

Correction to RAMpage ASPLOS Paper

Philip Machanick

Department of Computer Science

University of the Witwatersrand

2050 Wits, South Africa

philip@cs.wits.ac.za

Abstract

This paper contains corrections to published results on the RAMpage memory hierarchy. The originally published results contained erroneous values for cache miss penalties for a conventional cache architecture against which the RAMpage hierarchy was being compared. The incorrect results showed that RAMpage with context switches on misses to DRAM had similar performance to a conventional 2-way associative L2 cache-based hierarchy. The corrected results show that the RAMpage hierarchy in fact outperforms a conventional 2-way associative cache hierarchy by up to 29%, for the measured variations.

1 Introduction

The RAMpage memory hierarchy is a proposed variation on a conventional cache hierarchy, in which the lowest-level cache is replaced by and SRAM main memory, and DRAM becomes a paging device [Machanick 1996, Machanick and Salverda 1998a, Machanick and Salverda 1998b]. The benefit of the RAMpage approach is that it achieves a fully associative L2 with no extra hit penalty; the trade-off is higher penalties for replacements from and misses to DRAM, which are handled in software.

Results published at ASPLOS'98 showed that the RAMpage hierarchy was a significant win over a 2-level cache hierarchy with a direct-mapped L2 (second level) cache, but had similar performance to a 2-way associative L2 cache. The overall conclusion was that RAMpage represented a data point in choosing between hardware and software trade-offs. The direct-mapped L2 had similar hardware complexity for lower performance, whereas the 2-way associative L2 had more complex hardware than RAMpage, to achieve similar performance to RAMpage's more complex software [Machanick *et al.* 1998].

While the ASPLOS paper presented an interesting result which was justified by the data presented in the paper, the data was incorrect. For the 2-way associative cache, incorrect values were used for miss penalties, in which all block sizes were given the same miss penalty: larger block sizes should have obviously had a larger miss penalty.

The corrected results show that the RAMpage approach scales up better as the CPU-DRAM speed gap grows, and that it is up to 29% faster than the 2-way associative cache for the fastest CPU simulated (with lesser improvements for slower CPUs). This result is significantly more interesting than that reported at ASPLOS.

The remainder of this paper presents the corrected results—which should be read on conjunction with the original paper—followed by modified conclusions based on the corrected results.

2 Modified Results

2.1 Introduction

The ASPLOS paper presents the following results:

- *baseline*—direct-mapped L1 and L2 caches; infinite DRAM to avoid page faults to disk
- *RAMpage*—L2 cache replaced by a paged SRAM main memory, with DRAM playing the role of a paging device; other parameters the same
- *RAMpage with context switches*—a context switch is taken on a miss to DRAM; otherwise the same as the other RAMpage simulation
- *better L2*—2-way associative L2 cache; otherwise the same as other simulations

Of these results, only the “better L2” case was incorrect.

The error in the results resulted from the fact that misses to DRAM were given the same penalty irrespective of the size of the block. Clearly, this parameter is incorrect, since other parameters such as bus size and memory speed were not varied—and is inconsistent in any case with the other simulations. A consequence of the incorrect parameter is that the largest L2 blocks simulated had the best simulated performance for the “better L2” case, a result which was difficult to explain.

This section starts with a brief summary of simulation parameters, followed by corrected versions of the “better L2” case, with both of the original RAMpage cases repeated for reference. The baseline case is not used for final comparisons (but rather as a basis for comparing with hardware of like complexity to RAMpage) and the baseline comparisons are not affected by the correction to the results, so the baseline data is not repeated.

In conclusion, on the basis of the corrected results, corrected comparisons are presented.

2.2 Simulation Parameters

This subsection contains a very short summary of the simulation parameters, since nothing is changed from the ASPLOS paper.

As with the ASPLOS data, the speed of DRAM is fixed. The L2 cache is fixed at 4Mbytes (the RAMpage SRAM main memory size is increased in line with the extra space needed for tags in each comparable L2 cache variation, from a base of 4Mbytes), and the L1 cache is fixed at 16Kbytes each of data and instruction cache (direct-mapped). Although the L1 design is not particularly aggressive, a more aggressive L1 cache would increase the benefits of the RAMpage model, as a higher fraction of simulated time would be spent in lower levels of the memory hierarchy.

An infinite DRAM is modeled to avoid the need to handle page faults to disk.

Pages in DRAM are translated through an inverted page table [Huck and Hays 1993], the same organization as is used for pages in SDRAM in the two RAMpage simulated hierarchies. Inverted page tables are useful in RAMpage because they ensure that if a page is present in a given level of memory, so is its page table entry.

DRAM is modeled on a simple implementation of Direct Rambus [Crisp 1997], with a transfer speed of 1.5Gbyte/s, after an initial latency of 50ns before the first pair of bytes is delivered (equivalent performance can be achieved with SDRAM on a 100MHz 128-bit bus).

All variants use a fully-associative TLB with 64 entries.

The simulated RAMpage hierarchies reserve a portion of their SRAM main memory for operating system code and data, including the inverted page table. Consequently, a TLB miss will never go to DRAM (or worse, to disk) if the page that is being referenced is in the SRAM main memory. Also, context switching code will never need to go to DRAM or disk.

