IMORC: An infrastructure for performance monitoring and optimization of reconfigurable computers

Tobias Schumacher, Christian Plessl and Marco Platzner
Paderborn Center for Parallel Computing, University of Paderborn
Fuerstenallee 11, 33102 Paderborn, Germany
Email: {tobe | christian.plessl | platzner }@uni-paderborn.de
Telephone: +49 5251 60-6331

I. Introduction

For many years academic research has studied the use of application-specific coprocessors based on field-programmable gate arrays (FPGAs) to accelerate high-performance computing (HPC) applications. Since major supercomputer vendors now provide servers with integrated reconfigurable accelerators, this technology is available to a much broader group of users. Still, designing an accelerator and optimizing its performance remains a difficult task requiring significant hardware design expertise.

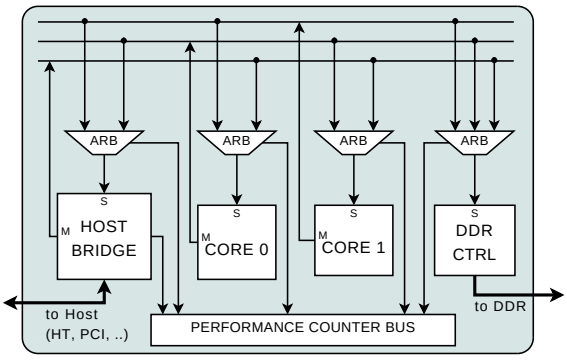
We aim at guiding the accelerator design process with a model-based approach that enables performance optimization throughout the design flow. Our approach uses an application model and an architectural model to estimate the effects of different architectural decisions and varying system parameters [1].

We target applications that are partitioned into communicating software tasks and hardware tasks (cores) that are implemented in reconfigurable hardware. While generally the interaction between the tasks is defined by their data-flow, more formalized models such as process networks, synchronous data-flow graphs, etc., can also be used. Maximizing the performance of an application consisting of many tasks is challenging since the cores affect each other when accessing shared resources. Hence, meticulous care has to be taken to avoid bottlenecks in an implementation.

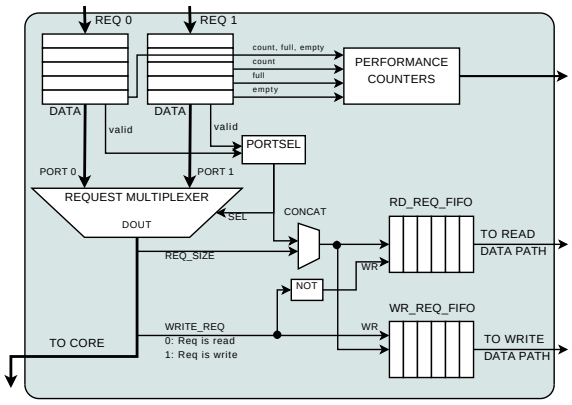
In this paper we introduce a communication infrastructure suitable for implementing and optimizing reconfigurable accelerators in HPC applications. This infrastructure provides a multi-bus interconnection network for connecting hardware cores and provides performance counters for monitoring resource usage at runtime. The collected statistics are used for performance optimization. The infrastructure is not tied to a particular FPGA technology or platform, which facilitates porting applications.

II. Related Work

There exists a number of proprietary core connection standards for FPGAs that are geared towards connecting on-chip peripherals to a CPU core for building a system-on-chip. Some of them, like Xilinx' CoreConnect [2] or Wishbone [3], define versatile buses that support multiple masters sharing the bandwidth available between all cores. Altera's Avalon fabric [4] on the other hand uses multiple buses with slave side arbitration reducing congestion and avoiding complex bus arbitration. Nallatech's DimeTalk [5] allows for building whole networks of interconnected cores that can even span multiple FPGA cards. All these standards solve congestion by equally sharing the bandwidth available. None of these solutions allows for monitoring the execution of the cores at runtime.



(a) Architecture diagram of our infrastructure



(b) Diagram of the request arbiter

III. Core Interconnect and Performance Counter Architecture

IMORC is a multi-bus architecture with slave-side arbitration. Figure 1a presents a block diagram of a two-core instance of our architecture. We pay special attention to provide high throughput by avoiding shared communication resources. To keep the logic resource requirements low the infrastructure is highly parametrized. For example, the width of the communication channels and the depth of FIFOs can be adapted to the actual demand of the application. The architecture is inspired by the Avalon architecture but improves on it in two ways. First, we support customization of the arbiter for optimization and for implementing model-specific scheduling. Second, we provide performance counters for monitoring communication system performance as well as application behavior at runtime.

a) Arbiters: Each core provides a master and multiple slave ports, which can connect to the master ports of other cores using a dedicated arbiter (see Fig. 1b). The request arbiter's *PORTSEL* component selects the input FIFO whose request is processed next. Our architecture allows for changing the arbiters in order to optimize the application performance or to implement different application execution models.

b) Performance counters: Each slave side arbiter contains a certain number of FIFOs for queuing requests, responses and data. To provide a monitoring application with statistics of the accelerator's execution, relevant information, e.g., the FIFO fill level, the duration of empty or full periods, and the latencies between requests and responses, is stored in performance counters. This statistical information is used to identify bottlenecks in the architecture and the application mapping, as well as to extract parameters for the performance modeling process.

IV. Presentation Overview

At the conference we will present our IMORC architecture and demonstrate its usefulness by means of a case-study. We will show a sample application using our infrastructure implemented on an AlphaData ADM-XP FPGA board (Xilinx Virtex 2VP70, PCI-66/64) as well as on an XtremeData XD1000 system (Altera Stratix II, Hypertransport). This demonstrates the portability of our infrastructure and shows how the performance counters aid the designer in finding and removing communication bottlenecks on a specific target architecture.

References

[1] T. Schumacher, E. Lübbers, P. Kaufmann, and M. Platzner, "Accelerating the cube cut problem with an FPGA-augmented compute cluster," in *Proc. of the ParaFPGA Symposium, Int. Conf. on Parallel Computing*, ser. NIC-Series, vol. 38, 2007, pp. 749–756.

[2] *The CoreConnect™Bus Architecture White Paper*, International Business Machines Corporation, Sept. 1999.

[3] *WHISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores, Revision: B.3*, OpenCores.org, Sept. 2002.

[4] *Avalon Memory-Mapped Interface Specification*, Altera®, 2007.

[5] *DIMEtalk User Guide*, Nallatech, Oct. 2005.