Annotated Data Types for Addressed Token Passing Networks

Gordon Cichon * **
gordon@cichon.de
* Infineon Technology Corp.
System Architecture, Data Communication
San Jose, CA 95112, USA

Winthir Bunnbauer * **
winthir@brunnbauer.org
** Technische Universität München
Institut für Integrierte Schaltungen
D-80290 München

Introduction to Annotated Data Types Annotated data types have proven to be a practical description form for interfaces of SoC components to random addressable buses (see [1]). The central idea behind this new approach is to define a component's interface to a random addressable bus in terms of a data structure it exposes to this bus. This data structure is modeled using a type system similar to that of computer languages, like C or VHDL. This data structure is annotated with additional information relevant for hardware description purposes. In [1], the underlying terminology and framework, as well as a method for synthesizing the functional adaptor part of hardware components, has been described.

Application: Token Passing Networks The paper applies annotated data type descriptions of the hardware component interfaces to a synthesis of its internal communication structure. The implementation of a component's functionality is recursively decomposed into smaller blocks which by themselves communicate through random addressable bus interfaces. This represents a form of token passing network or Petri Net. To facilitate the communication through random addressable bus interfaces along the arcs of the network, the information carried by each token is partitioned into an address part and a data part.

This new approach shows advantages for implementing hardwired pipeline structures: A distinction of tokens into state changes and workload data allows the synthesis of a state distribution network that can handle the propagation of state changes autonomously and efficiently. Figure 1 shows how the communications shell for a single node of such a network is synthesized. Additionally, our design system facilitates the synthesis of superscalar scheduler nodes which dispatch workload data among several functional blocks symmetrically, and reorder the results.

Conclusion This new design methodology has been shown to greatly reduce the overhead of communication and housekeeping tasks in the design flow. Much of the information utilized by the the discussed synthesis tools can be reused from the external interface synthesis and testbench synthesis of [1].

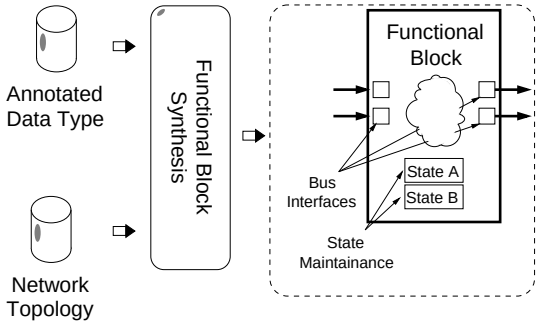

Figure 1. Communication Synthesis for Functional Blocks

This allows rapid development of a large number of new components with a complex inner structure within rapid timelines. Synthesis is completely under the control of the designer, there are no hidden algorithms or heuristics that affect the synthesis results in an unpredictable manner. Since the communication topology description together with the annotated data types of the interfaces are a dense description, the implementation model can be changed quite easily. The synthesized communication structure itself also benefits from the elaborate test infrastructure developed for random addressable bus interfaces.

References

[1] Gordon Cichon. Annotated data structure declarations for bus interface synthesis. In *Forum on Design Languages, Tübingen*, 2000.