RETROSPECTIVE:

The DASH Prototype: Implementation and Performance

Daniel E. Lenoski

Silicon Graphics
lenoski@sgi.com

James P. Laudon

ZSP Corporation
laudon@zsp.com

Our paper entitled "The DASH Prototype: Implementation and Performance" was given at the 19th ISCA in Gold Coast, Australia in May of 1992. This paper outlined our implementation experience and initial performance details of DASH, the first hardware implementation of the ccNUMA architecture. DASH was a large multi-faceted research project at Stanford University led by John Hennessy, Anoop Gupta, Monica Lam, and Mark Horowitz. The overall goal of DASH was to break the scalability barrier of bus-based SMP machines and provide the massive parallelism of distributed memory while maintaining the shared-memory paradigm. While there had been previous switch-based SMPs built in the early 1980s (e.g., the Cray X-MP, Univ. of Illinois Cedar, BBN TC-1000, and IBM RP3), DASH added hardware support for global cache coherence. Hardware cache coherence improved processor performance and removed the burden of coherence from the user or compiler. During the late 1980's our group was not alone, there were efforts at MIT (Alewife and J-Machine), University of Wisconsin (Multicube), Encore Computer (Gigamax), Kendall Square Research, and the IEEE Scalable Coherent Interface (SCI) standards effort, but ours was the first to build a hardware implementation of this new class of machine.

The high-level structure of DASH was a collection of nodes, each including one or more processors and a portion of the global memory, connected by a scalable interconnect (a 2-D mesh). Directory-based coherence, originally proposed by Censier and Feautrier in the late 1970s, was employed since it removed the need for the global bus found in snoopy systems. While the original directory schemes used a central memory/directory, moving to a distributed organization scaled memory bandwidth naturally with the number of processors. With this fundamental system structure in mind, and previous studies showing the potential of distributed-directories (see the Agarwal/Hennessy paper on directories in this collection), work began on the DASH prototype.

DASH Prototype Goals and Timeline

Scaling the cache-coherent SMP model to hundreds of processors raised many questions in the area of processor and system architecture, operating systems, compilers, programming languages, and parallel applications. We chose to build an actual hardware prototype of the architecture to address these questions as well as to:

- Understand the hardware complexities of actually building this type of machine.
- Provide more insight into the performance attributes of a real ccNUMA machine.
- Allow a comparison of real applications' complexities and performance (not simply small simulated kernels) among highly parallel shared-memory programs and their message-passing counterparts.

The difficulty of implementing a distributed directory protocol was of serious concern since it amounts to replacing the software controlled network interfaces on message-passing machines with hardware control for sending network messages to fetch remote memory and maintain cache coherence. At the time, it wasn't clear if this hardware complexity was tractable, and even if it was, would the performance of a ccNUMA be competitive with message-passing systems?

We began detailed architecture work on the system in fall of 1988. Early on, we made a choice to leverage an existing SMP system as our base

node because these machines provide the necessary hooks for controlling the processor caches from their bus interfaces. We were anxious to utilize a RISC-based SMP, and the recently announced SGI 4D/240 series was the only such machine on the market at the time. This choice turned out to be very fortuitous, since utilizing an existing system allowed us to leverage much of the system hardware and software and concentrate our efforts on the unique ccNUMA hardware and software.

Initial power-on of the prototype system was in the Fall of 1990 and a 16 processor system was stable in the Spring of 1991. We then started work on a larger 64 processor system, which resulted in a stable 48 processor prototype in the Spring of 1992. Nagging problems with our ribbon-cabled mesh links prevented us from reaching the goal of 64 processors in a single system (a 4x4 mesh of 4 processor nodes), but we were able to learn much from the 48 processor prototype.

Innovations in DASH

During the architecture phase of the project, our focus was on the coherence protocol and mechanisms that would minimize memory latency and maximize memory bandwidth. In addition, we realized that hiding memory latency would also be key since the distributed structure of a large ccNUMA would invariably lead to longer memory latency. Likewise, support for large-scale parallelism demanded that we pay attention to synchronization and inter-processor communication. Being one of the first to tackle these problems in the context of a ccNUMA machine, these goals led to many innovative solutions. These included:

- Software-controlled non-binding cache line prefetch to hide latency and increase memory pipelining.
- Release-consistency support with fence/memory barriers to help hide store latency.
- Queue-based test-and-set locks to allow efficient contended spin-locks.
- Fetch&Inc and Fetch&Dec (borrowed from the NYU Ultracomputer, but without combining) for support of efficient barrier synchronization and distributed queues.
- Update coherence and deliver instructions which provide low latency inter-processor word and cache line communication respectively.

