

Programming models for reconfigurable heterogeneous multi-cores

Christian Plessl, Marco Platzner, Andreas Agne, Markus Happe, and Enno Lübbers

The ReconOS operating system provides a common, multithreaded model for programming multi-core processors that feature reconfigurable hardware cores as well as general instruction set cores.

A paradigm shift from single-core to parallel multi-core processors has occurred over the last couple of years to further increase the performance of processors. Originating in high-performance computing, this trend has quickly reached general purpose and finally also embedded central processing units (CPUs). Continuing advances in chip manufacture will not only allow an increasing number of homogeneous CPU cores to be integrated in a single chip but will also allow for combining specialized cores to form a heterogeneous multi-core. Many experts believe that such heterogeneous multi-cores will provide advantages in performance and energy efficiency over homogeneous multi-cores, because the different cores can be tailored to specific applications.¹ Although there are established methods for programming homogeneous multi-cores, the programming of heterogeneous multi-cores is still the subject of ongoing research.

In our research, we are particularly interested in heterogeneous multi-cores that include both programmable instruction set processors as well as fixed and reconfigurable function hardware cores. Reconfigurable hardware cores leverage programmable hardware such as field programmable gate arrays. This technology permits the customization of hardware structures to implement highly specialized and efficient fixed-function coprocessors. This reconfiguration process is controlled by software and can occur either once at the startup of the system or even during runtime. Augmenting multi-core processors with reconfigurable cores is attractive, because the reconfigurability allows tailoring of the multi-core to particular applications after fabrication, and can thus be considered an enabling technology for building future self-aware adaptive computer systems.

Programming such heterogeneous multi-cores with reconfigurable cores poses additional challenges. The main challenge

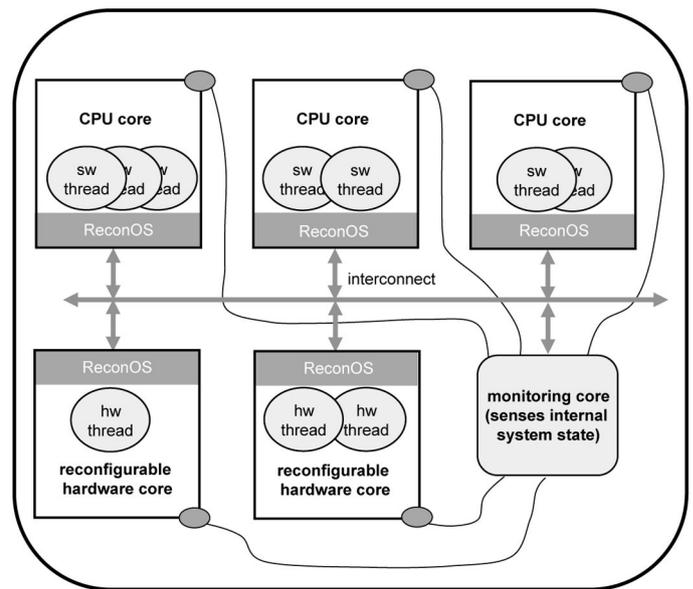


Figure 1. Schematic view of a heterogeneous multi-core processor featuring central processing unit (CPU) cores and reconfigurable hardware cores controlled by the ReconOS reconfigurable operating system. *sw*: Software. *hw*: Hardware.

we address in this work is the software integration of the reconfigurable cores with CPU cores. In contrast to CPU cores, reconfigurable cores are not general purpose instruction set processors that can execute arbitrary code but, once configured, can be considered fixed-function computation units. The typical way of integrating such fixed computational units with the rest of the application running on one or several CPU cores is to demote them to coprocessors that can only act under the direct control of a dedicated master CPU.

In previous work at the University of Paderborn, we have developed the ReconOS reconfigurable operating system.² ReconOS uses a fundamentally different approach to integrate reconfigurable cores with CPU cores. The basic idea of ReconOS is

Continued on next page

to provide a multithreaded programming abstraction that uses threads as the basic unit of computation. This programming model is already widely used for programming homogeneous multi-cores and is thus familiar to software developers. The contribution of ReconOS is to generalize this model to threads that can be executed either on CPU cores (software threads) or on fixed or reconfigurable hardware cores (hardware threads). In other words, instead of restricting reconfigurable cores to being slave coprocessors, ReconOS promotes reconfigurable cores to hardware threads, which act as real peers to software threads executing on the CPUs of the system.

