Power/Performance Optimization in FPGA-Based Asymmetric Multi-Core Systems

Bruno de Abreu Silva and Vanderlei Bonato

Institute of Mathematical and Computer Sciences, University of Sao Paulo, Brazil
email: {brunoas,vbonato}@icmc.usp.br

Tool Flow Proposal

In this research, we present a work in progress tool to exploit power/performance optimization techniques in FPGA-based asymmetric multi-core systems. The tool flow, which is proposed in Figure 1, is divided in two phases: compilation and execution time. In the first phase, the application code is given as input to the ROSE-based source-to-source compiler[1] which is being implemented to generate three files as output: a file containing application specific hardware parameters (hardware configuration file); a file contains software parameters (software configuration file); and the parallel application code[2].

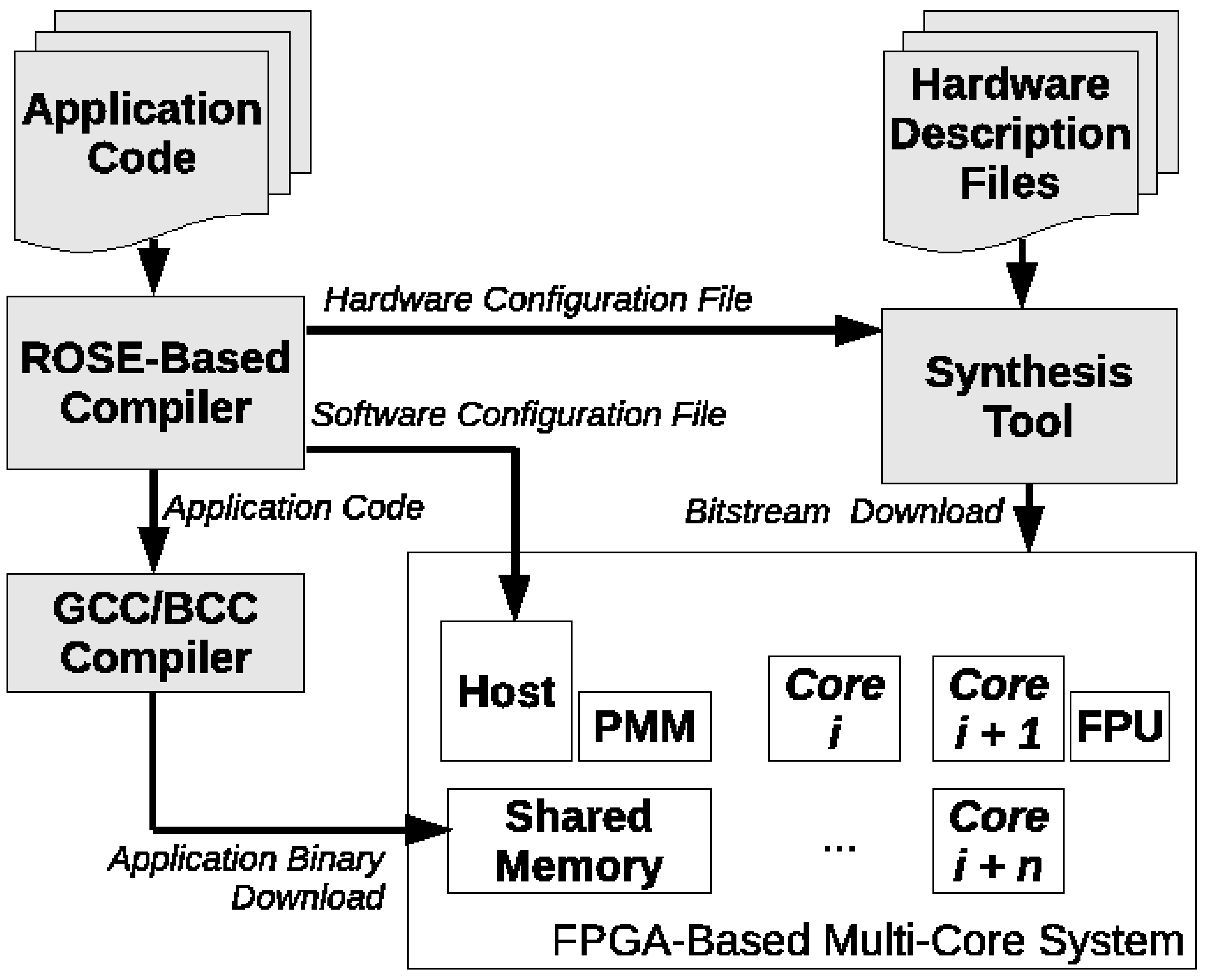
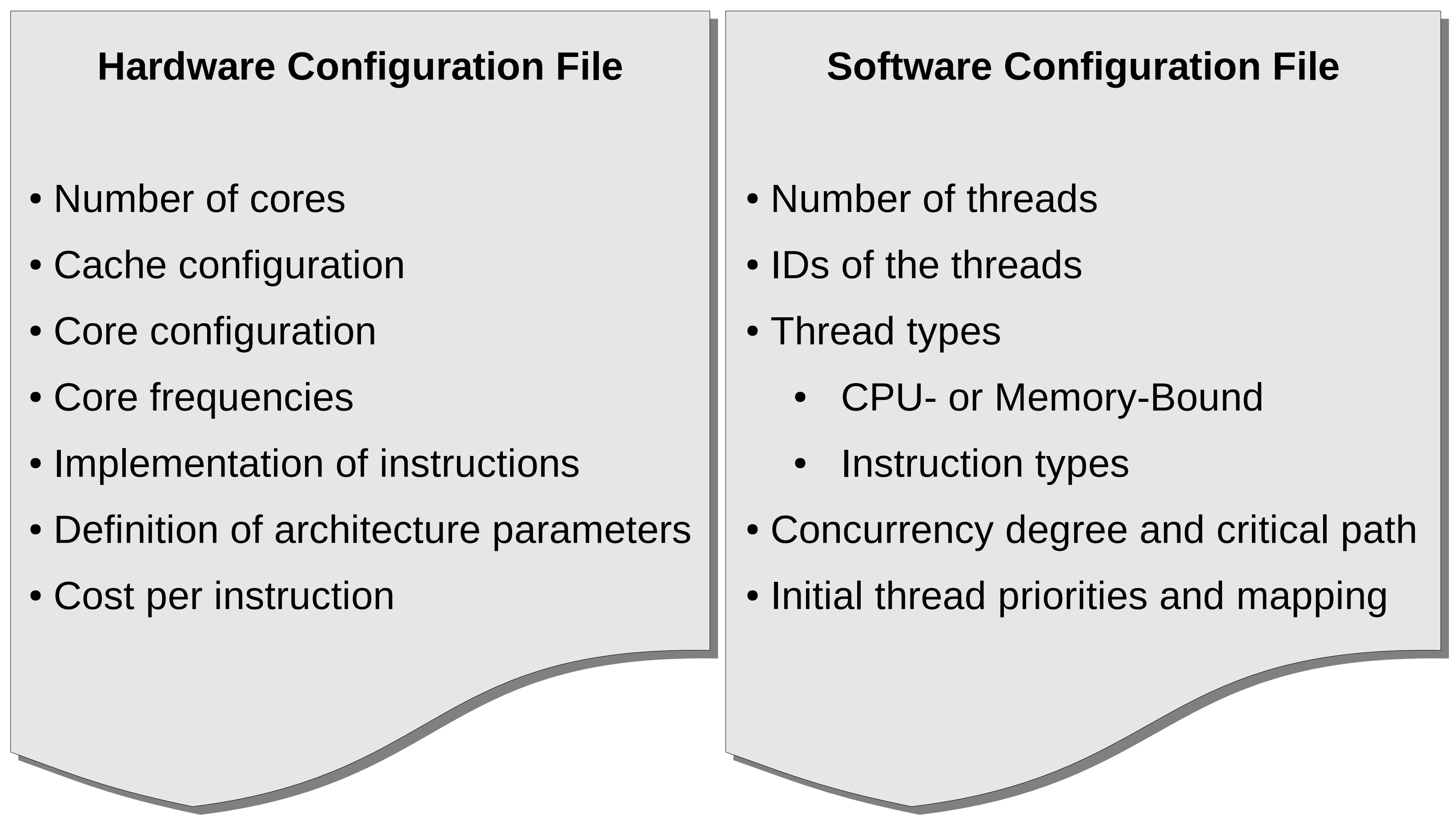


