' design space exploration and to reason about the application partitioning. This MPSoC-IR enables a *virtual mapping* of the considered application onto the anticipated platform architecture.

After a classification of the selected abstraction level, we will highlight major aspects of the underlying timing model, which enables an abstract and yet sufficiently accurate modeling of the anticipated architecture. This timing model is implemented by a modular simulation framework for rapid design space exploration of Network-on-Chip enabled heterogeneous MP-SoC platforms.

Packet Level TLM

For the conceptualization of heterogeneous signal processing systems, cycle-level TLM is still too detailed to explore large design spaces. Instead, our modeling framework is based on a *packet-level TLM paradigm*, where the considered data granularity are sets of functionally associated data, which are combined into Abstract Data Types (ADTs).

This higher level of abstraction is much closer to the initial application model, so the modeling efficiency as well as the simulation speed are again significantly improved compared to cycle-accurate TLM. The key aspect of our approach is that the underlying timing model outlined below is sufficiently accurate to investigate the performance impact of the anticipated MP-SoC architecture executing the application.

Unified Timing Model

The conceived timing model can be coarsely separated into the following aspects:

- A generic synchronization interface defines a concise set of communication primitives, which in principle follow the Open Core Protocol (OCP) semantics [27] and are not biased towards any specific communication architecture. Additionally the primitives are shaped to achieve reasonable timing accuracy at the highly abstract packet-level TLM layer.

- The communication timing model captures the impact on performance of the interconnection architecture. The communication timing model supports the full spectrum of available and proposed communication architectures ranging from today's shared buses to the emerging NoC paradigm [23, 35].

- The processing delay annotation virtually maps individual application tasks to the intended processing engines [34]. The resulting impact on performance is captured by calculating the timing of the external events, which are exposed by the generic communication interface.

- The concept of virtual processing units models the notion of shared computational resources. This covers both software operating systems as well as hardware multi-threading.

Exploration Framework

We have implemented the timing model outlined above by means of a versatile exploration framework as depicted in figure 1 below. One key aspect for efficient design space exploration is a declarative specification mechanism, i.e. the following aspects of the MP-SoC architecture are defined by a set of configuration files:

- configuration of the timing model

- number of available processors and number of supported parallel HW threads per processor

- mapping of the application task to processors and HW threads

- instantiation, parameterization and interconnection of the communication nodes

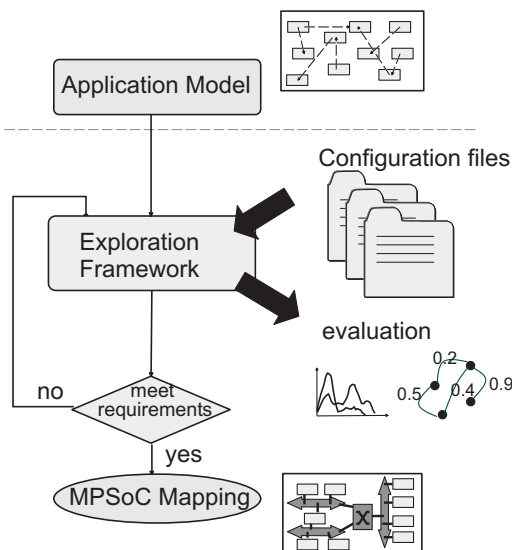- instantiation and address mapping of the memory architecture

Figure 1: MP-SoC Exploration Framework

The simulation results like latency, delay and utilization of processing elements and communication links are stored in a data base and compiled into a set aggregated histograms and performance graphs. Based on this results, the system architect can detect bottlenecks or poor utilization in the system and decide on further optimizations of the architecture model and/or the application mapping respectively.

Verification Aspects

Besides the systematic refinement flow, the value of the proposed MPSoC-IR also reaches into the verification of subsequent implementation steps. The information contained in the MPSoC-IR serves as a *predictor for the verification flow* [8] to verify the functional correctness of error-prone implementation models as well as to review the estimated timing properties.

In that our approach takes the synchronous design principle, which enabled the RTL based modeling, to the next level: transistor-level timing is neglected during the specification at the register transfer level. After the physical design, the clock synchronization points

enable the verification of the assumed timing budgets and physical gate- and wire-delays.

In close analogy, the packet-level events serve as synchronization points, which can be recovered at the cycle-accurate implementation level. By that, the timing annotations in the MPSoC-IR can be automatically translated into assertions.

Related Work

Multiple aspects in system-level methodologies and tooling are subject to ongoing research, e.g.

- formalized specification enabling links to synthesis and formal verification [14],

- component based design through automated instantiation and integration of predefined IP blocks [22],

- mapping of algorithmic data-flow models to abstract architecture models [6]

- abstract RTOS modeling [18, 17]

The acid test for research on system level design has always been the incorporation into actual design practice. In any case companies are now forced to evaluate and adopt new design methodologies and tooling to cope with MP-SoC complexity crisis.

