# Memory Decoupled Architectures and related issues
# Guest Editor's Introduction

Roberto Giorgi
Department of Information Engineering
University of Siena, Italy
giorgi@acm.org
http://www-dii.ing.unisi.it/~giorgi

It is my great pleasure to serve as guest editor for this special issue of TCCA Newsletter, which is hosting eight papers from the MEDEA (MEmory DEcoupled Architectures) Workshop, jointly held with PACT-2000 conference. The rationale behind this workshop was to revive the original idea of Memory Access Decoupling, presented in the famous paper of Jim Smith, "Decoupled Access/Execute Architectures" [1]. In that paper a novel architecture was proposed, as emerging among high performance architectures appearing in the industrial scenario (CDC Cyber 180/990, CSPI array processor) and the academy (Illinois SMA). At that time, Jim Smith came back to the University of Wisconsin to fuel his ideas. The main concept in Memory Access Decoupling was to use two instruction streams and queues to separate memory accesses and pure computations [2].

After about 20 years the scenario of high performance microprocessors is quite changed. Superscalar and VLIW architectures are the dominant paradigms, and a variety of tricks are used to enhance the performance or reduce the consumption. Instruction Level Parallelism, Out-of-Order Execution, Speculative Loads, Branch Prediction, Multithreading, Chip Multiprocessors, Dynamic Compilation, are just some of the keywords entered in our common vocabulary. In this new scenario we asked for contributions that could show how memory decoupling could be applied to achieve design goals, and possibly to explore new sources of parallelism. As observed by Roth, Zilles, and Sohi, in this issue first paper, today's processors can tolerate latencies of about 10 cycles, but we are approaching the case where the processor-memory gap is going to exceed 100 cycles. So, at this time, it is only clear that we do have some space to research new solutions.

The first paper is from the University of Wisconsin. It introduces a novel approach to decoupling, called Data-Driven Multi-Threading (DDMT). Since the latency for accessing memory is mainly due to misses, DDMT decouples the miss stream into multiple sub-streams that are speculatively executed as data-driven threads (DDT). The main stream and the DDTs are executed in parallel using Simultaneous Multi-Threading (SMT) model [3]. The approach provides significant speedup on most of the analyzed programs.

The paper from University of Southern California introduces the Hierarchical Decoupled Instruction Stream Computer (HiDISC). In this architecture, decoupled prefetching allows for improved memory system performance. The compiler generates three kinds of instructions (ALU, Load/Store, and Cache-Management instructions) that are executed by three separate processors at three different levels of the memory hierarchy. Decoupled execution permits to fetch data before higher-level processors request it.

The paper from University of Alabama in Huntsville (UAH), describes an interesting paradigm which puts together dataflow and multithreaded architecture concepts, named Scheduled Data-Flow (SDF). First, the high-level code is compiled in relatively small threads. Three portions constitute each thread: load, execute, and store portion. Then, the architecture facilitates the scheduling of the three decoupled portions by means of simple specialized hardware. The paper shows that this architecture does not exhibit pipeline hazards and has the potentials to perform better than traditional Superscalar processors.

In the paper from University of Notre Dame the authors explore decoupling of memory accesses on Simultaneous Multithreading (SMT) architecture. Instead of undistinguished threads, like in the original idea, the decoupled memory, compute or "mixed" threads are extracted and allocated on Memory, Memory/Exe, or Only-Execute functional units in a SMT fashion. They use code-slicing techniques and show that significant possibilities exist in the SPEC-95 benchmark suite for exploiting this idea.

Previous papers focus on processor support to decoupling. The following papers concentrate on other support to memory decoupling and related issues. A joint work from UAH and Washington University in St. Louis addresses the decoupling through a memory manager. The case is for an intelligent memory device, which embeds the Segregated Binary Tree algorithm, in order to facilitate garbage collection and compaction for Java and C++ programs. The evaluation includes programs from the JVM-98 benchmark suite.

The papers from the Computer Architecture group at the University of Pisa include a contribution from S. Bartolini, and C. A. Prete regarding decoupling a through software strategy to reduce cache misses. They achieve a miss reduction up to 70% for embedded system applications and systems with a requirement of a small cache. The strategy uses heuristics, which are tuned by a link-time profiling of traces and a consequent optimal code placement.

The paper from South Brittany University inspects low-power problems in functionally decoupled architectures like DSPs. Their approach introduces a high level power estimation technique that achieve satisfying accuracy while maintaining a relative independence from the particular instruction set. The technique focuses on a behavioral analysis of the DSP architecture.

The paper from Polytechnic University of Valencia and University of Belgrade examines decoupling of accesses to two independent cache organizations: classical first level cache and filter data cache. Filter data cache maintains the most heavily referenced blocks. The two caches are accessed in parallel by a Superscalar processor thus achieving a better performance than traditional split cache schemes.

Another original contribution from University of Pisa is due to Pierfrancesco Foglia. His work regards a classification algorithm for coherence overhead in a shared-memory multiprocessor system, which overcomes many limitations of previous algorithms. The paper also surveys the current state-of-art techniques used to recognize (and potentially eliminate) various sources of coherence overhead.

Finally, in this issue you will appreciate two invited papers related to recently released simulators. We all know the key importance of simulators in our research, and these two new simulators are very interesting tools to develop further research for low power and performance, the two most important computer design issues (besides cost trade-offs). Both simulators are "genetic" mutations of the SimpleScalar simulator. The papers illustrate them in detail. The one from Pennsylvania State University is named SimplePower and allows for cycle level, RT level power estimation. The other is named HydraScalar, and it is from University of Virginia. The latter simulator includes a plethora of choices for advanced branch prediction simulations, and it is capable of simulating multipath execution.

I wish to thank the MEDEA program committee for its excellent referring work that helped increase the quality of the accepted papers. Finally, a special thank to Prof. Cosimo Antonio Prete and Jelica Protic for their precious support in the organizing committee.

## References

[1] J.E. Smith, "Decoupled Access/Execute Architectures", Proceedings of 9th International Symposium on Computer Architecture, pp. 112-119, May 1982.
[2] J.E. Smith, "Retrospective: Decoupled Access/Execute Architectures", in 25 Years of the International Symposia on Computer Architecture, p. 42, 1998.
[3] D. M. Tullsen, S. Eggers, and H. M. Levy, "Simultaneous Multithreading: Maximizing On-Chip Parallelism", Proceedings of the 22nd International Symposium on Computer Architecture, pp. 392-403, June 1995.