ReCode: The Design and Re-design of the Instruction Codes for Embedded Instruction-Set Processors

Clifford Liem[1,2], Pierre Paulin[2], Ahmed Jerraya[1]

1. Laboratoire TIMA, INPG, Grenoble, France
2. SGS-Thomson Microelectronics, Crolles, France

There are many attractive reasons for the inclusion of an embedded processor in today's world of rapidly changing standards [1]. However, when a product finally hits its target window and is on the market, a designer often ponders on how well he made the design decisions in the creation of his/her instruction-set processor. In many cases, this is further motivated by a decision to redesign a low-cost version of the product which is already on the market. The designer is then faced with the challenge of better fitting the processor to the existing application code.

This abstract presents a design aid called *ReCode*, and describes its use in the analysis of existing instruction-set processors. ReCode allows the exploration of the relationship between the instruction-set and the corresponding application code of embedded processors. After analyzing the instruction-set and code, the designer can then use the set of editing functions to adjust the instruction set to the application code.

ReCode is a window-based tool which has two modes of analysis: static and dynamic. In the static mode, the tool allows the user to analyze the correspondence between the use of assembly codes and the compiled application code. This allows the designer to identify codes which have poor utilization and make changes accordingly. As the tool works with an instruction-set specification which can be regenerated, the compiler is consequently retargeted automatically for any changes.

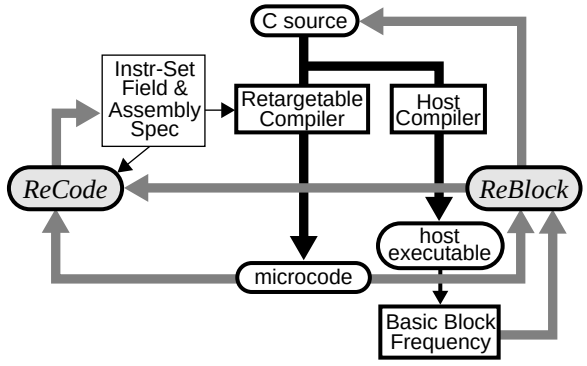

Figure 1. ReCode instruction analysis & ReBlock profiling: tools for analyzing application code.

A second tool, ReBlock serves as a retargetable profiler. When used independently, ReBlock links the information of a profiling execution of the application on the host with the microcode produced by the retargetable compiler. This information is then attached to the source code and shown in a window-based form. The tool then allows estimation of real-time performance on the level of blocks selected by the user [2].

In the dynamic mode, ReCode communicates with the ReBlock profiler to analyze the dynamic use of assembly codes. This can either be done on the level of blocks directly in the C code to identify bottlenecks or on a global level to identify hot and cold spots of the machine.

The tools have been tested on a set of existing instruction-set processors at SGS-Thomson Microelectronics including a videophone controller (H.261/263) and an audio decoding DSP (MPEG2, AC-3, Dolby Pro-logic). For each of the case studies, ReCode was able to extract application characteristics which correspond to how each processor was used. For example, regarding the videophone controller, characteristics of the control-flow codes showed where improvement in either the hardware functionality or optimization sequences in the compiler could be made.

For the audio decoding architecture, the tool was able to extract distributions on the usage of hardware resources such as register files and busses. This could serve as a methodology to reorganize the available data-flow transfers in the hardware, or as a basis for improving register allocation in the compiler.

Current work involves studies using the ReBlock profiler to allow the user to test the effect of branch penalties on application code. Given a pipelined penalty model, the user is able to explore different possibilities such as changing the number of instruction cycles which constitute a possible branch penalty.

[1] P. Paulin et al., , "Trends in Embedded Systems Technology: An Industrial Perspective", in Hardware/Software Co-design, ed. by M. Sami, G. DeMicheli, Kluwer Academic Publishers, 1996.

[2] L. Bergher et. al., "MPEG Audio Decoder for Consumer Applications", *Custom Integrated Circuits Conference*, May 1995.

ED&TC '97 on CD-ROM

Permission to make digital/hard copy of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for fee or commercial advantage, the copyright notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee. © 1997 ACM/0-89791-849-5/97/0003/$3.50