Position Paper: Dynamic Reconfiguration in NoC-based MPSoCs in the Avionics Domain

Robert Hilbrich
Fraunhofer FIRST
Berlin, Germany
robert.hilbrich@first.fraunhofer.de

Reinier van Kampenhout
Fraunhofer FIRST
Berlin, Germany
j.r.van.kampenhout@first.fraunhofer.de

ABSTRACT

Modern Network-on-Chip-based Multiprocessor Systems-on-Chip (NoC-based MPSoCs) bear the potential for higher performance, but may also allow the concentration of the same functionality on fewer devices in a complex system such as an aircraft. Albeit these advantages the avionics industry is still hesitant to adopt multi-core technology because software requirements such as predictability have to be met to guarantee safety and reliability. The impact that the application of multi-core processors has on these requirements is not yet fully understood. Therefore our research is driven by the software requirements in the avionics domain which are relevant for the application of multi-cores. We address the dynamic aspects of the system behaviour and investigate flexible partitions and online task migration as a methodology to improve resource utilization on a shared computing platform.

Categories and Subject Descriptors

C.1.4 [**Processor Architectures**]: Parallel Architectures—*Distributed architectures*; D.4.7 [**Operating Systems**]: Organization and Design—*Real-time systems and embedded systems*

General Terms

Design, Reliability, Performance, Measurement

Keywords

Avionics Domain, MPSoC, Multi-core, Network-on-Chip

1. INTRODUCTION

The ongoing trend towards processors with many cores and its effects on the development of software have been widely described. Increased performance is often regarded as the main driver for multi-core adoption. However, the advantages that this technology brings to the domain of (safety critical) embedded systems are often neglected. Besides improved performance, the interest of the avionics industry in multi-cores is also driven by the desire to have simplified architectures. Because modern general purpose processors are very complex and exhibit unpredictable behaviour, there is a significant interest in chips with many, but relatively simple cores so that the effort to pass certification processes can be reduced. Therefore we evaluate the use of MPSoCs to analyze potential advantages for the avionics industry such as improved performance and predictability, lower power usage and production cost, and the concentration of functionality on fewer devices.

To fully exploit the potential of the flexibility inherent to multi-cores, communication has to be acknowledged as a major bottleneck. Therefore software designs have to be *communication-centric* rather than computation-centric. The question arises how to move "computation", such as tasks, to locations where data is waiting and ready to be processed. The flexibility of moving tasks between cores bears many potential advantages for optimizing the overall resource utilization and implementing fault-tolerance. At the same time it creates new challenges which hinder the adoption in safety-critical domains.

We wish to evaluate how the flexibility offered by dynamic reconfiguration in multi- and many-core processors can be exploited to benefit from the advantages mentioned above. It is our immediate goal to determine the effect of task migration on worst-case execution times in real-time systems featuring a NoC-based MPSoC.

2. ON-CHIP INTERCONNECTS

Because technology scaling leads to an increasing gap between interconnection delay and gate delay, communication capacity becomes a performance bottleneck in Systems-on-Chip (SoCs). Bus-based structures are not suitable for on-chip interconnects anymore because they lack scalability, which leads to the use of packet-based Networks-on-Chip (NoCs). Furthermore the communication capacity in SoCs is scarce compared to the available computational power. This impacts software design because the execution time now heavily depends on the time required for communication. In order to reach optimal system performance, tasks need to be scheduled and deployed on cores based on their timing characteristics *and* communication profile.

The Tilera TILE*Pro*64$^{\text{TM}}$is a homogeneous multi-core processor featuring 64 identical cores. The cores are placed on tiles which are connected to each other and to external resources by the iMesh NoC, which consists of six 2D

IWMSE '10, May 1 2010, Cape Town, South Africa

mesh networks and features multi-hop routing. We assume that many-core processors with more than 100 cores will be available in the foreseeable future and believe that the most promising solution to interconnect those cores is the use of NoCs, which have been proven to be scalable. Thus we consider the TILE*Pro*64 processor a suitable and future-proof platform for studying challenges in software development that are encountered when applying MPSoCs in the field of avionics.