Communication and synchronization of threads is handled entirely by means of well-known operating system functions (for example, message boxes, shared memory areas and semaphores) regardless of whether a thread is a hardware or a software thread. In fact, since communication between threads is only possible through these functions, it is irrelevant for a multithreaded application to know whether a specific thread is implemented in hardware or software. The symmetry between hardware and software threads simplifies the specification and implementation of applications for heterogeneous multi-cores because it allows the developer to start with a pure multithreaded software implementation and to successively replace particular software threads with equivalent hardware threads. Thanks to the consistent operating system interface, no changes other than the switching of software to hardware threads are required. A schematic view of a heterogeneous multi-core with reconfigurable cores is shown in Figure 1.

In recent work, we have presented a video object tracking application that uses the capabilities of ReconOS to maintain quality-of-service requirements.³ The tracker uses a particle filter algorithm that generates a variable computational load depending on the complexity of the scene and the tracked object. Whenever the frame rate of the tracker drops below a user-defined lower threshold, the application uses ReconOS to migrate functionality from software to hardware, to increase the tracking rate. Conversely, functionality is migrated from hardware to software when the frame rate exceeds an upper threshold, to free resources for other hardware threads.

The ReconOS approach not only simplifies the application design process but also opens a variety of novel possibilities that are particularly interesting for future adaptive and self-aware computing systems. For example, if the same function is available as a software and a hardware thread implementation, the decision about where the function will be executed can be deferred until runtime. An encryption task, for instance, could be executed with either a low-performance software

implementation or a high-throughput hardware implementation for a reconfigurable core. If only small amounts of data need to be encrypted in the current system state, the encryption may be mapped to a software thread, whereas the hardware thread may be used when the current state requires encryption of large amounts of data.

This ability to adapt a computing system based on the current and expected user requirements and environmental conditions is an enabling technology for the new class of self-aware and self-expressive computing systems, which are the subject of study in the Engineering Proprioception in Computing Systems (EPiCS) project, a part of the European Union Seventh Framework Programme.⁴ Self-awareness relates to the system's continuous monitoring of its internal state (e.g., resource utilization, temperatures, performance and errors) and the state of the environment (e.g., current user requirements and available energy) for building models of the current and future behaviour of the system. These models form the basis for autonomous decision making to continuously optimize the system's operation under conflicting objectives, such as maximizing performance, minimizing energy consumption and minimizing latency. This adaptation process is known as self-expression.

With ReconOS we have developed a novel operating system for heterogeneous multi-cores comprising CPU cores and reconfigurable cores. The main innovation of ReconOS is to provide a unified, multithreaded programming model for both hardware and software threads, which allows implementation of adaptive applications that move functionality between software and hardware during runtime. Currently, these adaptations are mostly reactive. For example, they are triggered when detecting a certain system state such as a violation of quality of service constraints. In future work we plan to integrate machine learning techniques to make the adaptation process proactive. ReconOS is made available to the research community as open-source and you can download the source code and documentation from the project repository.⁵

The research leading to these results has received funding from the European Union Seventh Framework Programme under grant 257906.

Author Information

Christian Plessl, Marco Platzner, Andreas Agne, and Markus Happe

University of Paderborn
Paderborn, Germany

Christian Plessl is assistant professor of custom computing in the Department of Computer Science at the University of Paderborn, as well as the scientific and technical manager of the European Union Seventh Framework Programme EPiCS project. His current research interests include parallel and reconfigurable computer architectures, hardware-software co-design and self-aware adaptive computing systems.

Enno Lübbers

Intel Labs Europe
Munich, Germany

References

1. K. Asanovic, R. Bodik, B. C. Catanzaro, J. J. Gebis, P. Husbands, K. Keutzer, D. A. Patterson, *et al.*, *The Landscape of Parallel Computing Research: A View from Berkeley*. Tech. Rep. UCB/EECS-2006-183, EECS Department, University of California, Berkeley, Dec. 2006.
2. E. Lübbers and M. Platzner, *ReconOS: Multithreaded Programming for Reconfigurable Computers*, *ACM Trans. Embed. Comput. Syst.* **9** (1), pp. 8:1–8:33, Oct. 2009. doi:10.1145/1596532.1596540
3. M. Happe, E. Lübbers, and M. Platzner, *A self-adaptive heterogeneous multi-core architecture for embedded real-time video object tracking*, *J. Real-Time Image Process.*, pp. 1–16, 2011. doi:10.1007/s11554-011-0212-y
4. <http://www.epics-project.eu>. Engineering Proprioception in Computing Systems (EPiCS) project website. Accessed 8 March 2012.
5. <http://github.com/epics/reconos>. ReconOS project repository of source code and documentation. Accessed 8 March 2012.