Published in the proceedings of Wireless World Research Forum, Zurich, 2003

1 Wireless World Research Forum (WWRF)

A Consistent Design Methodology to Meet SDR Challenges

M. Holzer, P. Belanović, and M. Rupp
Vienna University of Technology
Institute for Communications and RF Engineering
Christian Doppler Pilot Laboratory for Design Methodology of Signal Processing Algorithms[1]
Gusshausstrasse 25/389, 1040 Vienna, Austria
{mholzer,pbelanov,mrupp}@nt.tuwien.ac.at

Subject Area: Identifying important new research areas for wireless communication systems beyond the third generation.

1 Introduction

The development of third generation mobile communication systems shows that traditional design techniques, which have for a long time been sufficient, now have to be enhanced. The design of such highly complex future systems is no longer possible in an efficient way.

Software Defined Radio (SDR) is a new technological development that will allow in future to support entirely new transmitter and receiver functions purely defined by software modules. In particular, such modules will also define the underlying hardware architecture on which the transmitter and receiver algorithms will run. While the development of advanced hardware architectures that can support SDR will carry higher initial costs, these costs will stretch over a much larger product range. Since such architectures allow for a complete re-use, the development costs will in average decrease.

As it is defined, SDR will be a collection of hardware and software technologies that enable reconfigurable system architectures for wireless networks and user terminals. On the way from a pure hardware radio to an ultimate SDR, where new services can be downloaded via air interface, many more new tools and architectures have to be developed and investigated in order to make such a system possible. Additional investigations have to be done to integrate all these new tools and architectures into one efficient design flow, which meeting the tight time-to-market requirements.

2 Requirements of SDR

As a new technology, Software Defined Radio (SDR)[1,2] will present several challenges to both hardware/software system architectures as well as the used design methodology itself. Hence, these

[1] This work has been funded by the Christian Doppler Pilot Laboratory for Design Methodology of Signal Processing Algorithms.

challenges can be distinguished into architectural and design requirements, as elaborated on in the following sections.

2.1 Architectural Requirements

A traditional approach for meeting the complexity demands of third generation systems is to use multi core systems[3-6], where a DSP and a microcontroller are working together with additional help from hardware accelerators (IP blocks). If many tasks are written in software, the flexibility will be guaranteed, but the requirements of power consumption and computational power for SDR of mobiles will not be met.

2.2 Design Requirements

The SDR approach also requires new flexible and much faster design processes, able to effectively reuse descriptions of the system, analysis and processing results.

A new design process has to take into account and produce both software and hardware updates for a product in the field. Several types of updates can become available, such as communication protocols, user environments, applications, etc. There can be many ways of adding a new feature to a product. For example, customers can buy a special chip card with a new application or upgrade of an existing one and simply put it into mobile, where the RTOS[7] then reconfigures the product accordingly. The next way of updating can be directly through the service provider. Customer can download the configuration file into the mobile and update the product. This is one of the new services, which providers can offer to customers. Such high flexibility and speed of updating existing products requires also a very flexible and fast design process, which can reuse and build on existing system information.

3 A Consistent Design Methodology

The design process, leading from concept to realization, passes through three general levels of refinement, namely the algorithmic, the architectural and the implementation levels. Typically, three separate teams can be associated to one of these stages each [8].

3.1 Shortcomings of Current Design Methods

In the design process, the three teams have necessarily distinct areas of expertise, to tackle each of the three stages. Inherently, each of the teams works with a dedicated set of tools on a system description that is optimized for its work. As such, this framework has several shortcomings. Descriptions of the system at the three stages of the design process are fundamentally different, making forward and backward communications between teams highly difficult. Consequently, system descriptions are constantly reformatted and rewritten by the corresponding experts to incorporate input from the other teams. This mode of operation is error-prone, slow and inefficient.

All the mentioned drawbacks of the current design process are especially severe in the wireless field. Increasing complexity of algorithms, such as in UMTS and WLAN (e.g. 802.11a), as well as the extremely tight time to market and cost requirements, together place a great burden on the design process. Complex designs must be produced quickly and correctly the first time.

4 Design Methodology Based on a Single System Description

Clearly, significant increase in efficiency, reduction of time to market and improvement in quality can be achieved by providing a consistent design process. Such a process must provide a unified design environment, supporting all the teams equally and allowing them to work on a single system description. Thus, each team would apply its expertise and refine the single system description on its way from concept to realization, but at any point in time each team will have insight into the current description of the system, without manual translations, thus avoiding any communication obstacles.

Even when all the currently available tools used by the three teams are integrated into a unified design environment, significant steps in the design process are not covered. These are the steps that are currently carried out manually and only made possible by the expertise and experience present in the various design teams. Completeness of any consistent design process hinges on its ability to allow these design steps to be performed on the single system description, either directly and manually by the corresponding experts, or automatically by future dedicated tools. Examples of such significant, yet manual or poorly supported tasks are hardware/software partitioning, architecture mapping, and bitwidth optimization.

A flexible, easily expandable, secure and fast implementation of a single system description is in the form of a database. A consistent design process based on SystemC and a database implementation of the single system description, as proposed in [9], is shown in Figure 1.
Here, the system description is implemented as a database and operated on by a set of tools, each with dedicated "in" and "out" porting software. The three design teams provide inputs, such as desired system behavior and structure, constraints, and tool options. Also, the designers receive outputs, such as status of the system description, results of simulations, estimates of hardware costs, and timing. Some of the tools are those currently used by the design teams, like CCS or SPW, while others are specially written to perform missing tasks, either automatically or manually by designers, as described before.

