SCope: SoC Co-simulation and Performance Estimation in SystemC

Héctor Posadas[1], David Quijano[1], Eugenio Villar [1] & Marcos Martínez[2]

{posadash, quijano, villar}@teisa.unican.es, marcos.martinez@ds2.es

[1] University of Cantabria, GIM, TEISA Dept

http://www.teisa.unican.es/gim

[2]DS2, Robert Darwin 2, Parque Tecnológico, Paterna, Spain

http://www.ds2.es

Abstract

SCope is a SystemC extension for system co-simulation and analysis. SCope includes capabilities for OS modeling, performance estimation (Time & Power) and time annotation. It also allows modeling the HW platform.

1. Introduction

Co-simulation has become one of the most important issues in HW/SW co-design of very complex systems, especially for Multiprocessor System on Chip (MpSoC). One of the most important ways of simulating complex HW/SW systems is the use of high-level languages. Among them, SystemC is one of the most accepted languages in the designer community.

However, the use of SystemC presents some limitations. The simulation of SW elements requires including the effects of a Real-Time Operating System (RTOS), and the HW platform both together. Thus, an extension of SystemC is necessary. SCope provides capabilities for modeling both the SW operation and the HW interconnections in a SystemC environment.

2. SW modeling and Performance Estimations

Two are the main limitations of SystemC for SW modeling. First, the execution of the refined SW code produces an untimed simulation. As a consequence, the system cannot be accurately co-simulated and performance estimations cannot be obtained. The timing effects of the target platform in the SW execution time are critical when modeling the whole system.

Secondly, SystemC does not directly support several features presented in typical RTOS. The SW refinement requires a model of the RTOS mechanisms for concurrency, scheduling, communication and synchronization. Thus, the simulation framework has to include a RTOS specification that provides all the common capabilities in the standard operating systems.

To overcome these two limitations, the previous tool PERFidiX has been integrated in SCope. This tool manages the SW execution. First, PERFidiX automatically instruments the SW code to obtain execution time estimations. The library dynamically estimates the time cost of the SW segment that is been executed. After that, the estimated time is annotated at the correct points where required, not only at static predefined points. Thus, the untimed simulation is moved into a timed one.

Furthermore, PERFidiX models a multiprocessor OS based on the POSIX API. Processes and threads can be scheduled using the POSIX defined priorities and policies. Channels, as mutexes, semaphores or message queues, and POSIX signals can be also used to communicate and synchronize the system SW components. For network communications, lwIP, a TCP/IP stack, has been integrated to implement the socket functions of POSIX.

The library also contains a set of Linux-based low-level I/O functions for drivers modeling. Interruption management is also considered in the OS model. HW interruptions are received from the system bus and execute the corresponding interrupt handlers. Some drivers, as a network driver model has been included in the OS model.

Finally, some middleware can be modeled with SCope. A CORBA model has been placed on top of SCope to run CORBA components, in collaboration with TIMA. This model uses the OS capabilities, and the bus and network models to simulate the execution of CORBA applications.

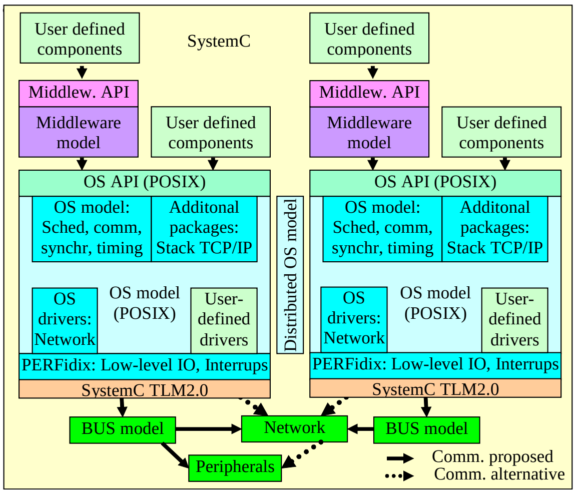


Figure 1: Scope system modelling

3. HW Platform Modeling

To analyse adequately MpSoC, the HW platform has to be considered. Thus, co-simulation mechanisms are required. These mechanisms involve modeling the communication infrastructure, and the HW module interfaces. Two communication models are provided to the user with their corresponding interfaces: bus and network.

To model buses, the TLM2 standard has been adopted. The library provides a fast, generic bus model with priority management and bandwidth control. The model allows including delays, connecting multiple masters, and bus chaining. Generic interfaces to connect the bus with the OS and specific peripherals are also provided. The library also includes some examples of peripherals as a network interface to connect the bus and the NoC.

