Compiler Techniques for Very Long Instruction Word Embedded Processors

Abstract

As applications become more sophisticated, there is a push toward the adoption of high-level languages for use in development of embedded software. This aids in the development process by allowing for code which is much more portable, easier to maintain and more straightforward to review. On the downside, compiler generated code lacks the compactness and speed which can be achieved by assembly language written by a highly skilled programmer. Although compiler optimization techniques have advanced to reduce this gap, room for improvement still exists.

We will evaluate a very long instruction word (VLIW) processor. A VLIW processor can perform multiple instructions on multiple data elements within a single clock cycle. Additionally, scheduling is performed only by the compiler at compile time. The Texas Instruments TMS320C6201 is a fixed point VLIW DSP processor capable of two multiplies, two shifts, two address calculations, and two adds per clock cycle through two separate data paths. Texas Instruments provides an optimizing C compiler for TMS320C62x processors with the following standard optimization features: branch optimizations / control flow simplification, alias disambiguation, copy propagation, common sub-expression elimination, redundant assignment elimination, loop induction variable optimizations / strength reduction, loop rotation, loop invariant code motion, inline expansion of function calls, file level optimizations, data flow optimizations,

expression simplification, register variables, register tracking / targeting, and cost-based register allocation.

Goals

We intend to evaluate the current state of the gap in efficiency between the Texas Instruments compiler and hand generated assembly language code, and suggest improvements to the compiler. In general this evaluation will be in the area of vectorization operations as this is the only class of optimizations which are exclusive to VLIW architectures. In order to achieve this end, we must become very familiar with the TMS320C62 architecture. We will also become familiar with general compiler optimization techniques specific to VLIW.

Tools

We will be using the Texas Instruments compiler and simulator available at UT. Ptolemy will be used to model the test cases as well as generate C code for them.

References

A. W. Appel, *Modern Compiler Implementation in C*, Cambridge University Press, ISBN 0-521-58390-X, 1998.

D. Bursky, "VLIW architecture offers a ten-fold improvement in digital signal processing. (TMS320C6201 from Texas Instruments Inc.)," *Electronic Design*, p. 52, March 1997.

J. Davidson and S. Jinturkar, "Improving Instruction-level Parallelism by Loop Unrolling and Dynamic Memory Disambiguation," In *MICRO-28: 28th Annual International Symposium on Microarchitecture*. Ann Arbor, MI, December 1995.

A. E. Eichenberger and E. S. Davidson, "Register Allocation for Predicated Code," *Proc. IEEE of the International Sumposium on Microarchitecture*, 1995.

S. Liao, S. Devadas, K. Keutzer, S. Tjiang, and A. Wang, "Code optimization techniques for embedded DSP microprocessors," *Proceedings of the 32nd ACM/IEEE conference on Design automation conference*, p. 599, 1995.

S. A.Mahlke, W. Y.Chen, R. A.Bringmann, R. E.Hank, W. W. Hwu, B. Ramakrishna Rau, and M. S.Schlansker, "Sentinel scheduling a model for compiler-controlled speculative execution," *ACM Transactions on Computer Systems*, vol. 11, no. 4, pp. 376-408, November 1993.

S. A.Mahlke, W. Y.Chen, W. W. Hwu, B. Ramakrishna Rau, and M. S. Schlansker, "Sentinel scheduling for VLIW and Superscalar Processors," *Proceedings of the fifth international conference on Architectural support for programming languages and operating systems*, pp. 238 - 247, 1992.

C. Norris and L. Polluck, "An Experimental Study of Several Cooperative Register Allocation and Instruction Scheduling Strategies," *Proc. IEEE of the International Sumposium on Microarchitecture*, 1995.

D. Pountain, "The word on VLIW," *Byte*, April 1996, pp. 61-62.

Ptolemy Project, *The Almagest: A Manual for Ptolemy*, Ptolemy Project (http://ptolemy.eecs.berkeley.edu/papers/almagest/index.html), 1997.

O. Sharp, "Compilers for parallel CPUs," *Byte*, pp. 97-99, February 1994.

M. Silverthorn, L. Adams, and R. Scales, "Guidelines for Software Development Efficiency on the TMS320C6000 VelociTI Architecture," Texas Instruments, April 1998, White Paper: SPRA434.

Texas Instruments, "TMS320C6x Code Development Flow," Texas Instruments, February 1999, Application Report: SPRA518.

Texas Instruments, *Tutorial on TMS320C6000 VelociTI Architecture*, 1998.

Texas Instruments, *TMS320C62x/C67x Programmer's Guide*, Texas Instruments, February 1998.

Texas Instruments, *TMS320C6000 Optimizing C Compiler User's Guide*, Texas Instruments, July 1998.