Efficient Implementation of Application-Aware Spinlock Control in MPSoCs

Diandian Zhang, RWTH Aachen University, Aachen, Germany
Li Lu, RWTH Aachen University, Aachen, Germany
Jeronimo Castrillon, RWTH Aachen University, Aachen, Germany
Torsten Kempf, RWTH Aachen University, Aachen, Germany
Gerd Ascheid, RWTH Aachen University, Aachen, Germany
Rainer Leupers, RWTH Aachen University, Aachen, Germany
Bart Vanthournout, Synopsys Inc., Leuven, Belgium

ABSTRACT
Spinlocks are a common technique in Multi-Processor Systems-on-Chip (MPSoCs) to protect shared resources and prevent data corruption. Without a priori application knowledge, the control of spinlocks is often highly random which can degrade the system performance significantly. To improve this, a centralized control mechanism for spinlocks is proposed in this paper, which utilizes application-specific information during spinlock control. The complete control flow is presented, which starts from integrating high-level user-defined information down to a low-level realization of the control. An Application-Specific Instruction-set Processor (ASIP) called OSIP, which was originally designed for task scheduling and mapping, is extended to support this mechanism. The case studies demonstrate the high efficiency of the proposed approach and at the same time highlight the efficiency and flexibility advantages of using an ASIP as the system controller in MPSoCs.

Keywords: Application-Specific Instruction-Set Processor (ASIP), Central Controller, Dynamic Scheduling, Multi-Processor System-on-Chip (MPSoC), Spinlock

MOTIVATION AND INTRODUCTION
Multi-Processor Systems-on-Chip (MPSoCs) are nowadays widely used because of their high performance and power efficiency. Representative examples are TI OMAP processors (Texas Instruments, Inc., 2013), Qualcomm Snapdragon processors (Qualcomm, Inc., 2013) and the ARM MPCore technology (ARM Ltd., 2013). In MPSoCs, applications are typically partitioned into tasks for parallel execution on different processing elements (PEs). To protect shared resources like memories or I/O periph-

DOI: 10.4018/jertcs.2013010104
erals and to prevent data corruption, mutual exclusion needs to be guaranteed. Spinlocks are widely used to ensure that a shared resource can only be accessed by one task at a time.

In literature, many studies have been carried out on the implementation of spinlocks. Most of them focus on the fairness of spinlock acquisition or the reduction of transaction costs due to repeatedly requesting the locks. However, they typically do not consider application knowledge. This leads to a random acquisition order of spinlocks. In many cases, this results in a low parallelism in task execution, and consequently worsens the system performance. A simple example is depicted in Figure 1, in which two PEs compete for the same spinlock. In the example, two different execution sequences of tasks are possible by acquiring the lock in different orders. The execution time shows that assigning the spinlock to \( PE_2 \) first (Figure 1(b)) is a better choice, which achieves a higher performance.

Despite its simplicity, this example shows that a smart control of spinlocks can increase the system performance. To guide the spinlock control in a smart way, application knowledge such as task sizes and task dependencies needs to be considered.

In this work, experiments are done by including application knowledge into the spinlock control flow. As the application knowledge varies from one application to another, it is important to have user-defined spinlock control algorithms, which can be adapted to the application. Instead of developing yet another control unit in the system to execute user-defined algorithms, which will introduce high area cost, they can naturally be implemented on an existing system controller. However, the controller should be programmable, can e.g. be a RISC processor or an Application-Specific Instruction-set Processor (ASIP). This work uses systems with an ASIP controller as the experiment platforms. This ASIP is called OSIP, which stands for Operating System ASIP and was originally designed for task scheduling and mapping.

The remainder of this paper is organized as follows. After discussing about the related work, a brief overview of OSIP-based MP-SoCs is given. Next, an application-aware spinlock control flow in OSIP-based systems is introduced, which starts from the high-level integration of application knowledge down to the low-level realization. Two case studies – a synthetic application and a multimedia application (H.264 video decoding) are performed, which demonstrate the efficiency of the proposed spinlock control mechanism and highlight the advantages of using an ASIP as the system controller. Finally, a summary and an outlook are given.

**RELATED WORK**

A commonly used technique to address mutual exclusion in multi-processor systems is to use spinlocks. Many implementations have been

---

**Figure 1. Impact of spinlock acquisition order**

![Figure 1. Impact of spinlock acquisition order](image-url)
Network Planning and Dimensioning for Broadband Access to the Internet Regarding Quality of Service Demands
www.igi-global.com/chapter/network-planning-dimensioning-broadband-access/21707?camid=4v1a

Integration of Third-Party Applications and Web Clients by Means of an Enterprise Layer
www.igi-global.com/chapter/integration-third-party-applications-web/6469?camid=4v1a