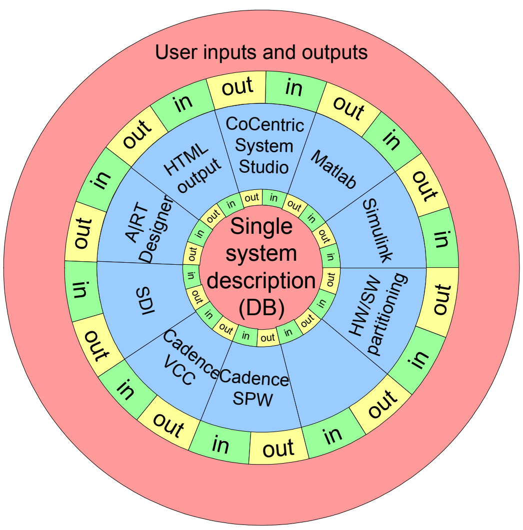

Figure 1: Consistent Design Process

However, the proposed consistent design process not only allows, such manual modifications to be performed more quickly and with less chance for error than they are currently, but it also provides for an easier environment to automate these procedures. Then, for example, the designer may not need to manually enter bit width information for each signal, but rather just provide necessary information to a range propagation algorithm, which enriches the database with bitwidth information automatically.

5 Meeting the Challenges of SDR

The design methodology shown in the previous section offers solutions to the new requirements brought about by the emergence of SDR systems. As shown in the following sections, this methodology introduces benefits over traditional design flows both in the initial design of SDR products, as well as in the development of key field updates that are so characteristic to the SDR technology.

5.1 Initial Design Challenges

SDR technologies herald new requirement for more sophisticated tools, working more closely together than those in currently available design flows. Such a close cooperation of different tools requires a unified information flow. This will be provided by usage of the design database, described in the previous section, and its communication interfaces. In this design methodology, requirements, intermediate results (reports), and design decisions are stored additionally to the system description so that the designer has fast and easy access to them.

Requirements like timing, area, and power can be specified for each SystemC instance. These requirements can also be extended to further ones. During the design flow design tools will generate results, which can be stored in the reports section. Comparison between requirements and results will help the designer to take further design decisions. These decisions can be given for each instance e.g. if the block should be implemented in hardware or software, or if that block should stay untouched for further optimizations.

5.2 Software Upgrade Challenges

A unique challenge posed by SDR systems is that of post-production development of updates for the system, including new protocols, applications and similar extensions. By addition of further functional blocks at the algorithmic level of the design, the system is extended to include the new functionality. The complete system can be verified at this level and once this is achieved, the design teams once again embark on the design process, refining the new system description toward the final product.

However, the re-design process should rely heavily on the initial design of the product. The target solution of the re-design process must match that of the initial design, only providing an extension. The extensions will be manifested in both hardware and software, utilizing the flexibility of the SDR platform in both those domains. In other words, a re-design update will enable the SDR system to provide new functionality to the user, by providing new firmware code (software) and new hardware configurations for flexible hardware of the SDR platform (hardware).

The re-design process will hence reuse many decisions made during the initial design. Hence, the initial design process provides shortcuts for the re-design process. Therefore, to enable the re-design process to take place, the design environment must provide all details of the initial design process.

Developments of field (post-production) updates to SDR systems are facilitated by the recording of every design decision in the single system description. Thus, the process of developing updates is not only made possible by the availability of this information, but also faster by the full automation and simplified expandability of the proposed design methodology, compared to efforts through traditional design flows.

6 Conclusions

This paper shows that a design methodology of an SDR requires a new approach to integrating all the different tools into a consistent design environment. Additionally, the high flexibility and speed of updating existing products in SDR requires a very flexible and fast design process, which can reuse and build on existing systems information. Provision of a single system description and a unified design environment to all teams in the design process has the potential to cope with the requirements for developing an SDR system.

7 List of References

1. Software Defined Radio Forum, www.sdrforum.org
2. Modular Multifunction Information Transfer System Forum on Software Defined Radio www.mmitsforum.org
3. S. Srikanteswara, J. H. Reed, P. Athanas, and R. Boyle "A Soft Radio Architecture for Reconfigurable Platforms"; IEEE Communications Magazine, Feb. 2000.
4. H. Zhang, V. Prabhu, V. George, M. Wan, M Benes, A. Abnous, and J. M. Rabaey "A 1-V Heterogeneous Reconfigurable DSP IC for Wireless Baseband Digital Signal Processing", IEEE Journal of Solid-State Cicuits, vol. 35, no. 11, pp. 1697-1704, Nov. 2000
5. B. Salefski and L. Caglar "Re-Configurable Computing in Wireless", Design Automation Conference DAC'01, pp. 178-183, 2001.
6. Z. Salcic and C.F. Mecklenbräuker "Software Radio – Architectural Requirements, Research and Development Challenges", The 8th International Conference on Communication Systems ICCS 2002, vol. 2, pp. 711-716, 2002
7. R.S. Janka, "Specification and Design Methodology for Real-Time Embedded Systems", Kluwer Academic Publisher, ISBN 0-7923-7626-9, 2002.
8. M. Rupp, A.Burg and E. Beck, "Rapid Prototyping for Wireless Designs: the Five Ones Approach", Signal Processing, vol. 83, no. 7, pp. 1427-1444, July 2003.
9. P. Belanović, M. Holzer, D. Mičušik, and M. Rupp, „Design Methodology of Signal Processing Algorithms in Wireless Systems", accepted for publication in International Conference on Computer, Communication and Control Technologies CCCT'03, Orlando, July 2003.