Trimaran ILP Reading List By Topical Category

This is not intended to be a comprehensive and exhaustive bibliography of papers in the area of instruction-level parallel processing (ILP). Rather, this list concentrates on those papers that best demonstrate the EPIC architectural philosophy advocated by HPL-PD and the compiler philosophy, framework and techniques embodied in Trimaran. This reading list is designed for people who wish to get up to speed on the use of HPL-PD and Trimaran, and who wish to further the state of the art of compiling for EPIC architectures.

A good starting point is to understand HPL-PD [1], the way in which the architectural features of HPL-PD were intended to be used [2], and the space of EPIC processors that can be described to, and compiled by, Trimaran [3, 4]. Thereafter, one could focus on those topics listed below that represent one's areas of research interest.

ILP survey

A survey of instruction-level parallel processing as of 1992 [5].

VLIW architecture

Commercial VLIW products [6-9].

EPIC architecture

The original specification of the EPIC style of architecture [1] and a discussion of the underlying philosphy as well as the motivation for the various architectural features [2]. Partial details of the IA-64, the first commercial instance of an EPIC architecture [10].

ILP compiler systems

ILP compilers of significant scope [11-14].

ILP optimizations

Various optimizations that have special relevance to an ILP processor [11, 15, 16, 12-14, 17, 18].

Acyclic scheduling

Trace scheduling [19, 13, 20], superblock scheduling [12], hyperblock scheduling [21], more general scheduling algorithms [22-24], and inter-region scheduling [11, 13, 25].

Modulo scheduling

Modulo scheduling of DO-loops [26-28, 2]. Modulo scheduling of WHILE-loops and loops with early exits [29, 30]. Modulo scheduling of loops with control flow, without the use of predication [31, 32]. Register-pressure sensitive modulo scheduling [33-36]. A survey of software pipelining techniques [37].

Array variable promotion and expanded virtual registers (EVRs)

Array variable promotion to eliminate array loads and stores that are redundant within or across iterations of the loop [38-40, 14]. The following papers also discuss the value flow analysis required [39, 14, 41] and the use of expanded virtual registers (EVRs) to avoid the premature introduction of register-register copy operations [39, 14].

Register allocation

Rotating register allocation [42]. Register allocation for predicated code [43, 44].

If-conversion and the use of predicates

If-conversion to form predicated code [26, 45, 14]. The application of predicates: an overview [2], in modulo scheduling [26, 27], in scheduling acyclic regions [21, 46, 47], in reducing the length of the critical path through the computation [48, 49], and as an intermediate form when scheduling control-intensive regions for a processor without predicated execution [50]. The benefits of predicated execution in modulo scheduling [51] and in acyclic regions [52].

Analysis of predicated code

Analysis of predicated code [53]. The application of such analyses to register allocation [43, 44].

Control speculation and recovery

Control speculation, i.e., the movement of operations to an earlier point than the branch or predicate-setting operation upon which it is dependent [19, 54, 46, 2, 47] and architectural and compiler techniques to support correct exception handling [55, 54, 56, 2].

Data speculation

Data speculation, i.e., the movement of a load to an earlier point than a store upon which it might be dependent [57-59, 2].

Tolerating branch latency

"Unbundled" branches [2]. Static branch prediction [60, 61]. Programmatically-computed dynamic branch prediction [62].

Critical path reduction (CPR)

Compiler transformations to reduce the length of the critical path through a computation [63, 48, 49, 64].

Tolerating data cache miss latency

Predicting which memory references will cause data cache misses and either prefetching them or scheduling them early [65] [66] [67] [68] [2].

Machine description

The machine description database and query interface used by the ILP compiler to understand the relevant details of the target processor [3, 4, 69]. Optimization of the machine description [70, 71].

Region-based compiling

An approach to structuring ILP compilers which is embedded in, and supported by, Trimaran [18].

