Author Retrospective

Improving Data Cache Performance by Pre-executing Instructions Under a Cache Miss

Trevor Mudge
University of Michigan
Ann Arbor, MI
tnm@eecs.umich.edu

ABSTRACT

The paper introduces and evaluates a technique, referred to as runahead, that prefetches instructions and data into the L1 caches on cache misses. The results of the experiments reported in the paper show that the CPI of a simple in-order pipeline could be reduced by 30%. The principle advantage of runahead over many other prefetching strategies, is twofold: 1) it prefetches both instructions and date; and 2) it is able to create non-sequential data prefetches.

Original Paper: http://dx.doi.org/10.1145/263580.263597

Categories and Subject Descriptors

C.0 [**Computer Systems Organization**]: General; C.1.1 [**Computer Systems Organization**]: Processor Architectures

Keywords

runahead; speculation; prefetching

1. INTRODUCTION

The work presented in the paper arose from a conversation that Jim Dundas and I had while he was a graduate student at The Univerisity of Michigan in the mid-1990's. We were brainstorming ideas that would improve the performance of a simple in-order pipeline with a minimum of extra logic. The driving factor was that we had been designing and building prototype Gallium Arsenide processors, which suffered from reduced device density compared to silicon [8, 1]. This led us to consider a number of unorthodox architectural ideas, including the topic of this paper: runahead pre-processing. The basic idea behind runahead is to allow a simple, yet very fast, processor pipeline to prefetch and pre-process instructions during cache miss cycles, instead of stalling. The pre-processed instructions are used to generate highly accurate data prefetches. Runahead allowed us to achieve a form of aggressive speculation with a simple in-order pipeline. Important from our point of view, the principal hardware cost is modest, because we make use of the execution pipeline when it would otherwise be idle. The only additional cost is a means of checkpointing the sequential state of the register file and memory hierarchy while instructions are pre-processed. This can be achieved with just a backup register for every register in the register file (simpler than a duplicate register file), because we discard all results that are computed during runahead episodes after the cache miss has been serviced.

ICS 25th Anniversary Volume. 2014
ACM 978-1-4503-2840-1/14/06.
http://dx.doi.org/10.1145/2591635.2591655.

2. BACKGROUND

In the early 1990's our group had been experimenting with Gallium Arsenide (GaAs) as an alternative to silicon. GaAs had the promise of improved performance through increased clock speed because it has a much higher electron mobility than silicon. It also had the promise of lower power—our prototypes ran with a supply voltages of 1-2V when the prevailing silicon supply voltages were 5V (or later in the decade, 3.3V). The drawback with GaAs was that the densities were an order of magnitude less than silicon. This still allowed us to build single chip microprocessors, and, as noted above, we built several MIPS-like cores in the mid-1990's, but those cores had to be relatively simple. This limited the range of architectural alternatives that we could explore to take advantage of this faster technology.

We experimented with a number of ideas that didn't affect gate count too much. We explored much deeper pipelines [7] than used at that time, as well as pipelined cache organizations [6]. The runahead idea also arose from exploring ways to improve performance without substantially increasing gate count. We viewed it as an inexpensive way to retrieve some of the architectural benefits that we had to give up because of logic density limitations. Our early experiments were reported in [3]. Subsequently Jim wrote his Ph.D. [4] thesis on the subject and explored many of the alternatives to the straightforward approach presented in the paper. In particular, he evaluated the idea of removing last level caches and just relying on runahead to maintain performance. The results were mixed and highly dependent on L1 size and benchmarks. He also showed that runahead works well for streaming benchmarks, as might be expected. In the case of STREAM, CPI was reduced by almost 80%. He fur-

ther examined simpler checkpointing schemes and proposed the idea of a co-processor to perform runahead.

Subsequent work on runahead took a different turn. Rather than viewing it as an inexpensive way to obtain architecture benefits in density constrained designs, runahead was examined in the context of more complex out-or-order machines. Several efforts took this direction. Examples are the work of Mutlu and Patt and their collaborators [5], and the work of Chaudhry and his colleagues at Sun on the ROCK processor [2]. The Sun design used threads to lookahead, coining the term "scout" thread, to describe their lookahead mechanism. The related work section in this paper contains a good overview of related work, if the reader wants to expire more recent developments.

3. REFERENCES

[1] R. Brown, T. Basso, P. Parakh, S. Gold, C. Gauthier, R. Lomax, and T. Mudge. Complementary GaAs technology for a ghz microprocessor. In *Gallium Arsenide Integrated Circuit (GaAs IC) Symposium, 1996. Technical Digest 1996., 18th Annual*, pages 313–316, Nov 1996.

[2] S. Chaudhry, R. Cypher, M. Ekman, M. Karlsson, A. Landin, S. Yip, H. Zeffer, and M. Tremblay. Simultaneous speculative threading: A novel pipeline architecture implemented in Sun's ROCK processor. In *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ISCA '09, pages 484–495, New York, NY, USA, 2009. ACM.

[3] J. Dundas and T. Mudge. Using stall cycles to improve microprocessor performance. Technical Report CSE-TR-301-96, Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan, September 1996.

[4] J. D. Dundas. *Using Stall Cycles to Improve Microprocessor Performance*. PhD thesis, The University of Michigan, Ann Arbor, Michigan, 1998.

[5] O. Mutlu, J. Stark, C. Wilkerson, and Y. Patt. Runahead execution: an alternative to very large instruction windows for out-of-order processors. In *High-Performance Computer Architecture, 2003. HPCA-9 2003. Proceedings. The Ninth International Symposium on*, pages 129–140, Feb 2003.

[6] K. Olukotun, T. Mudge, and R. Brown. Performance optimization of pipelined primary cache. In *Proceedings of the 19th Annual International Symposium on Computer Architecture*, ISCA '92, pages 181–190, New York, NY, USA, 1992. ACM.

[7] M. Upton, T. Huff, T. Mudge, and R. Brown. Resource allocation in a high clock rate microprocessor. In *Proceedings of the Sixth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS VI, pages 98–109, New York, NY, USA, 1994. ACM.

[8] M. Upton, T. Huff, P. Sherhart, P. Barker, R. McVay, T. Stanley, R. Brown, R. Lomax, T. Mudge, and K. Sakallah. A 160 000 transistor GaAs microprocessor. In *Solid-State Circuits Conference, 1993. Digest of Technical Papers. 40th ISSCC., 1993 IEEE International*, pages 92–93, Feb 1993.