論文の内容の要旨

AMS Extensions for C-Based System-Level Design Language

（Ｃベースシステムレベル設計言語におけるＡＭＳ領域への拡張に関する研究）

氏名　　　　　劉　宇

Abstract

Recently, system-level design languages (SLDLs), which can describe both hardware and software aspects of the design, are receiving attentions. However, most of SLDLs focus on digital systems. Analog mixed-signal (AMS) extensions to SLDLs enable current discrete-oriented SLDLs to describe and simulate not only digital systems but also digital-analog mixed-signal systems.

C/C++ based SLDLs such as SpecC and SystemC have been gaining greater attentions because uniform languages for hardware and software description can make system-level design more efficient. In this research, we work on the analog and mixed-signal extensions to the C-based system-level design language, such as SpecC.

SpecC language is a C-style language for the system-level design. It focuses not only on the system's simulation but also on the system's synthesis. With SpecC methodology, the system design starts from the capture of system's functionality in specification model, then, by the architecture exploration, the specification model is partitioned, allocated and scheduled into architecture model with the more detailed information contained, and then, by implementing the communications among the behaviors in architecture model, the architecture model is detailed into communication model having the detailed communication implementations, finally, the communication model is synthesized to implementation model by mapping the implementation model with RTL IP and RTOS IP. In the methodology of SpecC, the specification model is usually untimed, i.e., there is only execution order defined in the specification model. With introducing more detailed information of implementation during the design phases, the time of the behaviors is estimated and cycled, so called it is timed.

The AMS extended language of SpecC, named as SpecC-AMS, has the same syntax style

with original SpecC. It supports the models of both continuous and discrete behaviors, data and time which are needed in the mixed-signal system-level design with software and analog, digital circuits. It also supports the hierarchy with which a complex system can be decomposed into a set of interconnected pieces.

SpecC-AMS captures the functionalities of continuous specification in an extended class named as *con behavior*. Con behaviors communicate with each other via the ports of extended continuous variables. The con behaviors that are chained with a same network of continuous variables are grouped into a *CNT*. The CNT exchanges data with discrete behavior through the extended class named *ms channel*. The synchronization between CNTs and discrete behaviors are via the extended event type named as *AD Event* and *DA Event*.

SpecC-AMS models the continuous functionality with the solver-based method. The solver-based descriptions can provide flexibilities and conveniences for user to specify the functionalities of continuous behavior. The continuous solvers of SpecC-AMS are divided into built-in and built-out solvers. We integrate two of the most generally used continuous solvers in current version of SpecC-AMS. They are linear state-space equation solver and transfer function solver, which are widely used for modeling the continuous functionalities of time-domain and frequency-domain respectively.

SpecC-AMS code is executable with the extended compiler and simulation library. We extend the simulation kernel of SpecC so as to synchronize the continuous solver and discrete simulator processing concurrently in the distinguish natures. The synchronization is performed upon a new synchronization mechanism, so called *transaction-blended synchronization method*. Comparing with the traditional optimistic and lock-step synchronization method, transaction-blended method has the features as: 1) accommodation of both timed and untimed discrete description, 2) unlike optimistic concept, no backtracking is needed, and 3) less disturbance than in the lock-step concept.

We evaluate the SpecC-AMS with serial examples. The evaluation results show the extensions work well under both the timed/untimed system-level description, and have many attractive advantages than traditional design languages.