SystemCoDesigner – The System-Level Hardware-Software-Co-Design Tool

J. Falk, J. Gladigau, M. Glaß, C. Haubelt, S. Helwig, J. Keinert, M. Lukasiewycz,
T. Schlichter, T. Streichert, M. Streubühr, and J. Teich

codesigner@mycodesign.com

Hardware-Software-Co-Design - University of Erlangen-Nuremberg – Germany

http://www.mycodesign.com/research/scd

Abstract

SystemCoDesigner is a software tool for automatic design space exploration at the electronic system level and automatic platform-based prototyping of hardware/software systems using SystemC.

1. Introduction

80% of all design decisions are taken in the first 20% of the design time. Thus, a substantiated knowledge about possible solutions is mandatory to design high quality systems. SystemCoDesigner is a software tool for automatic design space exploration at the electronic system level and platform-based prototyping of hardware/software systems. By exploring the design space, a designer becomes more confident in decisions to be done. Many software tools exist at the electronic system level, but to the best of our knowledge none of them covers all aspects from specification over automatic exploration to automatic prototype implementation (cf. [4]). In the following, we will briefly describe the SystemCoDesigner design flow itself and the SystemC[1] input to this design flow.

2. SystemCoDesigner Design Flow

SystemCoDesigner is a software tool for automatic design space exploration at the electronic system level and hardware/software prototyping. The input is a functional model written in SystemC. During exploration, the goal is to optimally allocate resources and bind the SystemC modules onto these allocated resources. After decision making a prototype implementation of the resulting hardware/software system can be generated. The proposed design flow is shown in Figure 1.
As a first step, the designer has to write an executable model of the behavior of the application using SystemC. For automatic prototype implementation and simulation-based estimation during exploration it is necessary that the application is specified using the SysteMoC library (see Section 3). In a third and fourth step, the designer has to specify the architecture template and all possible mappings of SystemC functions onto the resources in the architecture template. Again, the prototype implementation demands the usage of predefined hardware resources (CPUs, busses, IP cores, etc.). Hardware accelerators can either be written manually by transforming SysteMoC modules or can also be synthesized automatically (Step 2).

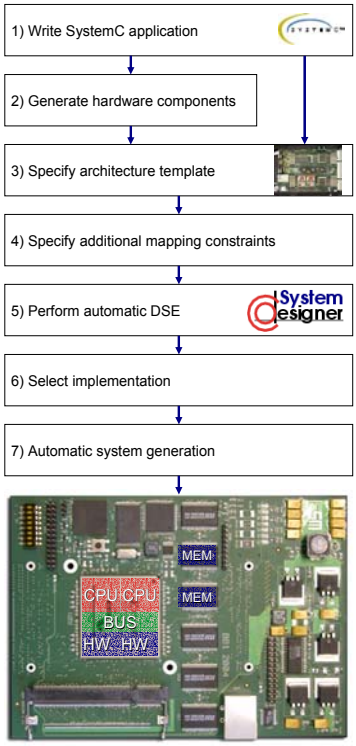

Figure 1: SystemCoDesigner design flow from SystemC to an FPGA-based hardware/software system.

Using this input information, the Design Space Exploration (DSE) can be performed automatically [1,3]. For this purpose, SystemCoDesigner uses state-of-the-art multi-objective optimization algorithms. However, finding good solutions from scratch is like looking for a needle in a haystack. To improve performance, symbolic techniques have been integrated into SystemCoDesigner [5,6].
SystemCoDesigner supports the evaluation of designs during exploration using SystemC simulation. Automatically generated system level performance models permit a fast estimation of latency and throughput numbers by a combined behavioral and timing simulation considering allocation and binding effects [7]. For this purpose, the allocated architecture is modeled in SystemC at a high level of abstraction. This allows for exploring effects from het-

[1] SystemC is a trademark of the Open SystemC Initiative

erogeneous multi-processor architectures, scheduling policy, and resource contention.
After selecting an implementation (Step 6), the System can be automatically implemented (Step 7). As a proof of concept, the mapping onto Xilinx FPGAs using Xilinx MicroBlaze[2] softcore processors and manually designed hardware components has been shown in [3].
SystemCoDesigner provides a graphical user interface, including a front end for specification of the architecture template and mapping possibilities as well as a visualization of the design space exploration and optimal implementations (see Figure 2). Using the GUI, the designer can further perform decision making by selecting a solution for prototype implementation. Moreover, the system level performance model used throughout the exploration can be generated and simulated using SystemC. The visualization of simulation traces is supported through automatic waveform generation. These waveforms are an additional support in design verification.

