MAPPING WAVEFORMS TO MOBILE PARALLEL PROCESSOR ARCHITECTURES

Cyprian Grassmann (Infineon Technologies AG, Munich, Germany; Cyprian.grassmann@infineon.com); Alfonso Troya (Infineon Technologies AG, Munich, Germany; alfonso.troya@infineon.com); Mirko Sauermann (Infineon Technologies AG, Munich, Germany; mirko.sauermann@infineon.com); Mathias Richter (Siemens AG, Germany; mathias.richter@siemens.com); Ulrich Ramacher(Infineon Technologies AG, Munich, Germany; ulrich.ramacher@infineon.com)

ABSTRACT

Portable software defined radio solutions for the consumer market will be based on mobile devices employing multiple programmable processing resources, as low power consumption needs to be maintained for an acceptable battery life. Therefore the designer needs to orchestrate the given system functions of the waveform efficiently onto the distributed processing resources. For the automation or at least a substantial machine support of that process an explicit and platform independent representation of parallelism is a prerequisite. This article is focusing on the basic concepts to model waveforms for parallel hardware systems not directly related to the software communication architecture (SCA) [1]. Nevertheless we discuss the implications for the SCA along with the changes we expect for the value chain.

1. INTRODUCTION

Performance requirements of SDR solutions require parallel processor architectures if power constraints of mobile devices have to be met. As waveforms need to be split into pieces of concurrently executing programs, the programming of these architectures is usually difficult and error-prone. A programming environment, which is completely compiler based would be desirable, but is currently out of reach at least if different kinds of parallelism have to be exploited efficiently. Additionally most of the waveform descriptions - including the SCA based descriptions - are lacking important information to allow for design decisions, which need to be taken during the mapping of the waveform to a parallel architecture. Taking the perspective of the "mapper" we derive the requirements for both the description of the waveform, which we call the system function model and the description of the architecture, which we call the system architecture model. Based on these descriptions we show approaches to the automation of the mapping process and discuss their limitations.

A substantial support of this mapping process with a separation of the functional and the architectural system models is of importance for two reasons: On the one hand system integrators requesting for higher level interfaces to the platform providers, still demanding for customized system functions – on the other hand platform providers need to be able to exploit sophisticated architectural features to meet the strong power and area budgets.

Therefore not only technical support, but also a new business model is needed for parallel mobile SDR platforms.

A functional system model of WLAN 802.11 and its mapping to the Infineon SDR baseband platform serves as a concrete example for the proposed approach.

2. MAPPING PROCESS

For the design space exploration phase we propose the mapping process shown in figure 1.