Note: A number of the HPL Technical Reports in this reading list have been published as journal articles or in conference proceedings. The technical reports were listed because they are more complete and comprehensive than the corresponding papers. However, for the benefit of individuals who would like to read the paper, and for the purposes of referencing

these works, we list here pairs of references. In each pair, the first reference is the technical report and the second one is the corresponding paper:

[5], [72]
[27], [73]
[28], [72]
[49], [74]
[53], [75]
[48], [76]

Bibliography

1. V. Kathail, M. Schlansker and B. R. Rau. HPL-PD Architecture Specification: Version 1.1. Technical Report HPL-93-80 (R.1). Hewlett-Packard Laboratories, July 1998.
2. M. Schlansker, B. R. Rau, S. Mahlke, V. Kathail, R. Johnson, S. Anik and S. G. Abraham. Achieving High Levels of Instruction-Level Parallelism with Reduced Hardware Complexity. HPL Technical Report HPL-96-120. Hewlett-Packard Laboratories, February 1997.
3. J. C. Gyllenhaal, W.-m. W. Hwu and B. R. Rau. HMDES Version 2.0 Specification. Technical Report IMPACT-96-3. University of Illinois at Urbana-Champaign, 1996.
4. S. A. Gupta, V. Kathail and B. R. Rau. Elcor's Machine Description System: Version 3.0. HPL Technical Report HPL-98-128. Hewlett-Packard Laboratories, July 1998.
5. B. R. Rau and J. A. Fisher. Instruction-Level Parallel Processing: History, Overview and Perspective. Technical Report HPL-92-132. Hewlett-Packard Laboratories, October 1992.
6. R. P. Colwell, R. P. Nix, J. J. O'Donnell, D. B. Papworth and P. K. Rodman. A VLIW architecture for a trace scheduling compiler. IEEE Transactions on Computers C-37, 8 (August 1988), 967-979.
7. G. R. Beck, D. W. L. Yen and T. L. Anderson. The Cydra 5 mini-supercomputer: architecture and implementation. The Journal of Supercomputing 7, 1/2 (May 1993), 143-180.
8. Trimedia TM-1 Media Processor Data Book. (Philips Semiconductors, Trimedia Product Group, 1997).
9. TMS320C62xx CPU and Instruction Set Reference Guide. (Texas Instruments, 1997).
10. C. Dulong. The IA-64 architecture at work. Computer 31, 7 (July 1998), 24-32.
11. J. R. Ellis. Bulldog: A Compiler for VLIW Architectures. (The MIT Press, Cambridge, Massachussetts, 1985).
12. W. W. Hwu, S. A. Mahlke, W. Y. Chen, P. P. Chang, N. J. Warter, R. A. Bringmann, R. G. Ouellette, R. E. Hank, T. Kiyohara, G. E. Haab, J. G. Holm and D. M. Lavery. The superblock: an effective technique for VLIW and superscalar compilation. The Journal of Supercomputing 7, 1/2 (May 1993), 229-248.
13. P. G. Lowney, S. M. Freudenberger, T. J. Karzes, W. D. Lichtenstein, R. P. Nix, J. S. O'Donnell and J. C. Ruttenberg. The Multiflow trace scheduling compiler. The Journal of Supercomputing 7, 1/2 (May 1993), 51-142.
14. J. C. Dehnert and R. A. Towle. Compiling for the Cydra 5. The Journal of Supercomputing 7, 1/2 (May 1993), 181-228.
15. W. W. Hwu and P. P. Chang. Inline function expansion for compiling realistic C programs. Proc. SIGPLAN '89 Conference on Programming Language Design and Implementation (Portland, Oregon, June 1989), 246-257.

16. S. A. Mahlke, N. J. Warter, W. Y. Chen, P. P. Chang and W. W. Hwu. The effect of compiler optimizations on available parallelism in scalar programs. Proc. 20th Annual International Conference on Parallel Processing (St. Charles, Illinois, August 1991), 142-145.

17. D. M. Lavery and W. W. Hwu. Unrolling-based optimizations for software pipelining. Proc. 28th Annual International Symposium on Microarchitecture (Ann Arbor, Michigan, November 1995), 327-337.

18. R. E. Hank, W. W. Hwu and B. R. Rau. Region-based compilation: Introduction, motivation, and initial experience. International Journal of Parallel Programming 25, 2 (April 1997), 113-146.

19. J. A. Fisher. Trace scheduling: a technique for global microcode compaction. IEEE Transactions on Computers C-30, 7 (July 1981), 478-490.

20. S. M. Freudenberger, T. R. Gross and P. G. Lowney. Avoidance and suppression of compensation code in a trace scheduler. ACM Transactions on Programming Languages and Systems 16, 4 (July 1994), 1156-1214.