Figure 1: A tool flow proposal, which generates at compilation time the Application Binary Code (the software itself, which is downloaded into the Shared Memory); software mapping and initial thread priorities (Software Configuration File); and hardware optimizations (Hardware Configuration File) for an FPGA-based multi-core system. At execution time, the Power Management Module (PMM) monitors FPGA temperature, power consumption and system performance. This information is given to the scheduler to combine information of both compilation and execution time to make better scheduling decisions and to send commands to the PMM to apply power reduction techniques such as DFS and clock gating.



Hardware Configuration File

- Number of cores
- Cache configuration
- Core configuration
- Core frequencies
- Implementation of instructions
- Definition of architecture parameters
- Cost per instruction

Software Configuration File

- Number of threads
- IDs of the threads
- Thread types
  - CPU- or Memory-Bound
  - Instruction types
- Concurrency degree and critical path
- Initial thread priorities and mapping

Figure 2: Hardware and software configuration files generated at compilation time. The software configuration file can be updated by the OS during the program execution whenever changes are detected by hardware/software monitors.

The hardware configuration file assigns values to hardware parameters. The multi-core system is composed by LEON3 processors[3] and the components, as cores and specialized units, are already implemented. The hardware configuration file together with all hardware description files are synthesized by an EDA tool and the bitstream is downloaded into an FPGA (Stratix IV by Altera).

The software configuration file provides for the eCos scheduler a software mapping and initial thread priorities to run the application on the multi-core architecture. To generate this information, the compiler analyzes the thread profiles. Cores can have different operating frequency domain, cache sizes, specialized implementations of instructions, as core $i + 1$ in Figure 1, which is the only one that has a floating-point unit (FPU) in this example. In this way, the software configuration file says to the scheduler that a given thread that has many floating-point operations should be executed, preferably, in a core featuring an FPU. Having applied the ROSE-based compiler, the application code is then compiled by GCC/BCC and an executable file is downloaded into the shared memory.

During execution time, there are two mechanisms to optimize the relation between power and performance: the operating system scheduler and the PMM (Power Management Module). Both mechanisms are controlled by the Host processor. In this way, the host processor executes the thread scheduling using information provided by the software configuration file. Meanwhile, PMM, which is a hardware module integrated with the operating system, applies DFS and clock gating techniques to the cores individually as needed and also monitors the FPGA temperature, power consumption and performance. The PMM and the scheduler cooperate to each other since the system behavior's information read by the PMM is used by the scheduler to perform task scheduling decisions and to send instructions to the PMM to apply DFS and clock gating.

Conclusion

We believe this work has a significant potential to optimize the relation between power and performance for multi-core platforms. In the proposed tool, new power-aware mapping and scheduling algorithms are explored as well as their integration with the operating system and the power management module. Combining hardware and software configuration files with run-time techniques provides further opportunities, since in a reconfigurable FPGA-based platform it's possible to align the hardware with the application and to adjust the application in run-time according to the software behavior.

Acknowledgements

The authors would like to thank the FAPESP (the Foundation for Support Research of the State of Sao Paulo) for the financial support provided.

References

[1] M. D. Hill and M. R. Marty. 2008. Amdahl's law in the multicore era. *IEEE COMPUTER*. vol. 41. 33–38.

[2] H. Ishebabi and C. Bobda. Automated architecture synthesis for parallel programs on FPGA multiprocessor systems. *Microprocessors and Microsystems*. 2009. vol. 33. no. 1. 63–71.

[3] S. Zhuravlev, J. C. Saez, S. Blagodurov, A. Fedorova and M. Prieto. Survey of Energy-Cognizant Scheduling Techniques. *IEEE Trans. Parallel Distrib. Syst.* 2012.

[1] ROSE compiler infrastructure (http://www.rosecompiler.org/)
[2] The eCos operating system has been used to run on the multi-core system (http://ecos.sourceware.org/)
[3] http://www.gaisler.com/