The actual hardware implementation phase also demanded innovative solutions such as:

- An efficient "forwarding" coherence protocol which minimized latency for accessing dirty data and writing to shared cache lines.
- Support for both invalidate and update coherence within the same directory protocol.
- Separate request and reply paths that prevented dead-lock on the normal memory requests together with retry mechanisms that handled race conditions in the distributed directory protocol.
- A high-bandwidth DRAM directory access path which performed read-modify-write cycles under the shadow of the main memory's fetch of 16-byte memory blocks.
- One of the first lock-up-free caches that implemented a remote access cache to track outstanding memory references and supplement the processor caches with features such as prefetch.

Lessons Learned

As one would expect, building and using the DASH prototype led to many new insights and lessons that were both positive and negative. The most positive result was that it was feasible to build a ccNUMA machine and to achieve good performance on highly parallel shared-memory applications. Furthermore, by analyzing the logic in the directory and network interface, the prototype demonstrated that adding hardware cache coherence added only 10% additional hardware over a non-coherent MPP system structure. Another lesson was that with close attention, it was possible to keep remote-to-local memory latency to within a 3 to 1 ratio. Several features included in the prototype proved very successful. Operations such as prefetch proved to be very powerful in hiding memory latency and improving the pipelining of memory operations. Fetch&Op performed at memory also greatly reduced the overhead of barrier-type synchronization by reducing the serialization time for atomic counter operations.

Other features that did not yield as much performance improvements as expected were queue-based locks and update and deliver operations. While these operations could greatly aid in specific low-level communication, the overhead of general communication associated with inter-processor

data sharing tended to swamp out the incremental enhancements that these operations provided. This was especially true on the prototype hardware where our remote access cache was as close as we could get the data to the processor (thus reducing latency by no more than a factor of 3).

Another somewhat unexpected result was the negative impact of using a bus-connected internode interface. Since memory operations needed to cross the processor's local bus twice and memory home's bus once, the resulting memory bandwidth when all processors are accessing remote memory was no better than one-third that of local memory. In fact, DASH's bus-bandwidth was a greater limit on global remote memory bandwidth than network bisection bandwidth. While not inherent to ccNUMA systems, this issue illustrated the limitations of simply extending a bus-based SMP with a ccNUMA network interface card.

The advantages of leveraging an existing SMP was also one of the indirect, but very positive, lessons from the DASH project. Using an existing SMP allowed a small university team to focus their attention on the important task of architecting and designing the hardware necessary to implement a ccNUMA machine. In addition, the choice of the SGI 4D/240 as the base node helped in the quick development of DASH, as its modest level of integration (by today's standards) allowed us to work primarily at the board level with PALs and FPGAs. Working at this level of integration reduced both design and debug time. There were some compromises due to leveraging an existing machine, but it reduced the time from concept to running real applications under Unix to less than 2.5 years. This increased the impact of the actual DASH hardware and helped validate the feasibility of the ccNUMA architecture.

Conclusions

The impact of the DASH project has been felt both in the academia and industry. Scalable shared-memory multiprocessors continue to be a hot topic of research. DASH helped validate the viability of the ccNUMA approach and provide a baseline to evaluate improvements in coherence protocols, scalable directory storage, and alternative system architectures such as COMA. ccNUMA systems have now been commercialized by a number of vendors including HP/Convex, Silicon Graphics, Sequent, HaL, and Data General. The DASH prototype helped pave the way for these commercial developments by detailing many of the fundamental design problems with ccNUMA machines and demonstrating that the shared-memory paradigm could be scaled and realize both good performance and good cost-performance.

Building the DASH prototype would never have been possible without the hard work of a number of individuals. These included our co-authors: Truman Joe, David Nakahira, Luis Stevens, Anoop Gupta, and John Hennessy. Additional contributions to the hardware development were made by Kourosh Gharachorloo, Wolf-Dietrich Weber, Mark Horowitz, Tom Chanak, John Maneatis and Monica Lam. Help from Silicon Graphics, namely Jim Barton, Forest Baskett, John Burger, Doug Solomon and John Carlson, was also instrumental. Dan Lenoski was supported by Tandem Computers during his graduate work. John Toole and Gil Weigand at DARPA provided the funding to support the greater team and build the DASH prototype.

For additional details on DASH see:

[1] D. Lenoski, J. Laudon, T. Joe, D. Nakahira, L. Stevens, A. Gupta, and J. Hennessy, "The DASH Prototype: Implementation and Performance.," *IEEE Trans. on Parallel and Distributed Systems*, 4(1)41-61, January 1993.

[2] D. Lenoski and W.-D. Weber, *Scalable Shared-Memory Multiprocessing*, Morgan Kaufmann Publishers, San Francisco, CA 1995