21. S. A. Mahlke, D. C. Lin, W. Y. Chen, R. E. Hank and R. A. Bringmann. Effective compiler support for predicated execution using the hyperblock. Proc. 25th Annual International Symposium on Microarchitecture (1992), 45-54.

22. K. Ebcioglu and A. Nicolau. A *global* resource-constrained parallelization technique. Proc. 3rd International Conference on Supercomputing (Crete, Greece, June 1989), 154-163.

23. S.-M. Moon and K. Ebcioglu. An efficient resource-constrained global scheduling technique for superscalar and VLIW processors. Proc. 25th Annual International Symposium on Microarchitecture (Portland, Oregon, December 1992).

24. J. A. Fisher. Global Code Generation for Instruction-Level Parallelism: Trace Scheduling-2. Technical Report HPL-93-43. Hewlett-Packard Laboratories, June 1993.

25. S. Abraham, V. Kathail and B. Deitrich. Meld scheduling: relaxing scheduling constraints across region boundaries. Proc. 29th Annual IEEE/ACM International Symposium on Microarchitecture (Paris, France, December 1996), 308-321.

26. J. C. Dehnert, P. Y.-T. Hsu and J. P. Bratt. Overlapped loop support in the Cydra 5. Proc. Third International Conference on Architectural Support for Programming Languages and Operating Systems (Boston, Mass., April 1989), 26-38.

27. B. R. Rau, M. S. Schlansker and P. P. Tirumalai. Code Generation Schemas for Modulo Scheduled DO-Loops and WHILE-Loops HPL-92-47. Hewlett Packard Laboratories, April 1992.

28. B. R. Rau. Iterative Modulo Scheduling. HPL Technical Report HPL-94-115. Hewlett-Packard Laboratories, November 1995.

29. P. Tirumalai, M. Lee and M. S. Schlansker. Parallelization of WHILE Loops on Pipelined Architectures. The Journal of Supercomputing 5 (1991), 119-136.

30. D. M. Lavery and W. W. Hwu. Modulo scheduling of loops in control-intensive non-numeric programs. Proc. 29th Annual IEEE/ACM International Symposium on Microarchitecture (Paris, France, December 1996), 126-137.

31. N. J. Warter, J. W. Bockhaus, G. E. Haab and K. Subramanian. Enhanced modulo scheduling for loops with conditional branches. Proc. The 25th Annual International Symposium on Microarchitecture (Portland, Oregon, December 1992), 170-179.

32. M. G. Stoodley and C. G. Lee. Software pipelining loops with conditional branches. Proc. 29th Annual IEEE/ACM International Symposium on Microarchitecture (Paris, France, December 1996), 262-273.

33. R. A. Huff. Lifetime-sensitive modulo scheduling. Proc. SIGPLAN '93 Conference on Programming Language Design and Implementation (Albuquerque, New Mexico, June 1993), 258-267.

34. J. Llosa, M. Valero, E. Ayguadé and A. Gonzalez. Hypernode reduction modulo scheduling. Proc. The 28th Annual International Symposium on Microarchitecture (Ann Arbor, Michigan, November 1995), 350-360.

35. A. E. Eichenberger and E. S. Davidson. Stage scheduling: a technique to reduce the register requirements of a modulo schedule. Proc. 28th Annual International Symposium on Microarchitecture (Ann Arbor, Michigan, November 1995), 338-349.

36. J. Llosa, M. Valero and E. Ayguade. Heuristics for register-constrained software pipelining. Proc. 29th Annual IEEE/ACM International Symposium on Microarchitecture (Paris, France, December 1996), 250-261.

37. V. H. Allan, R. B. Jones, R. M. Lee and S. J. Allan. Software pipelining. Computing Surveys 27, 3 (September 1995), 367-432.

38. D. Callahan, S. Carr and K. Kennedy. Improving register allocation for subscripted variables. Proc. SIGPLAN '90 Conference on Programming Language Design and Implementation (June 1990), 53-65.

39. B. R. Rau. Data flow and dependence analysis for instruction level parallelism, in Fourth International Workshop on Languages and Compilers for Parallel Computing, U. Banerjee, D. Gelernter, A. Nicolau and D. Padua (Editor). (Springer-Verlag, 1992), 236-250.