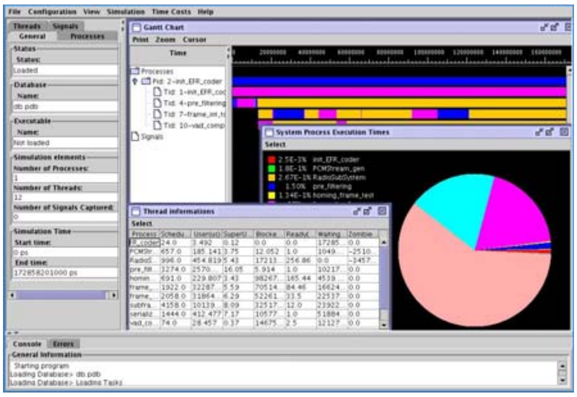
To model networks, the Sicosys tool has been also integrated. This tool is optimised to model MpSoC networks, allowing definitions of multiple network configurations, routing protocols or router models. The tool is able of delivering the packages at the adequate nodes and estimating the delay and network utilization depending on the network characteristics.

Regarding to the system architecture proposed, Scope proposes a platform where each node contains each bus, with processors, memory and peripherals. The operating system is able of moving tasks among the processors of the same node dynamically, and even among different nodes, with some limitations.

4. SCope GUI Tool

To simplify the use of SCope and optimise results a graphic interface is provided. This interface allows the designer to easily configure the library to model adequately the target platform and to analyse the simulation results.

With that tool, the processors utilizations, process executions, channels uses and some platform information can be graphically analysed.



Status
Status:
Loaded

Database
Name:
db.pdb

Executable
Name:
Not loaded

Simulation elements
Number of Processes:
1
Number of Threads:
12
Number of Signals Captured:
0

Simulation Time
Start time:
0 ps
End time:
172858201000 ps

Process	Schedu	User(us)	Super(s	Blocke	Ready(	Waiting	Zombie
FR_coder	24.0	3.492	0.12	0.0	0.0	17285...	0.0
PCMStr...	657.0	185.141	3.75	12.052	1.0	1049...	-2510...
RadioS...	996.0	454.819	5.43	17213...	256.86	0.0	-1457...
pre_filt...	1274.0	2570...	16.05	5.914	1.0	10217...	0.0
homin...	691.0	229.807	3.43	98267...	165.44	4539...	0.0
frame_...	1922.0	32287...	5.59	70514...	84.46	16624...	0.0
frame_...	2058.0	11864...	6.29	52261...	33.5	22537...	0.0
subfra...	4158.0	10139...	8.09	32517...	12.0	23922...	0.0
serializ...	1444.0	412.477	7.17	10577...	1.0	51884...	0.0
vad_co...	74.0	28.457	0.37	14675...	2.5	12127...	0.0

```
Starting program
Loading Database> db.pdb
Loading Database> Loading Tasks
```

Figure 2: SCope interface Tool

This tool will be available soon from:
http://www.teisa.unican.es/scope
(Meanwhile, more information can be found at http://www.teisa.unican.es/perfidix)

5. Conclusions

Scope is a SystemC library that extends the SystemC kernel without modifying it. The library allows the designer to simulate refined SW components containing POSIX functions or middleware interfaces, over a complete MpSoC model.

The library provides a complete OS functionality, and produces timed simulations of the SW components, using dynamic estimation and annotation.

To model the platform, bus and network timed models are provided, allowing some configuration options and the possibility of easily integrate HW components.

Finally, the tool provides a graphic interface where the simulation results can be analysed to optimise the system refinement.

6. References

[1] H. Posadas, F. Herrera, P. Sánchez, E. Villar and F. Blasco: “System-level performance analysis in SystemC”, in Proceedings of the Design, Automation and Test Conference, IEEE, 2004.

[2] H. Posadas, E. Villar and F. Blasco: “Real-Time Operating System modeling in SystemC for HW/SW co-simulation”, in Proceedings of DCIS, IST, Lisbon, 2005.

[3] H. Posadas, J. Adámez, P. Sánchez, E. Villar and F. Blasco: “POSIX modeling in SystemC”, Proceedings of the Asia and South-Pacific Design Automation Conference, IEEE, 2006.

[4] H. Posadas, J. Adámez, E. Villar, Francisco Escuder, Francisco Blasco: "RTOS modeling in SystemC for Real-Time embedded SW simulation: A POSIX model", Design Automation for Embedded Systems, V.10, N.4, Springer, pp.209-227. 2006-12

[5] H. Posadas, D. Quijano, E. Villar, Francisco Escuder, Marcos Martínez: "TLM interrupt modelling for HW/SW co-simulation in SystemC" XXI Conference on Design of Circuits and Integrated Systems, DCIS'06 . 2006-11

Acknowledgement

This project has been funded by the MEDEA+ LoMoSA+ project and the Spanish MEC through the TEC2005-03301 project