Automatic SystemC Design Configuration for a Faster Evaluation of Different Partitioning Alternatives

N. Bannow, K. Haug
Robert Bosch GmbH, Automotive Electronics
Driver Assistance Systems
Nico.Bannow@de.bosch.com,
Karsten.Haug@de.bosch.com

W. Rosenstiel
University of Tübingen
Wilhelm-Schickard-Institut für Informatik
Rosenstiel@informatik.uni-tuebingen.de

Abstract

In this paper we present a methodology that is based on SystemC [1] for rapid prototyping to greatly enhance and accelerate the exploration of complex systems to optimize the system architecture. The approach introduces a methodology to automatically configure system components with regards to the memory mapping of modules. The approach reduces the implementation effort that in conventional approaches has to be done by hand to re-assign and re-configure modules in a system. This does not only save time for manual adaptation but also reduces the chance to introduce errors like known from complex manual modifications. The new approach for automatic system configuration is derived as one of the results and features that come along with the Module-Adapter (MA) based approach that we have proposed in different presentations [2], [3], [4]. One of the main goals, our proposed methodology has to fulfill, are industrial requirements such as applicability for complex system development, integration of existing IP, improved code quality and decreased development effort. The automated system configuration as well as the whole MA based approach greatly support the designers in the concept phase to simulate a design before the implementation starts.

1 Memory map and requirements

Usually the modules of modern complex designs are memory mapped. The basic idea behind memory mapping is to select modules only via addresses rather than using an additional ID or something equal. For any address this address decoder chooses the accessed bus port and therefore the module behind this bus port. The address to port mapping is usually hard-coded in the address decoder of the bus. Because the address maps are fix, they can not be changed at runtime.

Often, in an early prototyping phase, the system's target architecture is still unknown. Most likely for the modules themselves, the memory mapping – meaning the used address space – will not change to often and therefore can be fixed in an early design phase. However, for every system-reconfiguration, a new address decoding in the affected address decoder of the bus is required. Especially, if in the conception phase modules are shifted from one bus to another or if new modules, buses or bus-bridges are invoked, the corresponding address decoders need to be adapted. Especially affected is the bus with its address decoder that needs to contain the information which addresses are mapped to which port. Therefore, manual changes are required after any system re-partitioning. Even a single modification requires a feasible manual implementation effort inside all buses and is therefore also very error-prone. With a growing system complexity, this effect is rapidly increasing.

2 Automating system configuration

The proposed approach consists of components that "talk" to each other in an initialization phase before the simulation starts. Any required information of the system is exchanged in this period like shown in Figure 1:

During the initialization phase, every module sends one special initialization request to the bus, it is connected to. Any request also contains a pointer to the module's parameter object. Here, beside many other parameters, also the information about the address map is contained. One more information the bus gains during initialization is, whether the connected module is a bus-bridge or not. While the modules send an initialization request to the bus, the bus stores the module information in a list. After all modules sent an initialization request to the bus to which they are connected to, the bus evaluates if all modules are correctly initialized. Than, the bus itself initializes all directly connected modules.

Now in this stage the buses contain all information of the address map to port relations of the directly connected modules. However, the buses still do not have any information of modules that are not directly connected, meaning of modules that are indirectly connected through

3-9810801-0-6/DATE06 © 2006 EDAA

the bus-bridges. Therefore, in the next step, the information for all modules that are registered in the buses are send as initialization requests over all bridges that are directly connected to each bus. Each bus-bridge that receives an initialization request from one bus, forwards this request to the other bus. All of these module initialization requests that are received by the buses through one bus-bridge are forwarded to all other bus-bridges to which the bus is connected to. In this way all address mappings are forwarded through all buses via all bus-bridges. Therefore, after initialization, all buses are notified about all address mappings.