3. SOFTWARE REQUIREMENTS IN THE AVIONICS DOMAIN

In the avionics industry functional requirements are increasing and there is demand for more computer based systems. At the same time it is still a safety-critical domain at the highest level with human lives at stake. To save space, weight and power, the avionics industry transitions to an Integrated Modular Avionics (IMA) architecture so that computing resources can be shared and thus used more efficiently. IMA describes computing modules with standardized components and interfaces. Requirements for the operating system are contained in the ARINC 653 specification which among others implements functionality to achieve a *strong partitioning* of software applications on the shared computing platform. Strong partitioning is a way to implement process isolation and is a key requirement for developing reliable safety-critical software. It refers to time and space partitioning with one or more partitions being located on a single IMA module, each containing one or more software tasks.

Aircraft software applications can be distinguished based on the following characteristics. There are *synchronous tasks* which occur periodically and are deterministically scheduled based on fixed scheduling schemes with hyper-periods. Combined with a high criticality level, these tasks are required to exhibit a fully deterministic and predictable behaviour, which has to be proven during the certification process. On the other hand there are *asynchronous tasks* which are event- or data driven and can use additional computational resources to complete as early as possible.

While current IMA boards usually contain a processor with a single core, the avionics industry may benefit from the availability of multi-core processors such as MPSoCs because performance increases and more functionality can be concentrated on fewer devices. Although this may lead to significant savings of power, space and weight, the safety requirements such as redundancy, dissimilarity and predictability have to be satisfied as well. With our work, we want to evaluate the possibility of integrating synchronous and asynchronous tasks on a single MPSoC.

4. DYNAMIC RECONFIGURATION

Our research group already developed a software tool that generates deterministic partition schedules for synchronous tasks on multiprocessor systems so that the development and certification process for avionics software can be simplified. The shortcomings of this approach led to our current research.

In communication-centric architectures not only the computing resources must be scheduled, but also the capacities of the NoC must be reserved beforehand. This avoids communication bottlenecks which lead to congestion and a potential functional failure of the processor. We must also take into account that while cores may be homogeneous with respect to their architecture, the cost to access resources varies because the amount of hops to reach other cores and resources depends on the location. Furthermore the deterministic scheduling approach does not incorporate dynamic aspects of system behaviour like the occurrence of asynchronous tasks, fail-over mechanisms and methods of auto-tuning the system behaviour.

Deterministic mapping may not be absolutely necessary for tasks with a lower criticality level, in fact these might benefit from dynamic reconfiguration at runtime. We believe that combining those with synchronous tasks on one device leads to better overall utilization of the resources. This means however that the challenges of sharing resources in concurrent applications have to be addressed. The meaning of the term *partition* is specified in ARINC 653 as a group of tasks that is granted access to the computing resources of a processor board according to a temporal scheduling scheme. We extend the *spatial* element of the term by additionally assigning cores and communication capacity to a partition. A *fixed* partition consisting of synchronous tasks can thus still be statically mapped and have its resources reserved at design time, while asynchronous tasks may benefit from being contained in a dynamically reconfigurable *flexible* partition whose size and shape can vary over time. To combine both types on a single processor *segregation* is necessary to ensure partition isolation.

The resource requirements of such flexible partitions may vary because of their dynamic character. Temporal variations can occur because new data arrives and needs to be processed, while spatial variations are due to the varying distance to each communication partner (multi-hop paths). These variations can potentially be exploited by reconfiguring the partitions. Computing resources can for instance be temporarily available because some flexible partitions are idle, waiting for data. Another flexible partition could *borrow* these resources in order to finish earlier. A way to exploit spatial variation in resource requirements is to *transform* a partition by reallocating some of its tasks. The goal of this is to decrease its distance to a resource with which it needs interaction in the near future, thus reducing overall network traffic.

Also fault-tolerant systems can benefit from online reconfiguration. Upon detection of a fault measures must be taken to ensure correct operation of the system. Reconfiguration can be applied to avoid usage of the faulty component, or the partition could be relocated completely.

The basic mechanism on which reconfiguration relies is the migration of tasks and data. The overhead inherent to migration must be predictable in safety-critical systems.

Our multi-core lab comprises of a TILEncore^{TM} platform featuring the TILE*Pro*64 processor on which we evaluate a variety of task migration methodologies to analyze task migration overhead on NoC-based MPSoCs. This overhead consists of the time and resource usage that are required for a single migration, and the mechanisms that need to be implemented in software. If the overhead is known and predictable, the feasibility of dynamic reconfiguration for safety-critical real-time systems can be evaluated. Based on the gathered insights we plan to implement a software library which facilitates the use of flexible partitions.