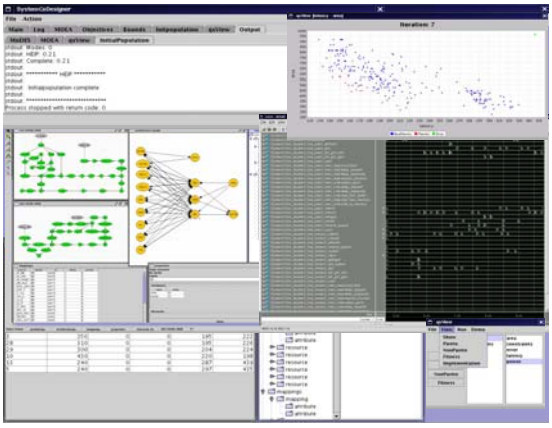

Figure 2: SystemCoDesigner GUI allows to control design space exploration parameters as well as the visualization of exploration results and simulation traces.

3. Automatic Platform-Based Hardware/Software Prototyping

In order to use the automatic hardware/software prototyping features, a synthesizable subset of SystemC must be used as input to SystemCoDesigner. Beside the synthesizable subset of a C-based behavioral synthesis tool, the designer has to specify the application using the SysteMoC library [2]. SysteMoC is a SystemC-based library. In addition to the actor-oriented concepts of SystemC requiring that communication between modules is only performed using channels connected to the port of a module, SysteMoC distinguishes between the communication behavior and the actions performed by a SystemC module. This is done by specifying a so called *firing FSM*. The firing FSM models the externally visible states of an actor as well as state transitions and corresponding activation conditions.
The execution model of a SysteMoC design is as follows: Firstly, for each actor it is tested whether a state transition exists with a fulfilled activation condition leaving the current state. Secondly, for each actor with at least one activated state transition, one of them is chosen non-deterministically. Finally, all channels in the system are updated. This execution model omits unnecessary event updates inside the SystemC simulation kernel and, thus, improves the simulation speed. For more details see [2].
SystemCoDesigner supports automatic software synthesis from SysteMoC descriptions and hardware/software system generation for Xilinx FPGAs using Xilinx MicroBlaze softcore processors. Hardware synthesis requires behavioral synthesis tools such as Cynthesizer[3] by Forte Design Systems or Catapult[4] by Mentor Graphics.

4. Conclusion

SystemCoDesigner is a powerful tool for increasing the confidence in early design decision through an automatic design space exploration at the electronic system level and automatic FPGA-based hardware/software prototyping. To the best of our knowledge SystemCoDesigner is the first tool available for automatically going from SystemC behavioral descriptions to hardware/software implementations including automatic optimization.

5. References

[1] T. Blickle, J. Teich, and L. Thiele. System-Level Synthesis Using Evolutionary Algorithms. Journal on Design Automation for Embedded Systems, 3(1), pp. 23-58, 1998.
[2] J. Falk, C. Haubelt, and J. Teich. Efficient Representation and Simulation of Model-Based Designs in SystemC. In Proceedings FDL'06, Forum on Design Languages, pp. 129 - 134, 2006.
[3] C. Haubelt. Automatic Model-Based Design Space Exploration for Embedded Systems - A System Level Approach. Ph.D. Thesis, University of Erlangen-Nuremberg, 2006
[4] C. Haubelt, J. Falk, J. Keinert, T. Schlichter, M. Streubühr, A. Deyhle, A. Hadert, and J. Teich. A SystemC-based Design Methodology for Digital Signal Processing Systems. In EURASIP Journal on Embedded Systems, Special Issue on Embedded Digital Signal Processing Systems, 2007.
[5] C. Haubelt, T. Schlichter, and J. Teich. Improving Automatic Design Space Exploration by Integrating Symbolic Techniques into Multi-Objective Evolutionary Algorithms. In International Journal of Computational Intelligence Research (IJCIR), Special Issue on Multiobjective Optimization and Applications, 2(3). pp. 239-254, 2006.
[6] M. Lukasiewycz, M. Glaß, C. Haubelt, and J. Teich. Symbolic Archive Representation for a Fast Nondominance Test. In Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization, 2007.
[7] M. Streubühr, J. Falk, C. Haubelt, J. Teich, R. Dorsch, and T. Schlipf. Task-Accurate Performance Modeling in SystemC for Real-Time Multi-Processor Architectures. In Proceedings of Design, Automation and Test in Europe, pp. 480-481, 2006.

[2] MicroBlaze is a trademark of Xilinx, Inc

[3] Cynthesizer is a trademark of Forte Design Systems
[4] Catapult is a trademark of Mentor Graphics Corporation