6. CONCLUSION

This paper analyzes the convergence of EDA companies and IP providers in the context of trends in the signal processing domain and recent architectural developments. We argue, that neither traditional 'RTL-and-below' EDA nor traditional IP products and business models are fit for the emerging MP-SoC design era: EDA suffers from declining design starts, increasing embedded software content, and a complexity beyond the capabilities of the RTL design entry. On the other hand, platform architectures entirely based on fixed or configurable commodity IP fail to meet conflicting performance, flexibility and energy efficiency constraints of versatile signal processing products. Additionally, system houses more and more refuse to pay royalties for COTS IP.

We envision a convergence of EDA and IP products on the basis of a two-phase system level design flow.

First, new architectural concepts like hardware operating systems and on-chip networks achieve a separation of processing and communication services from the architectural resources. This *virtualization* enables the investigation of application partitioning and resource allocation at an abstraction level on top of cycle accuracy.

Subsequent to the definition of the MP-SoC platform architecture, a new generation of ADL based *IP generation tools* provides the required design productivity and implementation quality to develop application specific processing elements and communication fabrics.

We argue, that semiconductor and system companies will endorse this convergence of EDA and IP, which provides 'MP-So-level' design productivity on the basis on an EDA-like subscription business model. By that they are back in the state to develop highly differentiated products on the basis of in-house IP.

7. ACKNOWLEDGEMENTS

8. REFERENCES

[1] Arteris Unveils Strategy, Technology for enabling Network on Chip (NoC) Design. Press Release, March 2003.

[2] A. Cochrane, C. Lennard, K. Topping, S. Klostermann, N. Weyrich, K. Ahluwalia. *AMBA AHB Cycle Level Interface (AHB CLI) Specification*, 2003.

[3] A. Hofmann, H. Meyr, R. Leupers. *Architecture Exploration for Embedded Processors with LISA*. Kluwer Academic Publishers, 2002. ISBN 1-4020-7338-0.

[4] A. Nohl, G. Braun, A. Hoffmann, O. Schliebusch, R. Leupers, H. Meyr. A Universal Technique for Fast and Flexible Instruction-Set Architecture Simulation. In *Proceedings of the Design Automation Conference (DAC)*, 2002.

[5] A. Wieferink, T. Kogel, R. Leupers, G. Ascheid, H. Meyr, G. Braun, A. Nohl. A System Level Processor/Communication Co-Exploration Methodology for Multi-Processor System-on-Chip Platforms. In *"Proc. Int. Conf. on Design, Automation and Test in Europe(DATE)"*, Februry 2004.

[6] A.D. Pimentel, C. Erbas. An IDF based Trace Transformation Method for Communication Refinement. In *Proceedings of the Design Automation Conference (DAC)*, June 2003.

[7] T. Agerwala. Systems Trends and their Impact on Future Microprocessor Design. Keynote of 35th Annual International Symposium on Microarchitecture, November 2002.

[8] B. Bailey. Property Based Verification for SoC. Int. Symp. on System-on-Chip (SoC), November 2003. Invited Talk.

[9] C. J. Glossner, T. Raja, E. Hokenek, M. Moudgill. A Multithreaded Processor Architecture for SDR. *Proceedings of the Korean Institute of Communication Sciences*, 19(11):70–85, November 2002.

[10] J. Connel and B. Johnson. Early hardware/software integration using systemc 2.0, 2002.

[11] ConvergenSC. *CoWare, http://www.coware.com*.

[12] D. Gajski. Transaction Level Modeling. In *"Proc. of the IEEE/ACM/IFIP Int. Conference on Hardware/Software Codesign and System Synthesis"*, 2003.

[13] D. Gajski, J. Zhu, R. Dömer, A.Gerstlauer, S. Zhao. *SpecC: Specification Language and Methodology*. Kluwer Academic Publishers, 2000.

[14] F. Balarin, Y. Watanabe, H. Hsieh, L. Lavagno, C. Passerone, A. Sangiovanni-Vincentelli. Metropolis: An integrated electronic system design environment. *IEEE Computer*, 36(4):45–52, April 2003.

[15] G. Fettweis. Embedded vector signal processor design. In *"Proc. Int. Workshop on Systems, Architecturs, Modeling and Simulation(SAMOS)"*, July 2003.

[16] H. Blume, H. Hübert, H. T. Feldkämper, T. G. Noll. Model-Based Exploration of the Design Space for Heterogeneous Systems on Chip. In *Proceedings of the IEEE Conference on Application Specific Architectures and Processors*, 2002.

[17] H. Yu, A. Gerstlauer, D. Gajski. RTOS Scheduling in Transaction Level Models. In *"Proc. of the IEEE/ACM/IFIP Int. Conference on Hardware/Software Codesign and System Synthesis"*, 2003.

