Predictable Embedded Multiprocessor System Design

Marco Bekooij, Orlando Moreira, Peter Poplavko, Bart Mesman, Milan Pastrnak, and Jef van Meerbergen

1 Philips Research Laboratories, NL-5656 AA Eindhoven, Netherlands
2 Eindhoven University of Technology, NL-5600 MB Eindhoven, Netherlands

Abstract. Consumers have high expectations about the video and audio quality delivered by media processing devices like TV-sets, DVD-players and digital radios. Predictable heterogenous application domain specific multiprocessor systems, which are designed around a networks-on-chip, can meet demanding performance, flexibility and power-efficiency requirements as well as stringent timing requirements. The timing requirements can be guaranteed by making use of resource management techniques and the analytical techniques that are described in this paper.

1 Introduction

Multimedia signal processing and channel decoding in consumer systems is, for performance and power-efficiency reasons, typically performed by more than one processor. The processing in these embedded systems has often stringent throughput and latency requirements. In order to meet these requirements the system must behave in a predictable manner such that it is possible to reason about its timing behavior. The use of analytical methods is desirable because simulation can only be used to demonstrate that the system meets its timing requirements given a particular set of input stimuli. During the design of a multiprocessor system these analytical methods are needed for the derivation of the minimal hardware such that the timing requirements can be met. Given an instance of the multiprocessor system, these methods are needed to program the system in such a way that the timing requirements are met.

A traditional implementation of a predictable multiprocessor system for channel decoding and video processing is a pipe of tightly coupled dedicated hardware blocks with some glue logic for the communication between the blocks. However, there are many reasons to consider this design style as becoming less viable.

First of all, we are currently witnessing the convergence of previously unrelated application domains. For example, ordinary TV-sets are gradually evolving from straightforward terminals to interactive multimedia terminals. Digital auto radios are being combined with navigation systems, and wireless telephone functionality which provides a low bandwidth uplink. DVD players are evolving into DVD writers with complex graphics pipelines that allow to include special effects in our home-brew videos. This convergence leads to heterogeneity and
many options, which all must be supported in a robust and efficient way by the multiprocessor system. The flexibility that is needed in these systems is often more than can be provided with dedicated hardware blocks. The reason is that these blocks are typically designed for one particular function. Also the order of the hardware blocks is fixed in a dedicated hardware pipe while the same blocks could be reused in different applications if the order could be adapted.

Another important reason why a pipe of tightly coupled dedicated hardware blocks becomes a less viable solution, is that the applications become more dynamic as new algorithms seek to take advantage of the disparity between average and worst-case processing. Also the increased interaction with the environment makes systems more dynamic. Often tasks and complex data types are created and deleted at run-time based on non-deterministic events like pressing of a remote control button. This results in many irregularities in the control flow of the application which can be handled effectively by sequential processors but not with dedicated hardware. This dynamism requires more scheduling freedom than can be provided by tightly coupled and synchronized hardware blocks.

The so-called "design productivity gap" states that the increase in our ability to design embedded systems does not match with the exponential growth over time of the number of transistors that can be integrated in an IC as described by Moore’s law. To close the gap, system design methods must harness exponential hardware resource growth. This requires a modular, scalable design style, which is definitely not the case for a pipe of dedicated hardware blocks. The design style must also be composable, which is the case if the correctness (including the timing) of a complete system can be established by using only the (independently proven) features of its components.

The rapid increase of the masks’ cost makes it necessary to design a single System-on-Chip (SoC) for a complete product family. The required flexibility of these SoCs make programmable multiprocessor systems an attractive option. A tradeoff between flexibility and performance must be made due to the power dissipation limit of approximately one 1W of cheap plastic IC-packages and of approximately 100 mW for battery powered devices. So, for power-efficiency reasons these systems will typically contain a mix of dedicated hardware blocks, application domain specific processors, and general purpose microprocessors. It is therefore necessary that the timing analysis techniques are applicable for systems in which application domain specific processors and dedicated hardware blocks are applied that do not support preemption.

A Network-on-Chip (NoC), like the network proposed by the Æthereal project [1], seems a promising option to connect processors with each other. A connection in such a network describes the communication between a master and a slave and such a network can offer differentiated services for these connections. Connections with a guaranteed throughput service which have a guaranteed minimum bandwidth, a maximum latency, a FIFO transaction ordering and end-to-end flow-control are essential for the design of predictable systems as will be explained in the next sections.