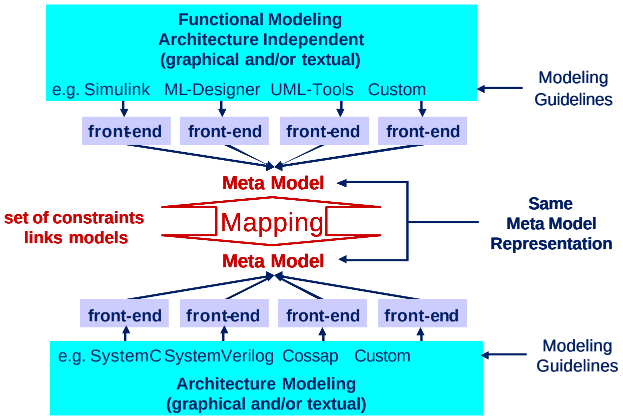

Figure 1: mapping process for design space exploration

Meta-models of the system function and the system architecture are represented by the same meta-model. This approach has basically two advantages: The functional and the architecture model can be captured with the modeling toolset of choice and the common meta-model for function and architecture is at least for a tool supported mapping process a prerequisite. The mapping itself needs to be expressed by a set of constraints, like .e.g. the logic constraint language used in Metropolis [2]. Strictly following this approach leads to a functional model which is independent from the target platform, as architecture specific transformations or adaptations are only captured by the set of constraints – the mapping itself. This approach is in accordance with the Model Driven Architecture (MDA), as the MDA specifies a Platform Independent Model (PIM) and a Platform Specific Model (PSM) and mappings which are transformations of the PIM into a PSM. The functional system model is a PIM, but the model of the system architecture is not the PSM, but rather provides essential information to derive the PSM from the PIM. In fact the model of the system architecture is an image of the target system and allows the execution of the PSM to gather detailed profiling data, before any silicon is produced. Therefore figure 1 shows the situation for a design space exploration, where the PIM is repeatedly mapped to the virtual prototype of the hardware for an iterative refinement of the functional and architectural system model.

As soon as the target architecture converged throughout the design space exploration, the meta-model of the system architecture serves to optimize the mapping of the system function to the system architecture. During that phase the system architecture stays unchanged but still multiple mapping alternatives have to be considered – especially for parallel architectures.

It is obvious that modeling guidelines need to be specified for each toolset, because the toolset may provide an expressiveness which can’t be mapped to the meta-model or workarounds have to be specified to circumvent semantic limitations of the toolset.

Converters – or simply parsers - drive the meta-model from the toolset specific model. Today we are examining Simulink and MLDesigner for the functional model, using XML as the intermediate representation for the meta-model. The architecture model is captured using SystemC, which allows to build cycle accurate models and which results in an executable model of the architecture.

3. FUNCTIONAL MODEL - WAVEFORM

The waveform is captured on a functional level. It is decomposed into functional blocks like filter, FFT, demodulation, etc.. Usually there is an intuitive decomposition of a waveform into functional blocks. For the rest of the design process these functional blocks are considered to be black boxes, specified by their interfaces computational load and timing. Therefore during a partitioning and mapping of the functional model to the architecture, only the computational load, timing and the dependencies between the functional blocks have to be considered. It is important to point out, that the visual modeling of the functional system model is not only convenient and intuitive, but also essential to the mapping onto parallel architectures as dependencies between functional modules have to be modeled explicitly. If one would use sequential c-code to model the system function, it is obvious that dependencies between functional modules have to be recovered by sophisticated analyses like “point-to analyses” and it is very likely that it can’t be done completely automatically. Hence the decomposition on the functional level is crucial and already critical for the whole design process. Anyway this limitation is reasonable, as the modification of the functional blocks (decomposition into multiple smaller blocks or fusion to larger blocks) usually requires the intervention of an engineer, as algorithms need to be adapted. The code or hardware which is needed to hook the functional blocks together and to orchestrate their execution can be generated without intervention of an engineer, as long as the functional model provides sufficient information for that generation process. A sufficient set of information about the system function goes beyond what is covered by a waveform description which is conform with the SCA, as e.g. timing requirements are not captured. Languages like WDL [3] try to solve this problem, by incorporation of different, existing languages. Unfortunately the tool support for the WDL, which supposes to be based on Ptolemy [4] from the UC Berkeley, is still very poor, so that an evaluation of that concept is too cumbersome. We started to model the WLAN 802.11b waveform with Simulink and in parallel with ML-Designer. Unfortunately none of the tools fulfills all the requirements mentioned above, so that we were forced to implement workarounds to overcome the semantic limitations, usually caused by the underlying scheduler of the tool.

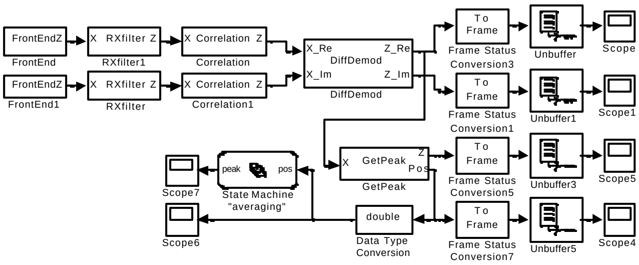

Figure 2: part of the WLAN 802.11b Simulink model

Finally the design data captured with either of these tools is converted to a XML-representation, accompanied by files, which provide a generic encapsulation of the behavior. Figure 3 shows how we encapsulated the functionality inside a pl-file, which basically contains the behavior of the functional module in plain c-code. Additionally the pl-file contains information about the interfaces and parameters to the module. This information allows for generating different representations from the same source. Up to now we have the generation of S-functions for a Simulink model and the generation of plain c-functions in place. The generation of ML-Designer modules is straightforward, as the pl-file fomat is native to the ML-Designer and we modified it only slightly for our purposes.

