Building a UML Profile for On-Chip Distributed Platforms

Tuomas Lindroth¹, Raicea Lavinia², Tiberiu Seceleanu¹, Nastooh Avessta¹ and Jukka Teuhola¹
¹ Dept. of Information Technology, University of Turku, Turku, Finland
² Dept. of Applied Electronics, Polytechnic University, Bucharest, Romania
E-mail: {tuomas.lindroth, tiberiu.seceleanu, nastooh.avessta, jukka.teuhola}@utu.fi, raicea.lavinia@elia.pub.ro

Abstract— In this paper, we approach the definition of design methodologies for on-chip distributed platforms. The focus is on one segmented bus approach, the SegBus platform. The approach suggests the creation of a Unified Modeling Language (UML) profile that will allow a unified representation of both platform and application.

Segmented bus; UML; platform profile

I. INTRODUCTION

The approach based on UML starts by building a profile targeting the SegBus platform. The profile will provide, at first, the graphical elements for describing a specific system instantiation, at high levels of abstraction.

However, the most important part of the profile consists of specifying the structural constraints of the platform. For instance, it is important to enforce that in the design there are not introduced more segment arbiters than the number of segments, that there is a single central arbiter unit for the whole platform, and so on. These requirements are specified as UML constraints in OCL [3].

Considering the SegBus profile, an actual instantiation of the platform will follow a class diagram model, which allows a better specification of platform properties, while following along the lines projected by the profile constraints.

II. BUILDING THE SBPROFILE

The development of the profile starts by concentrating on the structural characteristics of the platform, and their representation as UML elements [4, 5]. The SBProfile is a collection of stereotypes to sustain application development on the SegBus platform. It also contains an additional UML model, for the application representation, in order to be able to map the application to the target platform.

In the following, analysis proceeds with the modeling of hardware system resources, seen from the SegBus platform [1, 2] perspective. The profile defines the structure elements of the platform. Hence, it contains the platform itself, the stereotype SegBusPlatform, one element modeling the segments, Segment, the stereotype representing the SA, SegmentArbiter, etc. All the elements are generalizations of the metaclass uml20.classes.Class. The structural view is illustrated in Figure 1.

Figure 1 The SBProfile elements.

Hierarchy While the profile specification of the above figure presents a “flattened” view of the platform, an actual model must contain indications on hierarchical composition of the system.

Thus, at the top level is the platform itself, SegBusPlatform composed of the Segment(s) and one CA. The Segment, in its turn, will be composed of several functional units, one segment arbiter, one multiplexer and bus lines. The connection between these modules is done via border units.

Still, the structural element definition of the profile and its prospected utilization in system build-up is, however, not enough. Also the connections between these elements have to be defined, as compositions, aggregations, etc. As an example, later is the description of association between the central arbiter unit and segment arbiters (the CA_SA stereotype, generalizing the uml20.kernel.AssociationClass - as in Figure 1).
Along the same lines, a “functional” element of the platform (FU) is described by the FunctionalUnit stereotype. This virtual element can consist either of one master, one slave, or as one master and one slave. Thus, a proper aggregation must be specified for this element, also derived from uml20.kernel.AssociationClass. This is visualized in Figure 2.

Figure 2 The SBProfile association definitions and constraints.

**Constraints** It becomes necessary that the relations between the profile elements are restricted as mentioned earlier, by associating constraints written in OCL.

**Description of the profile elements.** The platform (SegBusPlatform) is characterized by the number of segments it contains, by the geometry - linear or circular, the size of the data packages used in communication, and by the widths of data and address lines.

In its turn, each of the Segment(s) is composed of one SegmentArbiter, and FunctionalUnit(s) (masters and / or slaves), and interfaces to the neighboring segments BorderUnit(s).

The central element of a segment is the SegmentArbiter. This coordinates both the intra- and inter-segment communication. The active units of the system are represented by the masters. They are contained by FunctionalUnit(s). One FunctionalUnit may contain up to one Master and one Slave (see Figure 1 for the related constraint). The FunctionalUnit’s (unique at the system level) ID (natural number) is inherited by both the contained Master and Slave. The FunctionalUnit methods contain procedures to produce data and to communicate with the SegmentArbiter. Sending and receiving data are procedures placed within Masters and Slaves, respectively.

The BorderUnit element is an interface between one segment and its neighbors. The internal FIFO is characterized by the Depth tag.

The platform model may be better illustrated otherwise as a class diagram, where the constraints introduced follow from the employment of the SBProfile. A high level vision of one platform instance is given in.

**III. CONCLUSION**

A UML-supported methodological roadmap was presented for specifying, modeling and implementing embedded systems on segmented bus architecture. The principles of a UML-based SoC design methodology were analyzed and the SegBus design methodology introduced. One of the main aspects of the process consists of the (partial) creation of the SBProfile, for which the principal structural elements are described.

It comes as a natural observation that a UML-only based approach to SoC platform-based design is not feasible. UML can, however, be used for structuring and relating the different aspects of system design.

**REFERENCES**