MODELING OF SOFTWARE-HARDWARE COMPLEXES

Position Statement

Hermann Kopetz
Technische Universität, Wien, Austria
hk@vmars.tuwien.ac.at

The ever-increasing complexity level of embedded systems, the technology trends of the semiconductor industry to large production series of chips, the new possibilities of the FPGA technology to develop customized hardware of substantial size by software techniques, and the increased competition in the world market entail the need for an integrated development strategy for embedded hardware-software systems.

From a strictly functional point of view, where a user accesses the services of a component, it is irrelevant whether the services of the component are provided by software that is executed on a commercial off-the-shelf computer or by a special hardware-solution (e.g., by a dedicated field-programmable gate array--FPGA). From the non-functional points of view, there are substantial differences: For example, the power efficiency of an FPGA solution is two orders of magnitude better than the power efficiency of a software-on-CPU solution. If we go to a custom hardware solution (i.e., a specially designed chip), the power-efficiency can be improved by another order of magnitude. Power-efficiency is a very important system property in battery-operated devices. Similar differences can be noticed concerning speed and needed silicon area of the implementation of a given function.

These observations suggest that a uniform modeling approach for hardware-software systems is appropriate for the design and implementation of embedded systems. At the Platform-Independent-Model (PIM) level the services of a component should be specified without regard to the chosen implementation platform. Since embedded systems are time-sensitive, this PIM specification must include the temporal properties as well as the value properties of the services. Here we can learn from hardware modeling, since a state-for-the-art hardware specification contains detailed information about a variety of temporal parameters: clock speed, delays, jitter, just to name a few.

We feel that a message-based view is an appropriate level of abstraction for the service specification of a component at the PIM level. A message is an atomic unit that combines the data aspects (value domain) and the timing aspects (instant when a message is sent and instant when a message is received) into a single concept. The emerging field of WEB services provides a good starting point for the specification of component services using this message abstraction. While the message specification in the value domain and in the domain of message exchange patterns are well covered, a precise temporal specification of the message transmission and message arrival (that is needed in embedded systems) is not part of the WEB service specification. WEB services are intended to be used in an open environment (the WEB environment), where precise temporal properties are difficult to guarantee. At present, the precise specification of the temporal properties of message-based services at the PIM level is still an open research issue.

Given that such a precise specification of the message-based component services (in the value and time domain) is available, it is a question of compiler technology to translate this specification into a platform specific model (PSM) for the chosen execution platform, i.e, into a form that can be executed on a CPU (classical software solution) or in a form that can be directly executed on the target hardware (e.g., on an FPGA). Such a model-based approach that distinguishing clearly between the PIM and the PSM has a number of distinct advantages: a given PIM design can be implemented in different implementation technologies (PSMs) without changing the service specification of the component and without a need to retest the system that uses the modified component. For example a new application can be implemented at first in software-on-a-CPU, and the follow up implementation for the mass market can be implemented in an FPGA or a dedicated ASIC chip. Furthermore the issue of technology obsolescence is addressed: if, during the lifetime of the embedding system (e.g., an airplane) the original hardware chips are no longer available, a new version of the service can be implemented on a new hardware base without changing the service specification of the component.