40. E. Duesterwald, R. Gupta and M. L. Soffa. A practical dataflow framework for array reference analysis and its use in optimizations. Proc. SIGPLAN '93 Conference on Programming Language Design and Implementation (Albuquerque, New Mexico, June 1993), 68-77.

41. R. Bodik and S. Anik. Path-sensitive value flow analysis. Proc. 25th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'98) (San Diego, California, January 1998), 237-251.

42. B. R. Rau, M. Lee, P. Tirumalai and M. S. Schlansker. Register Allocation for Software Pipelined Loops. Proc. SIGPLAN'92 Conference on Programming Language Design and Implementation (San Francisco, June 17-19 1992).

43. A. E. Eichenberger and E. S. Davidson. Register allocation for predicated code. Proc. 28th Annual International Symposium on Microarchitecture (Ann Arbor, Michigan, November 1995), 180-191.

44. D. M. Gillies, D.-c. R. Ju, R. Johnson and M. Schlansker. Global predicate analysis and its application to register allocation. Proc. 29th Annual IEEE/ACM International Symposium on Microarchitecture (Paris, France, December 1996), 114-125.

45. J. C. H. Park and M. S. Schlansker. On predicated execution. Technical Report HPL-91-58. Hewlett-Packard Laboratories, Palo Alto CA, May 1991.

46. D. I. August, W. W. Hwu and S. A. Mahlke. A framework for balancing control flow and predication. Proc. 30th Annual IEEE/ACM International Symposium on Microarchitecture (Research Triangle Park, North Carolina, December 1997), 92-103.

47. D. I. August, D. A. Connors, S. A. Mahlke, J. W. Sias, K. M. Crozier, B.-C. Cheng, P. R. Eaton, Q. B. Olaniran and W. W. Hwu. Integrated predicated and speculative execution in the IMPACT EPIC architecture. Proc. 25th Annual International Symposium on Computer Architecture (Barcelona, Spain, June 1998), 227-237.

48. M. Schlansker, V. Kathail and S. Anik. Parallelization of control recurrences for ILP processors HPL-94-75. Hewlett-Packard Laboratories, Palo Alto CA, August, 1994 1994.

49. M. Schlansker and V. Kathail. Techniques for Critical Path Reduction of Scalar Programs HPL-95-112. Hewlett-Packard Laboratories, Palo Alto CA, 1995.

50. N. J. Warter, S. A. Mahlke, W. W. Hwu and B. R. Rau. Reverse if-conversion. Proc. SIGPLAN '93 Conference on Programming Language Design and Implementation (Albuquerque, New Mexico, June 1993), 290-299.

51. N. J. Warter, D. M. Lavery and W. W. Hwu. The benefit of predicated execution for software pipelining. Proc. 26th Annual Hawaii International Conference on System Sciences (Hawaii, January 1993), 497-506.

52. S. A. Mahlke, R. E. Hank, R. A. Bringmann, J. C. Gyllenhaal, D. M. Gallagher and W. W. Hwu. Characterizing the impact of predicated execution on branch prediction. Proc. 27th International Symposium on Microarchitecture (San Jose, California, November 1994), 217-227.

53. R. A. Johnson and M. S. Schlansker. Analysis of Predicated Code. Technical Report HPL-96-119. Hewlett-Packard Laboratories, December 1996.

54. S. A. Mahlke, W. Y. Chen, R. A. Bringmann, R. E. Hank, W. W. Hwu, B. R. Rau and M. S. Schlansker. Sentinel scheduling: a model for compiler-controlled speculative execution. ACM Transactions on Computer Systems 11, 4 (November 1993), 376-408.

55. K. Ebcioglu. Some design ideas for a VLIW architecture for sequential-natured software, in Parallel Processing (Proc. IFIP WG 10.3 Working Conference on Parallel Processing, Pisa, Italy), M. Cosnard, M. H. Barton and M. Vanneschi (Editor). (North Holland, Amsterdam, 1988), 3-21.

56. D. I. August, B. L. Deitrich and S. A. Mahlke. Sentinel Scheduling with Recovery Blocks. Technical Report CRHC-95-05. Center for Reliable and High-Performance Computing, University of Illinois at Urbana-Champaign, February 1995.

57. W. Y. Chen, S. A. Mahlke, W. W. Hwu, T. Kiyohara and P. P. Chang. Tolerating data access latency with register preloading. Proc. 1992 International Conference on Supercomputing (Washington, D. C., July 1992), 104-113.

