Internal Design Representations for Embedded Systems
Extended Abstract of Embedded Tutorial

Lothar Thiele

Computer Engineering and Networks Lab (TIK)
Swiss Federal Institute of Technology (ETH)
Zurich, Switzerland

Internal Design Representations

Modern embedded computing systems tend to be heterogeneous assemblages of concurrent subsystems, typically described in different languages and semantics which are well established in the various application fields. For example, the specification of the functional and timing behavior necessitates a mixture of different basic models of computation and communication which come from transformative or reactive domains. There is little hope that a single language will replace this heterogeneous set of languages. In fact, experiments with system specification languages show that there is not a unique universal specification language to support the whole life cycle. A similar problem occurs when reused components shall be integrated, possibly described in another language and incompletely documented. Examples would be components of other companies or "legacy code". The lack of coherency of the different languages, methods and tools is a substantial obstacle on the way to higher design productivity and design quality. A design process must be able to bridge the semantic differences for verification and synthesis and should account for limited knowledge of system properties.

The embedded tutorial will describe approaches which allow for the common representation of different languages and incomplete specifications. In particular, recent results on the use of internal design representations targeted to scheduling and design space exploration will be reviewed. Here, the information useful for methods like allocation of resources, partitioning the design and scheduling must be estimated or extracted from the various input languages and mapped onto internal representations which describe properties of the subsystems and their coordination. Design methods work on these internal representations and eventually refine them by adding components and reducing non-determinism.

As an example, we will describe the internal models SPI (System Property Intervals) [8] and FunState [7] which

- support abstraction mechanisms as necessary for the design of complex systems,
- allow global system optimization across the boundaries of different input languages,
- employ behavioral intervals and process modes to allow the common representation of different languages and incomplete specifications, and
- provide constructs to model the representation and selection of function variants of processes and process groups.

Design Space Exploration of Network Processors

As an example, design space exploration and scheduling of network processors will be discussed. The need for intelligent and network packet processing at high data rates, required by many emerging applications, have led to the development of a new class of devices called network processors (NPs). NPs are highly programmable dedicated processors optimized to perform packet processing functions, and will become critical components of next-generation networking equipment.

It is expected that the next generation of network processors will consist of general purpose processing units and dedicated modules for executing run-time extensive functions. Therefore, the purpose of a high level exploration is to select appropriate functional units such that the performance of the processor is maximized under various conflicting constraints.

We introduce an internal design representation for embedded systems operating on packet streams, such as network processors. In particular, the basic internal models described above are extended by the following concepts.

- To investigate the effect of a scheduling algorithm, we represent the "scheduling block" in the form of a network consisting of nodes which operate on event streams. In addition, there are (virtual) resource streams which model the available resources. This way, it is possible to describe and analyze packet scheduling, task scheduling and hierarchical approaches like GPS (weighted fair queuing) and fixed priority schemes, see [5].
- We introduce a calculus meant for reasoning about packet streams which allows for a unified treatment of several problems arising in the network packet processing domain such as packet scheduling, task scheduling and architecture/algorithm explorations, see [6]. The approach unifies the methods commonly used in the domain of communication networks, see [4], with those used in operating systems and real-time scheduling.

To illustrate the potential, we provide a scheme for design space exploration of network processors taking into account conflicting goals such as cost, memory, delay and flow constraints, see also [2].

References

[1] F. Baccelli, G. Cohen, G.J. Olsder, and J.-P. Quadrat. *Synchronization and Linearity*. John Wiley, Sons, New York, 1992.

[2] T. Blickle, J. Teich, and L. Thiele. System-level synthesis using evolutionary algorithms. *Design Automation for Embedded Systems*, 3(1):23–58, 1998.

[3] J.Y. Le Boudec. Application of network calculus to guaranteed service networks. *IEEE Trans on Information theory*, 44(3), May 1998.

[4] R.L. Cruz. A calculus for network delay. *IEEE Trans. Information Theory*, 37(1):114–141, 1991.

[5] L. Thiele, S. Chakraborty, M. Gries, A. Maxiaguine, and J. Greutert. Embedded software in network processors – models and algorithms. *Workshop on Software for Embedded Systems, Lake Tahoe, Lecture Notes in Computer Science*, 2001.

[6] L. Thiele, S. Chakraborty, and M. Naedele. Real-time calculus for scheduling hard real-time systeme. In *Proc. IEEE Internation Conference on Circuits and Systems*, volume 4, pages 101–104, 2000.

[7] L. Thiele, K. Strehl, D. Ziegenbein, R. Ernst, and J. Teich. Funstate— an internal design representation for codesign. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 9(4):524–544, August 2001.

[8] D. Ziegenbein, K. Richter, R. Ernst, J. Teich, and L. Thiele. Representation of process mode correlation for scheduling. In *Proceedings International Conference on ComputerAided Design (ICCAD '98), San Jose, USA*, pages 54–61, 1998.