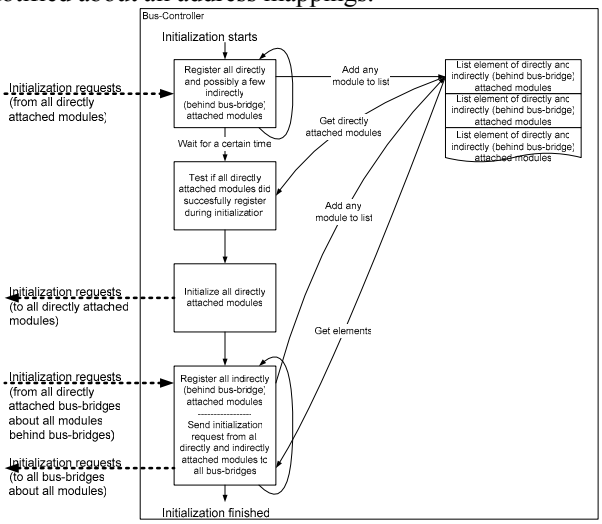

Figure 1: Bus initialization phase

In the current implementation of the MA-based simulation model, no initialization requests are allowed after the initialization phase of the bus. Therefore, modules can not be added during runtime and the memory mapping can not be changed. This could easily be modified, however, this mechanism by default is disabled because most standard buses of embedded systems do not support re-configuration at runtime.

During initialization, the bus creates a list of all directly and indirectly (via bus-bridge) connected modules. But how does the bus maps addresses at runtime to ports? Any requesting module from the directly connected module list that wins the arbitration is being granted. The winning requested destination address is now evaluated with the full address map list like shown in Figure 2. If the address fits to an address map of this list, the used port of this module is identified. The destination module can either be directly connected to the bus or it may be located behind a bus-bridge. In the first case the corresponding module is directly accessed or in the second case, the request is send to the bus-bridge behind which the required address is located. If the request is send to a bus-bridge, this bridge now forwards the request to the connected bus until it is granted.

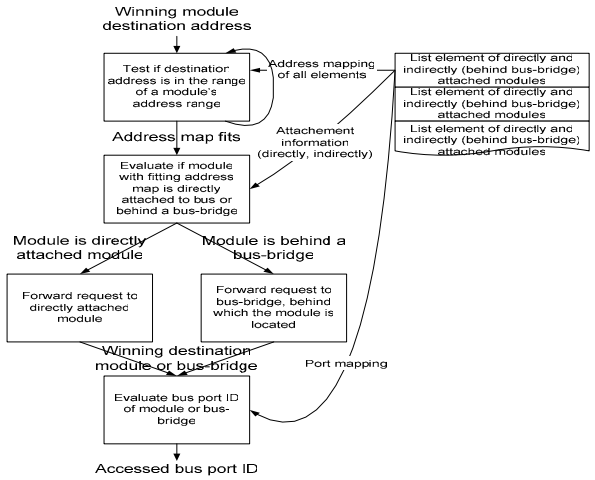

Figure 2: Runtime address to port mapping

3 Configuration of real systems

The described approach of an automatic system configuration takes part in the system initialization phase prior to the default behavior. It is mainly developed for simulation issues. Therefore, once the simulation model is moved into a real system, the module information that formerly were transferred in the initialization phase will no longer be transferred but hard-coded inside the corresponding modules and buses. Nevertheless, it is also possible to move the approach of an automated system configuration from simulation into a real system. However, in such a case all modules and also the buses and bus-bridges need to have re-configuration capability.

4 Benefits

We have shown that the MA-based approach as well as the derived methodology to automatically configure a system has major advantages versus existing approaches:

- Automatic system re-configuration,
- Faster evaluation of different system architectures,
- Reduced implementation time for faster time to market and therefore better cost efficiency,
- Less error prone than manual implementation,
- Integrated into the MA-based simulation approach.

References

[1] Open SystemC Initiative. *SystemC, Version 2.0.* www.systemc.org, 2001

[2] N. Bannow, K. Haug, W. Rosenstiel. *Performance Analysis and Automated C++ Modularization using Module-Adapters for SystemC.* FDL'04

[3] SpeAC, MEDEA+/BMBF Project Number: 01M3067B

[4] N. Bannow, K. Haug. *Experience in High Level Modeling with SystemC using TLM.* ECSI TLM Workshop, 2005