Abstract

TLM Synthesis Studio provides an environment to design, simulate, and synthesize complex digital systems in transaction level. It provides the designers with a library of configurable TLM components.

1. Introduction

In the recent years, the Electronic System Level (ESL) industry has defined a level of abstraction in modeling and design of systems. This level of abstraction is generally higher than RTL in that the timing details are not considered and the communications between modules are modeled by high-level channels instead of a wire or a group of wires. This level of abstraction, which is going to become the starting point in system level design, is called transaction level and the act of modeling a system in this level is referred to as Transaction Level Modeling (TLM). In a general definition of TLM, the system is divided into two parts: communication and computation parts. In this definition, TLM is considered as modeling the communication parts of a system at a high level of abstraction (e.g., by functions). With this definition, the computation parts (modules) of a design can be at various levels of abstraction. It is obvious that the higher level the modules are designed, the faster their simulation process and the easier their connection with communication parts are.

TLM Synthesis Studio provides an environment to design, simulate and synthesize complex digital systems. In our system level design flow, a designer can partition high level system specification into computation and communication parts. In the next step, the designer can model the communication parts by our powerful TLM library of communication protocols. The computation parts can be modeled in terms of high level C/C++ or SystemC RT level. TLM synthesis uses a library of synthesizable TLM protocols to synthesize transaction level descriptions into SystemC RTL code. The designer can either specify the TLM design in terms of several of the supported commercial bus structures, packet switches, or use the standard TLM channels in his or her design to be synthesized. Either way, our tool creates synthesizable RT level SystemC code from the TLM design.

2. TLM Design Methodology

Figure 1 shows a TLM based system design flow. At first step, a system designer partitions the system into computation parts and communication parts. The computation parts of the system are partitioned tasks of system in the traditional system design flow. Each computation part can be implemented in hardware or software. Since system is specified in transaction level with a high level specification language like C++, both HW and SW parts of system can be modeled in a high level C++ language. For the communication parts, the designer must design and implement the structure of interconnections between modules and their communication protocol.

In the next stage of the TLM system design flow, the designer can implement the hardware computation parts in RTL SystemC and even high level C++. Software computation parts are developed in high level C++. Communication parts can be implemented using the high level components of the TLM library. The completed system can be simulated and verified in a SystemC simulation environment. For more accurate system modeling and simulation, the designer must replace the high level communication parts with the corresponding cycle accurate or RTL models. Therefore, a TLM to RTL synthesis tool can help designers with system level design flow especially in order to fast design space exploration.

3. TLM Synthesis Framework
We have developed a framework that includes a set of predefined classes for describing ports and different kinds of modules (See Figure 2). We have considered the following parameters in the design of TLM framework:

- **TLM synthesis purpose:** The main purpose of designing TLM framework is synthesizing transaction level description of digital systems into RT level. Since the TLM library has general components to model the communication and computation parts of system and many low level details are hidden from the designers, it is difficult to extract the corresponding RTL model of the same system. Therefore, having a synthesis tool can accelerate the design process of large digital systems.

- **High simulation speed:** Since high simulation speed is a major factor for fast design space exploration, our aim is to design a framework and components with higher simulation speed such as using function calls instead of TLM channels.

- **Accurate modeling:** One problem in transaction level design is accurate system modeling. It means that the execution order of high level description of the system must be the same as the execution of the corresponding real system.

- **Design refinement:** In top-down design flow, a designer describes his or her system in a high abstraction level. At each level, the designer makes some decisions about the architecture of the system. Then the designer must refine that architecture into the lower level of abstraction and incorporate more details into it. We have considered several levels in our framework which can be used in the design refinement process.

4. TLM Studio Library

The TLM Studio Model Library is a part of TLM Synthesis Studio solution. It contains a collection of SystemC models, including embedded processors, bus components, peripherals, and memory modules. All components are available at different levels of abstraction including un-timed transport, timed transport, phase-order, and cycle-accurate levels. TLM Synthesis Studio components are also compatible with TLM standards from OSCI to enable IP re-use.

5. TLM Synthesis Studio Environment

Our graphical user interface provides an environment for system level design and verification. It consists of the following parts:

- **System design:** provides facilities for managing projects and systems. A designer can easily design and configure a complete system using provided TLM library components.

- **File view:** enables users to edit their generated TLM source files. It has common edit features as well as syntax highlighting for TLM, SystemC and C++.

- **TLM simulation:** is provided an interface and a link to the free OSCI library and free C++ compilers.

Figure 3 shows an overall view of the developed environment.

6. Conclusions

In this project, we have developed an environment for system level design and synthesis based on SystemC and OSCI TLM library. It provides a library of TLM components at different levels of abstraction to speedup design process and simulation.

7. References


Acknowledgement

The authors would like to thank the High-Tech Industries Center (HTIC) for their funding of this project. The support of HTIC has provided funding for personnel, equipment, exhibition, and management of this and several other CAD research and development.