This format provides an excellent basis for a further processing. Alternative tools for the design capture are currently under evaluation and UML based modeling environments are already identified as promising candidates. A good support for a XML-based data exchange and a highly customizable code generation are key factors for a tool to be successfully integrated in the proposed design flow.

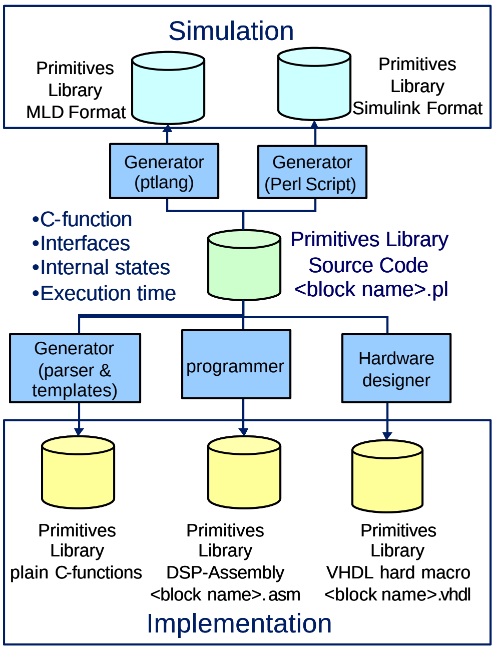
Simulation
Primitives Library MLD Format
Primitives Library Simulink Format
Generator (ptlang)
Generator (Perl Script)
•C-function
•Interfaces
•Internal states
•Execution time
Primitives Library Source Code <block name>.pl
Generator (parser & templates)
programmer
Hardware designer
Primitives Library plain C-functions
Primitives Library DSP-Assembly <block name>.asm
Primitives Library VHDL hard macro <block name>.vhdl
Implementation

Figure 3: generic representation of functionality by pl-file

4. ARCHITECTURE MODEL – TARGET HARDWARE

Comparable to nowadays models of the functional level, the models on the architectural level usually also suffer from completeness. Only complete system models allow the collection of relevant performance data and to guide the system development process. Therefore the architectural model has to be a virtual prototype which is an image of the real hardware to be built. Looking for a complete model of a system the designer still has to carefully choose the trade-offs between simulation speed and the appropriate degree of detail.

We chose to model the baseband architecture [5] shown in figure 4 as a SystemC model integrating external simulators for processor cores with according SystemC wrappers. The resulting virtual prototype provides a cycle accurate simulation of the complete system with 21 processors, bus system, memories and peripherals. The conversion of the SystemC model into a suitable meta-model is still ongoing. Currently a specification of the hardware parameters, which are essential to the mapping process, is extracted manually.

Given the hardware parameters, the PSM is derived from the XML representation of the functional model. The PSM itself is again a XML-file using an extended schema of the XML-file of the functional model, as additional information is captured to allow for a code generation for the target architecture.

