The CAE Architecture: Decoupled Program Control for Complexity-Effective Performance.

Ronny Krashinsky and Mike Sung
6.893 Project Report (checkpoint 1)
MIT Laboratory for Computer Science, Cambridge, MA 02139
{ronny,darkman}@mit.edu

Abstract

Monolithic superscalar architectures will not scale into the next era of computer architecture. Their design is based on structures with a high degree of connectivity that will not be available in future chips in which a clock cycle covers a tiny fraction of the area. Their performance is based on reckless speculation that will not be tolerated in future complexity-effective designs. The processor of the future is composed of many decoupled elements working independently but in collaboration. As went the supercomputers, so will the superprocessors.

A promising next-generation architecture has been demonstrated in decoupled access/execute machines. These processors have split apart the memory access and execution portions of a program, and thus have immediately exposed a large amount of ILP. By allowing these streams to slip relative to each other, these machines enjoy the benefits of out-of-order execution and memory latency hiding with very little overhead. Additionally, the queues which connect these decoupled elements together provide the benefits of register renaming without the complexity required in superscalar architectures.

This work presents decoupled control flow, the next step which will enable processors of the future to reach new levels of performance. In a decoupled control/access/execute (CAE) machine, a control processor runs ahead and feeds directives to the memory access processor and the main execution processor; the directives are in the form of commands to execute basic blocks. The execution engine is then responsible for processing streams of valid instructions and data values, obtained without the overhead of speculation. This is a fundamental departure from the model in which an execution engine must actively fetch instructions and data values, or speculate to hide latency. As a result, new levels of performance are obtainable.

References

[1] Wm. A.Wulf. Evaluation of the WM computer architecture. *journal*, 0.

[2] Wm. A.Wulf. The WM computer architecture. *Computer Architecture News*, 16(1):???, March 1988.

[3] E. Rotenberg *et. al.* A study of control independence in superscalar processors. *journal*, 0.

[4] E. Rotenberg *et. al.* Trace processors. *journal*, 0.

[5] James E. Smith *et. al.* The astronautics zs-1 processor. *journal*, 0.

[6] M. Farrens, P. Ng, and P. Nico. A comparison of superscalar and decoupled access/execute architectures. *journal*, 0.

[7] L. Gwennap. Mips r10000 uses decoupled architecture. *journal*, 0.

[8] P. T. Hulina, L. Kurian, E. B. John, and L. D. Coraor. Design and vlsi implementation of an access processor for a decoupled architecture. *journal*, 0.

[9] L. K. John, A. Subramanian, P. T. Hulina, and L. D. Coraor. Improving the parallelism and concurrency in decoupled architectures. *journal*, 0.

[10] L. Kurian, P. T. Hulina, and L. D. Coraor. Memory latency effects in decoupled architectures. *journal*, 0.

[11] J. E. Smith. Dynamic instruction scheduling and the astronautics zs-1. *IEEE Computer*, 22(7):21–35, July 1989.

[12] James E. Smith. Decoupled access/execute computer architecture. In *ISCA 9*, 1982.

[13] J. Tubella and A. Gonzalez. Control speculation in multithreaded processors through dynamic loop detection. *journal*, 0.

[14] G. Tyson and M. Farrens. Code scheduling for multiple instruction stream architectures. *journal*, 0.

[15] G. Tyson, M. Farrens, and A. Pleszkun. Misc: A multiple instruction stream computer. *journal*, 0.