Review

A Review of RISC and Forth Machine Literature

Lawrence P. Forsley
Laboratory for Laser Energetics
University of Rochester

Recently considerable attention has been given to the development of a Reduced Instruction Set Computer, or RISC, which may alleviate many of the design problems and instruction execution bottlenecks associated with modern computers. Forth may be a member of the RISC class of machines. This paper examines current RISC and Forth machine literature, draws parallels, and notes differences among the efforts.

Reduced Instruction Set Computers

Computer architectures have increasingly tended toward direct hardware execution of high level languages (HLL). For example, Intel's iAPX-432 chip has been referred to as an Ada computer. These computers, like the Digital Equipment Corporation (DEC) VAX, tend to have fully orthogonal instruction sets and execute instruction sequences commonly produced by Fortran and other compilers. Recently, this trend towards more complicated computer architectures has been resisted by the development of the Reduced Instruction Set Computer.

The term RISC was coined at the University of Berkeley [PATT82] for a computer chip designed to efficiently implement procedural languages, such as C and Pascal [SEQU83]. In particular, these designs were optimized to reduce the overhead of procedure calls, which were estimated to be upwards of 40% of program execution time [PATT82]. One method employed in the Berkeley RISC I and RISC II chips was an overlapped register window which precluded the necessity of copying parameters during a procedure call.

Another aspect of the RISC architecture is simplicity. The Motorola 68000 is noted as having approximately 68,000 transistors, and the 32-bit RISC I has 45,000; many of which support a large (78) register file [PATT82] [WILK82]. A simplified architecture is necessary to deal with the realities of VLSI design: limited power dissipation, limited intra- and inter-chip communication and debugging difficulties. A RISC computer is also optimized towards relatively few instructions which execute at high speed. Taken together, this results in a chip where the simple control logic occupies only 6% of the chip area (as opposed to 50% in some VLSI chips) and isn't microcoded [PATT82], thereby leaving room for the register file. RISC I had approximately 31 instructions and limited addressing modes: index plus displacement, and by hardwiring register 0 to all 0's, absolute and register indirect [SEQU83].

Interestingly, simulated RISC I HLL benchmarks indicate speeds exceeding those of a 10 MHz Motorola 68000 [PATT82] and the DEC VAX 780 computer [LARU82] [PATT82]. Several manufacturers, most notably Ridge Computers [FOLG83] and Pyramid, have since developed commercial RISC-like computers.

Castan and Organick described an even simpler RISC machine for evaluating functional programming languages [CAST82]. They envisage the development of a Direct Executable

Language which is not linear, but is list structured, and is known as a "3L-model" where 3L refers to Lisp Like Language. Both the 3L form and μ3L processor for executing it are similar to the tree structures and processor described by Vaughan and Smith in this *Journal* issue. Although LISP was the first language implemented for a 3L processor, a Pascal environment is under development.

The μ3L processor is vertically microprogrammed and has only 10 instructions. It contains a register stack with fewer than 256 32-bit wide entries, and all of its data paths are 32 bits wide. The authors expect to use a network of μ3L processors sharing some common memory to independently execute the 3L lists.

Forth

Although Forth currently has little parameter passing overhead, unlike most procedural languages, the Forth interpreter follows the RISC premise of a simple hardware design and provides for efficient execution of a HLL (Forth). However, the introduction of named parameters on the stack and frames of local storage [GLAS83] would increase parameter passing significantly. There have been no reports of native Forths running on conventional RISCs for comparison, nor has a VLSI Forth chip appeared. Indeed, prior to this issue of the *Journal* there have been very few papers on Forth hardware implementations.

In the mid-1970's Lorenson, of the European Southern Observatory, published a technical note describing a microcoded version of Forth for the Hewlett Packard 21MX computer [LORE77]. Rust presented a 16-bit bit-slice implementation, supported by a Zilog Z-80 in an S-100 evironment, at the 1981 Rochester Forth Standards Conference [RUST81]. Winkel described another bit-slice implementation in Forth Dimensions [WINK81]. More recently, Wada and Kaneda, from Kobe University in Kobe, Japan, published 4 papers on a bit-slice implementation similar to Rust's.

Their first two papers describe the machine's internals [WADA82a] and its performance [WADA82b]. The machine has a hardware 4K x 16 bit return stack and dual 8K x 16 bit parameter stacks for simultaneous access to TOS and TOS-1. The machine uses the Z-80 to manage the dictionary and execute the Forth text interpreter. Two additional papers by these authors discuss the execution of Forth and Pascal code [KANE83] and microcoding of the Pascal P machine interpreter [WADA83]. They report that Forth executes twice as fast as Pascal on their machine [KANE83], and that the Forth is over 100 times faster than a 2 MHz Z-80 running polyForth and about 50 times faster than a DEC LSI-11 running polyForth [WADA82b]. Their machine had a variable clock cycle of either 175 or 350 nsec.

Not surprisingly, most of the Forth machines published in the literature to date have been based upon Advanced Micro Devices bit-slice technology. The bit-slice approach offers a simplified building block approach to architecture and reasonably fast execution. However, it is likely that the bit-slice Forth machines are more complex than necessary, reflecting the use of general purpose commercial bit-slice devices. Much of the interest in a VLSI version of Forth stems from process control and telecommunications applications where costs preclude using a bit-slice version. Unfortunately a bit-slice Forth design will be significantly different from a VLSI design.