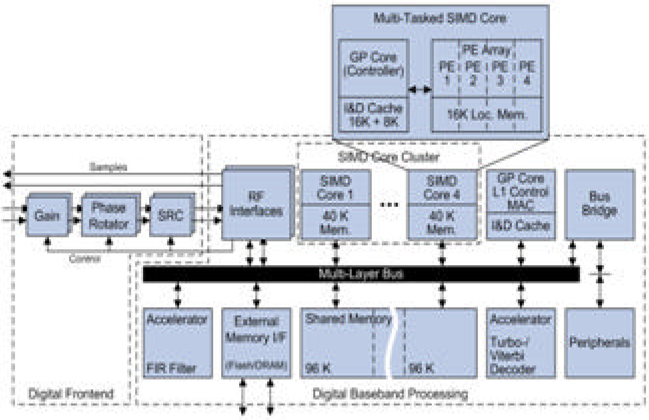

Figure 4: Multi SIMD Core (MuSIC) baseband architecture

The assignment of functions to processor cores, buffer sizes and strategies and the according synchronization dependencies are captured within the PSM. Given this set of information the code generation process is simplified because the mapping knowledge is represented explicitly rather than being an implicit part of the code generator itself. Therefore the PSM provides the separation of concerns and allows for a relatively dump code generation, but requires a sophisticated transformation from the PIM to the PSM.

5. PARTITIONING AND CODE GENERATION

As a first step towards an automated partitioning of the system function, we generate a task graph from the functional block diagram, assigning costs to its nodes and vertexes. The cost for a node is derived from a profiling, which is collected for each function and the cost for a vertex is derived from the interface descriptions of the pl-files and the overall throughput requirements of the system. The graph is then partitioned and scheduled onto the given architecture by a complete search [6].

The complete search leads to an optimal mapping but is pretty time consuming. Nevertheless the complete search is applicable to standards like WLAN 802.11b and WCDMA within a reasonable time. We are also investigating branch and bound approaches to reduce the run-time for the automated partitioning and scheduling without degrading the result to significantly.

We are using Eclipse as framework [7] for the implementation. Parsers for the different file formats and the graph partitioning and scheduling are implemented as plug-ins to the Eclipse framework.

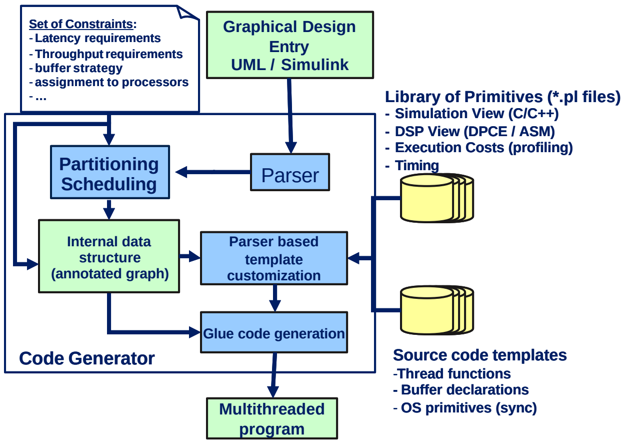

Figure 5: partitioning, scheduling and code generation process

6. BUSINESS IMPLICATION

The proposed design process allows for establishing an interface between the integrator and the platform provider on a higher level of abstraction. In the ideal case the integrator can use models of radio standards (waveforms) to arrange a model of a multi-standard modem and use this model to communicate the requirements to the platform provider. Within this scenario the models of the radio standards can be provided by the platform manufacturer, a third party or the system integrator itself, as long as theses models are specified in a standardized way.

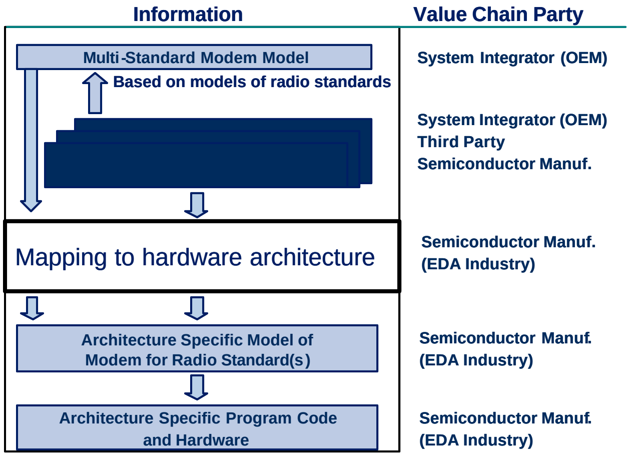

Figure 6: value chain with model based modem design

A standardized description should especially incorporate timing constraints and use case scenarios, which reflect the interaction of the system with the environment, as some of the internal parameters of the system can only be determined based on external parameters and stimuli. As the description should provide all information, which is needed to make partitioning and scheduling decisions during the mapping of the system function to the system architecture, the standardization needs to be driven by the requirements of the mapping process and the capabilities and limitations of the mapping techniques and tools.