The simulations are varied around three axes:

- the architecture variants listed in Subsection 2.1
- L2 cache block size (in the conventional architecture; equivalently SRAM main memory page size in the RAMpage architectures)
- CPU speed (200MHz to 4GHz), to model the growing CPU-DRAM speed gap

Multiple traces are interleaved to simulate a multiprogramming workload. While a superscalar CPU is not simulated, the CPU speed is taken as an issue rate, rather than a clock rate, and is intended to represent the average issue rate without memory hierarchy stalls.

For other details of the simulated architectures, see the ASPLOS paper [Machanick *et al.* 1998].

2.3 Corrected Results

issue rate	block size (bytes)						speedup best vs. best L2
	128	256	512	1024	2048	4096	
200MHz	8.48	7.11	6.41	5.99	6.03	6.09	1.04
	9.92	7.67	6.72	6.16	<i>6.05</i>	6.11	1.03
	6.27	6.24	6.24	6.25	6.28	6.35	
500MHz	3.46	2.88	2.61	<i>2.44</i>	2.45	2.49	1.05
	3.96	3.08	2.68	2.56	2.52	2.43	1.05
	2.58	2.56	2.55	2.56	2.59	2.65	
1GHz	1.80	1.49	1.35	<i>1.26</i>	1.27	1.29	1.05
	1.93	1.54	1.39	1.26	1.23	1.25	1.07
	1.35	1.33	<i>1.32</i>	1.34	1.36	1.41	
2GHz	0.95	0.78	0.70	<i>0.66</i>	0.67	0.68	1.08
	0.99	0.75	0.67	0.63	0.62	0.62	1.15
	0.73	<i>0.71</i>	<i>0.71</i>	0.72	0.75	0.80	
4GHz	0.53	0.44	0.39	<i>0.37</i>	<i>0.37</i>	0.39	1.08
	0.47	0.38	0.33	0.33	0.31	0.31	1.29
	0.42	<i>0.40</i>	<i>0.40</i>	0.41	0.44	0.49	

TABLE 1. Corrected Comparisons

Elapsed simulated time (s) for 1.1 billion-reference combined traces.

*Each row contains the RAMpage hierarchy at the top, then the RAMpage hierarchy with context switches in misses, and finally, the 2-way associative L2 cache (best in each line in italics; best for CPU speed **bold**)*

Table 1 contains corrected data, illustrating how the RAMpage hierarchy is more scalable than the conventional 2-way associative L2 cache.

Note also that the L2 block size at which the best performance is achieved is more in line with expectation than in the ASPLOS paper: the 2-way associative L2 generally has its best results with a block size of around 512 bytes, and performance degrades significantly as block size increases. As is to be expected, taking a context switch on a miss results in larger SRAM page sizes being favourable for RAMpage.

2.4 Comparison

While the simpler RAMpage implementation is the fastest for the slowest system, the addition of context switches on misses becomes progressively more useful as the CPU-DRAM speed gap grows. Both RAMpage simulated systems scale better as the CPU-DRAM speed gap grows, but the version with context switches on misses becomes progressively more competitive as the speed gap grows.

For the 4GHz issue rate, the context-switched version is 29% faster than the best case of the 2-way associative L2 cache, while the simpler RAMpage version is 8% faster than the conventional L2 cache.

3 Conclusions

Read in conjunction with the ASPLOS paper, these revised results show that RAMpage is an idea of significant potential. Further, the results are more in line with expectation: larger page sizes (blocks) work better for RAMpage than for the 2-way associative L2, especially when a context switch is taken in a miss.

A speed improvement of 29% is sufficiently large to justify considering implementation. However, there is a practical problem of integrating the required changes into an operating system. While it is possible that the implementation could be hidden from the operating system for the straightforward RAMpage case, implementation of context switches on misses would be difficult without operating system changes.

Further work on RAMpage is planned. Projects currently underway include trying more variations on memory system parameters (particularly changes in L2/SRAM main memory size), and more detailed quantification of operating system effects.

Other areas which could be useful to explore include hardware implementation of full associativity, the value of RAMpage in multithreaded applications, and further detail of implementation.

Acknowledgments

Jean Bilbrough pointed out an inconsistency in the data which led me to discovering the error in the reported results.

References

- [Crisp 1997] R Crisp. Direct Rambus Technology: The New Main Memory Standard, *IEEE Micro*, vol. 17, no. 6, November/December 1997, pp 18-28.
- [Huck and Hays 1993] J Huck and J Hay. Architectural Support for Translation Table Management in Large Address Space Machines, *Proc. 20th Int. Symp. on Computer Architecture (ISCA'93)*, San Diego, CA, May 1993, pp 39-50.
- [Machanick 1996] The Case for SRAM Main Memory, *Computer Architecture News* vol. 24, no. 5, December 1996 pp. 23-30
- [Machanick and Salverda 1998a] P Machanick and P Salverda. Preliminary Investigation of the RAMpage Memory Hierarchy, *South African Computer Journal*, no. 21, August 1998, pp. 16-25.
- [Machanick and Salverda 1998b] P Machanick and P Salverda Implications of Emerging DRAM Technologies for the RAMpage Memory Hierarchy, *Proc. SAICSIT'98*, Gordon's Bay, South Africa, November 1998, pp 27-40.
- [Machanick et al. 1998] P Machanick, P Salverda and L Pompe. Hardware-Software Trade-Offs in a Direct Rambus Implementation of the RAMpage Memory Hierarchy, *Proc. ASPLOS-VIII Eighth Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, San Jose, October 1998 pp. 105-114.