It is interesting to note that most of the conclusions drawn by Patterson, Castan and others have been from RISC simulators, as very few RISC devices had been built by 1983. With the increasing interest in a VLSI Forth, it would be reasonable to apply the same RISC simulation techniques to Forth, which would provide more realistic performance expectations than have previously been available. However, we need to study the static (e.g. memory utilization) and dynamic (word interpretation) execution of Forth first, which will require the development of Forth code analyzers.

Acknowledgements

This work was supported by the U.S. Department of Energy Office of Inertial Fusion under contract DE-AC08-80DP40124 and by the Laser Fusion Feasibility Project at the Laboratory for Laser Energetics which has the following sponsors: Empire State Electric Energy Research Corporation, General Electric Company, New York State Energy Research and Development Authority, Northeast Utilities Service Company, Southern California Edison Company, The Standard Oil Company, and University of Rochester. Such support does not imply endorsement of the content by any of the above parties. This work was also supported by a research contract with Digital Equipment Corporation.

I would like to thank Kathleen Wilkenson for the computerized library searches, and John Cassady for providing references to Wada's first two papers.

Bibliography

[CAST82] Castan, M.; Organick, E. I., (Dept. of Computer Sci., Univ. of Utah, Salt Lake City, UT) "Micro 3L: An HLL-RISC Processor for Parallel Execution of FP-Language Programs", *Proceedings of the 9th Annual Symposium on Computer Architecture*, IEEE, New York, NY, 1982, pp. 239–247.

[FOLG83] Folger, D. and Basart, E., (Ridge Computers, Sunnyvale, CA) "Computer Architectures — Designing for Speed", *Digest of Papers Spring COMPCON 83, Intellectual Leverage for the Information Society*, IEEE, New York, NY 1983, pp. 25-31.

[GLAS83] Glass, H. (University of South Florida, Tampa, FL) "Towards a More Writable Forth Syntax", *Proceedings of the 1983 Rochester Forth Applications Conference*, Institute for Applied Forth Research, Inc., Rochester, NY 1983, pp. 125–132.

[KANE83] Kaneda, Yukio; Wada, Koichi; and Maekawa, Sadao, (Faculty of Engineering, Kobe University, Kobe, Japan), "High-Speed Execution of Forth and Pascal Programs on a High-Level Language Machine", Wilson, D. R. and van Spronsen, C. J. (eds), *MICROCOMPUTERS: Developments in Industry, Business and Education*, Elsevier Science Publishers B. V., 1983, pp. 259–266.

[LARU82] Larus, J. R., (Dept. of Electrical Eng. and Computer Sci., Univ. of California, Berkeley, CA) "A Comparison of Microcode, Assembly Code, and High-Level Languages on the VAX-11 and RISC I", *Computer Archit. News* 4, Vol. 10, No. 5, Sept. 1982, pp. 10–15.

[LORE77] Lorenson, Svend, (European Southern Observatory), "A Microcoded Forth for the HP21MX Computer", *ESO Technical Note*, Garsching, West Germany, 1977.

[PATT82] Patterson, D. A. and Piepho, R. S., (Dept. of Electrical Eng. and Computer Sci., Univ. of California, Berkeley, CA) "RISC Assessment: A High-Level Language Experiment", *Proceedings of the 9th Annual Symposium on Computer Architecture*, IEEE, New York, NY, 1982, pp. 26–29.

[RUST81] Rust, T., (Dave Nutting Associates, Arlington Heights, Illinois), "ACTION Processor FORTHRIGHT", *Proceedings of the 1981 Rochester Forth Standards Conference*, Institute for Applied Forth Research, Inc., Rochester, NY 1981, pp. 309–315.

[SEQU83] Sequin, C. H. and Patterson, D. A. (Computer Sci. Div., Univ. of California, Berkeley, CA) "Design and Implementation of RISC I", (Randell, B. and Treleaven, P. C., editors), *VLSI Architecture, 1982 Advanced Course on VLSI*, Prentice-Hall, Englewood Cliffs, NJ, 1983, pp. 276–298.

[WADA82a] Wada, Koichi; Kaneda, Yukio; and Maekawa, Sadao (Kobe University, Kobe, Japan) "System Design and Hardware Structure of a Forth Machine System", *Systems, Computers, Controls*, Vol. 13, No. 2, Scripta, Silver Spring, MD, 1982, pp. 19–28. Translated from Denshi Tsushin Gakkai Ronbunshi, Vol. 54-D, No. 3, March 1982, pp. 338–345.

[WADA82b] Wada, Koichi; Kaneda, Yukio; and Maekawa, Sadao (Kobe University, Kobe, Japan) "Software and System Evaluation of a Forth Machine System", *Systems, Computers, Controls*, Vol. 13, No. 2, Scripta, Silver Spring, MD, 1982, pp. 11–18. Translated from Denshi Tsushin Gakkai Ronbunshi, Vol. 54-D, No. 3, March 1982, pp. 346–353.

[WADA83] Wada, Koichi; Nakamatsuju, Toshiyuki; Kaneda, Yukio; and Maekawa, Sadao, (Kobe University, Kobe, Japan) "Realization and Evaluation of Pascal Machine on a High-Level Language-Oriented Stack Machine" *Systems, Computers, Controls*, Vol. 14, No. 4, Scripta, Silver Spring, MD 1983, pp. 1-9. Translated from Denshi Tsushin Gakkai Ronbunshi, Vol. 66-D, No. 4, pp. 369-376.

[WILK82] Wilkes, M. V. (Digital Equipment Corp., Hudson, MA) "Keynote Address, The Processor Instruction Set", *Proceedings of the 15th Annual Workshop on Microprogramming*, IEEE, New York, NY, 1982. pp. 3-5.