To help to get SCA accepted for commercial applications, the SCA needs to standardize the way in which the mentioned system requirements and the system usage have to be captured. Additionally the coupling of SCA with CORBA as middleware is also a reason for a reservation of the commercial community. At least the mobile handset market, with its tight area and power constraints, currently prohibits the implementation of a CORBA compliant software layer.

With the software radio specification from the OMG [8] a first step towards a commercialization of the SCA is taken, as a PIM was specified based on an UML profile for SWRadio. The PIM is a prerequisite for the mapping to non-CORBA platforms. According mappings are still required to prove the concept. Additionally the PIM has to be extended to capture the mentioned timing constraints and use case scenarios.

7. CONCLUSION

We showed that the mapping of waveforms to mobile parallel architectures requires the modeling of the system function, at least if parts of the design flow should be executed automatically. Moreover these models can be used to capture the requirements of the system integrator and serve as interface between the system integrator and the platform provider. This interface can only be established if the integrator and the platform provider agree on a mandatory standard for the functional system description, including timing constraints and use case scenarios, as described in this article.

8. REFERENCES

[1] Joint Tactical Radio System (JTRS) Joint Program Office, “Software Communications Architecture Specification - JTRS-5000, SCA V3.0”, *Standard*, JTRS, August 2004, http://jtrs.army.mil/documents/sca_documents/V3.0/SCARelease3.0.pdf

[2] F. Balarin and Y. Watanabe and H. Hsieh and L. Lavagno and C. Paserone and A. Sangiovanni-Vincentelli, “Metropolis: an integrated electronic system design environment”, IEEE Computer, vol 36, no. 4, pp 45-52, apr 2003.

[3] E. D. Willink, “The Waveform Description Language”, in *Software Defined Radio*, pp. 365-397, Wiley & Sons, Sussex, England 2002

[4] Edward A. Lee, "Overview of the Ptolemy Project", Technical Memorandum No. UCB/ERL M03/25, University of California, Berkeley, CA, 94720, USA, July 2, 2003.

[5] H.-M. Bluethgen and C. Grassmann and W. Raab and U. Ramacher, "A Programmable Baseband Platform for Software-Defined Radio", SDR'04 Technical Conference, Phoenix, November 2004

[6] M. Nunkesser, "Mapping Task Graphs to Digital Signal Processors", Technical Report TR 460, Department of Computer Science, ETH Zürich October 2004

[7] Eclipse Platfrom Technical overview, http://www.eclipse.org/whitepapers/eclipse-overview.pdf

[8] OMG, "PIM and PSM for Software Radio Components", *2nd FTF Convenience Document*, dtc/2005-08-03, OMG, August 2005, http://www.omg.org/cgi-bin/apps/do_doc?dtc/05-08-03.pdf

[9] Morgan Björkander and Cris Kobryn, "Architecting Systems with UML 2.0", IEEE Software, July/August 2003

[10] SysML Specification v. 0.85 Draft, http://www.sysml.org/artifacts/spec/SysML-v0.85R1-PDF-041011.zip

[11] W. Raab et al., "A Development System for Heterogeneous Multiprocessor Architectures", IEEE Workshop on Heterogeneous Reconfigurable Systems on Chip (SoC), Hamburg, 2002.

[12] S. Niemann, "Functional Modeling With UML", *Technical Report*, I-Logix, 2004

[13] J. Long, "Relationshiops between Common Graphical Representations in System Engineering", *Technical Report*, Vitech Corp., 2002