Architecture Exploration of Parameterizable EPIC SOC Architectures*

Ashok Halambi Radu Cornea Peter Grun Nikil Dutt Alex Nicolau

Architectures and Compilers for Embedded Systems (ACES) Laboratory
Center for Embedded Computer Systems
University of California, Irvine, CA 92697-3425, USA
http://www.cecs.uci.edu/~aces

1 Introduction

Design Space Exploration (DSE) of programmable systems-on-chip (SOC) incorporating parameterizable processor cores is difficult due to the complex and intrinsically non-structured interactions between different architectural features of the processor (such as wide parallelism, and deep pipelines), the compiler and the application. Changing different processor features implies generating detailed operation conflict information – represented as Reservation Tables (RTs). If done manually, it can be a very tedious and error prone task, especially for deep pipelines, with complex resource sharing and large non-structured instruction sets. In this paper we use RTGEN[2], an approach for automatic generation of RTs, to drive rapid architectural exploration of a large number of designs. We present exploration experiments on a large set of VLIW-like EPIC[1] architectures, for varying port sharing, number of functional units, multicycling units, and with varied latency configurations. Our experiments uncovered several non-intuitive architecture design points, giving the system-level designer further flexibility in exploration of programmable SOC architectures.

2 HPL-PD EPIC Architecture

We use the HPL PlayDoh (HPL-PD)[4] – a parametric EPIC load/store architecture with both VLIW and Superscalar features – as the platform for investigating the coupling of processor architecture and compiler technology. For EPIC-style architectures, Register File (RF) ports are a critical resource, motivating the need for exploring the relationship between RF port sharing and performance.

We perform architecture exploration experiments by modifying the multi-cycle latencies of Functional Units (FUs) and by sharing RF ports between FUs. With each such architectural configuration we also vary the number of FUs (from 4 to 10) in order to study its impact on performance.

3 Experimental setup

Figure 1 presents the flow of our experimental setup. Starting from an architectural description in an ADL, RTGEN generates the set of RTs in MDES format. The MDES files are used by Trimaran[5] to generate code and simulate the application for the specified architecture. The simulation results are then presented to the user, who can decide what architectural features to change in the ADL representation.

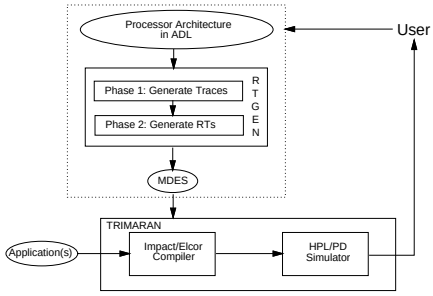

Figure 1. The flow of our experimental setup

4 Exploration Results

By specifying (and modifying) HPL-PD using an ADL (EXPRESSION[3]) and using RTGEN to automatically generate the RTs for the different architecture versions, we were able to rapidly evaluate a large set of architectures. Due to the large number of architectures, other approaches requiring manual specification of RTs would have been prohibitively complex and probably not feasible. Furthermore, we observed several non-intuitive design configurations that were obtained due to our ability to allow interaction between the compiler, the architecture and the application. Detailed experimental results are in the full paper [1] and can be accessed from: http://www.cecs.uci.edu/~aces

References

[1] R. Cornea, A. Halambi, P. Grun, N. Dutt, and A. Nicolau. Compiler-architecture exploration using reservation tables generation. Technical Report 99-31, University of California, Irvine, 1999.

[2] P. Grun, A. Halambi, N. Dutt, and A. Nicolau. RTGEN: An algorithm for automatic generation of reservation tables from architectural descriptions. In *Proc. ISSS*, 1999.

[3] A. Halambi, P. Grun, V. Ganesh, A. Khare, N. Dutt, and A. Nicolau. EXPRESSION: A language for architecture exploration through compiler/simulator retargetability. In *Proc. DATE*, March 1999.

[4] V. Kathail, M. Schlansker, and B. Rau. HPL PlayDoh architecture specification: Version 1.0. Technical Report HPL-93-80, HP Labs, 1994.

[5] Trimaran Release: http://www.trimaran.org.

*This work was partially supported by grants from NSF (MIP-9708067) and ONR (N00014-93-1-1348).

[1] Explicitly Parallel Instruction Computing