58. G. M. Silberman and K. Ebcioglu. An architectural framework for supporting heterogeneous instruction-set architectures. Computer 26, 6 (June 1993), 39-56.

59. D. M. Gallagher, W. Y. Chen, S. A. Mahlke, J. C. Gyllenhaal and W. W. Hwu. Dynamic memory disambiguation using the Memory Conflict Buffer. Proc. Sixth International Conference on Architectural Support for Programming Languages and Operating Systems (San Jose, California, October 1994), 183-195.

60. W. W. Hwu, T. M. Conte and P. P. Chang. Comparing software and hardware schemes for reducing the cost of branches. Proc. 16th Annual International Symposium on Computer Architecture (May 1989), 224-233.

61. J. A. Fisher and S. M. Freudenberger. Predicting conditional jump directions from previous runs of a program. Proc. Fifth International Conference on Architectural

Support for Programming Languages and Operating Systems (Boston, Mass., October 1992), 85-95.

62. S. Mahlke and B. Natarajan. Compiler synthesized dynamic branch prediction. Proc. 29th Annual IEEE/ACM International Symposium on Microarchitecture (Paris, France, December 1996), 153-164.

63. M. Schlansker and V. Kathail. Acceleration of first and higher order recurrences on processors with instruction level parallelism. Proc. Sixth Annual Workshop on Languages and Compilers for Parallel Computing (Portland Oregon, 1993).

64. M. S. Schlansker, S. A. Mahlke and R. A. Johnson. Bypassing the Branch Bottleneck Using Control Critical Path Reduction. Technical Report HPL-98-?? Hewlett-Packard Laboratories, (to appear) September 1998.

65. S. G. Abraham, R. A. Sugumar, D. Windheiser, B. R. Rau and R. Gupta. Predictability of load/store instruction latencies. Proc. 26th Annual International Symposium on Microarchitecture (December 1993), 139-152.

66. S. G. Abraham and B. R. Rau. Predicting Load Latencies Using Cache Profiling. Technical Report HPL-94-110. Hewlett-Packard Laboratories, November 1994.

67. F. J. Sanchez and A. Gonzalez. Cache sensitive modulo scheduling. Proc. 30th Annual IEEE/ACM International Symposium on Microarchitecture (Research Triangle Park, North Carolina, December 1997), 338-348.

68. T. C. Mowry and C.-K. Luk. Predicting data cache misses in non-numeric applications through correlation profiling. Proc. 30th Annual IEEE/ACM International Symposium on Microarchitecture (Research Triangle Park, North Carolina, December 1997), 314-320.

69. B. R. Rau, V. Kathail and S. A. Gupta. Machine-Description Driven Compilers for VLIW Processors. HPL Technical Report HPL-98-40. Hewlett-Packard Laboratories, March 1998.

70. J. C. Gyllenhaal, W.-m. W. Hwu and B. R. Rau. Optimization of machine descriptions for efficient use. Proc. 29th Annual IEEE/ACM International Symposium on Microarchitecture (Paris, France, December 1996), 349-358.

71. A. E. Eichenberger and E. S. Davidson. A reduced multipipeline machine description that preserves scheduling constraints. Proc. SIGPLAN'96 Conference on Programming Language Design and Implementation (Philadelphia, Pennsylvania, May 1996), 12-20.

72. B. R. Rau. Iterative modulo scheduling. International Journal of Parallel Processing 24, 1 (February 1996), 3-64.

73. B. R. Rau, M. S. Schlansker and P. P. Tirumalai. Code generation schemas for modulo scheduled loops. Proc. 25th Annual International Symposium on Microarchitecture (Portland, Oregon, December 1992), 158-169.

74. M. S. Schlansker and V. Kathail. Critical path reduction for scalar programs. Proc. 28th Annual International Symposium on Microarchitecture (Ann Arbor, Michigan, November 1995), 57-69.

75. R. Johnson and M. Schlansker. Analysis techniques for predicated code. Proc. 29th Annual IEEE/ACM International Symposium on Microarchitecture (Paris, France, December 1996), 100-113.

76. M. S. Schlansker, V. Kathail and S. Anik. Height reduction of control recurrences for ILP processors. Proc. 27th Annual International Symposium on Microarchitecture (San Jose, California, November 1994), 32-39.