[18] M. G. J. Madsen, K. Virk. Abstract RTOS modelling for multiprocessor system-on-chip. In *International Symposium on System-on-Chip*, pages 147–150. IEEE, nov 2003.

[19] K. Goossens, J. van Meerbergen, A. Peters, P. Wielage. Networks on Silicon: Combining Best-Effort and Guaranteed Services. In *"Proc. Int. Conf. on Design, Automation and Test in Europe(DATE)"*, 2002.

[20] K. Keutzer, S. Malik, A.R. Newton, J.M. Rabaey, A. Sangiovanni-Vincentelli. System-level design: Orthogonalization of concerns and platform-based design. *IEEE Transactions on Computer-Aided Desig of Integrated Circuits and Systems*, 19(12):1523–1543, December 2000.

[21] L. Benini, G. De Micheli. Networks on Chips: A New SoC Paradigm. *IEEE Computer*, pages 70–78, January 2002.

[22] M.-A. Dziri, W. Cesrio, F.R. Wagner, A.A. Jerraya. Unified Component Integration Flow for Multi-Processor SoC Design and Validation. In *"Proc. Int. Conf. on Design, Automation and Test in Europe(DATE)"*, 2004.

[23] M, Ariyamparambath, D. Bussaglia, B. Reinkemeier, T. Kogel, T. Kempf. A Highly Efficient Modeling Style for Heterogeneous Bus Architectures. In *"Proc. IEEE Int. Symp. on System-on-Chip (SoC)"*, November 2003.

[24] M. Grammatikakis, M. Coppola, F. Sensini. *Software for Multiprocessor Networks on Chip*, chapter 14, pages 281–303. Kluwer Academic Publishers, 2003.

[25] M.J. Rutten, J.T.J. van Eijndhoven, E.G.T. Jaspers, P. van der Wolf, O.P. Gangwal, A. Timmer, E.-J.D. Pol. A Heterogeneous Multiprocessor Architecture for Flexible Media Processing. *IEEE Design & Test of Computers*, 19(5):39–50, July-August 2002.

[26] O. Ogawa, K. Shinohara, Y. Watanabe, H. Niizuma, T. Sasaki, Y. Takai, S. Bayon de Noyer and P. Chauvet. A Practical Approach for Bus Architecture Optimization at Transaction Level. In *"Proc. Designers' Forum, Int. Conf. on Design, Automation and Test in Europe(DATE)"*, 2003.

[27] Open Core Protocol International Partnership (OCP-IP). *OCP datasheet, http://www.ocpip.org*.

[28] P. Magarshack, P. Paulin. System-on-chip Beyond the Nanometer Wall. In *Proceedings of the Design Automation Conference (DAC)*, 2003.

[29] R. Subramanian, U. Jha, J. Medlock, C. Woodthorpe, K. Rieken. Novel Application-Specific Signal Processing Architectures for Wideband CDMA and TDMA Applications. In *Proc. of the IEEE Vehicular Technology Conference (VTC)*, 2000.

[30] S. Lakshmanamurthy, K.-Y. Liu, Y. Pun, L. Huston, U. Naik. Network Processor Performance Analysis Methodology. *Intel Technology Journal*, 6(3), Aug. 2002.

[31] SystemC initiative. *http://www.systemc.org*.

[32] T. Gloekler, H. Meyr. *Design of Energy-Efficient Application-Specific Instruction Set Processors*. Kluwer Academic Publishers, 2004. ISBN 1-4020-7730-0.

[33] T. Grötker, S. Liao, G. Martin, S. Swan. *System Design with SystemC*. Kluwer Academic Publishers, 2002.

[34] T. Kogel, A. Wieferink, R. Leupers, Gerd Ascheid, H. Meyr, D. Bussaglia, M. Ariyamparambath. Virtual Architecture Mapping: A SystemC based Methodology for Architectural Exploration of System-on-Chip Designs. In *"Proc. Int. Workshop on Systems, Architecturs, Modeling and Simulation(SAMOS)"*, July 2003.

[35] T. Kogel, M. Doerper, A. Wieferink, R. Leupers, G. Ascheid, H. Meyr, and S. Goossens. A Modular Simulation Framework for Architectural Exploration of On-Chip Interconnection Networks. In *CODES+ISSS*, October 2003.

[36] T.A.C.M. Claasen. High Speed: Not the Only Way to Exploit the Intrinsic Computational Power of Silicon. In *In Proceedings of the International Solid-State Circuits Conference*, 1999.

[37] Tensilica. *http://www.tensilica.com*.

[38] D. M. Tullsen, S. Eggers, and H. M. Levy. Simultaneous multithreading: Maximizing on-chip parallelism. In *Proceedings of the22th Annual International Symposium on Computer Architecture*, 1995.