SCALABLE SHARED-MEMORY MULTIPROCESSING

DANIEL E. LENOSKI

WOLF-DIETRICH WEBER

SCALABLE SHARED-MEMORY MULTIPROCESSING

Scalable Shared-Memory Multiprocessing

Daniel E. Lenoski
Wolf-Dietrich Weber

Morgan Kaufmann Publishers
San Francisco, California

Executive Editor Bruce Spatz
Production Manager Yonie Overton
Assistant Production Editor Julie Pabst
Cover Design Feroza Unvala
Text Design Jim Love/Studio Arno
Composition Ed Sznyter
Copyeditor Ken DellaPenta
Proofreader Liz Sizensky
Printer Quebecor Printing

Morgan Kaufmann Publishers, Inc.
Editorial and Sales Office
340 Pine Street, Sixth Floor
San Francisco, CA 94104–3205
USA
Telephone 415/392-2665
Facsimile 415/982-2665
Internet mkp@mkp.com

99 98 97 96 95 5 4 3 2 1

Library of Congress Cataloging-in-Publication Data

Lenoski, Daniel E.
Scalable shared-memory multiprocessing / Daniel E. Lenoski, Wolf-Dietrich Weber.
p. cm.
Includes bibliographical references and index.
ISBN 1-55860-315-8
1. Multiprocessors. 2. Memory management (Computer science)
I. Weber, Wolf-Dietrich. II. Title.
QA76.5.L457 1995
004'.35--dc20 **95-19615**
CIP

To Karen and Véronique
and Veronica, Steven, and Nicole, too

This page intentionally left blank

Foreword
by
John L. Hennessy

The idea of multiprocessors dates back to the first electronic computers. As early as the 1950s, MPs were advanced as a technique both to increase reliability and to improve performance. Early efforts concentrated on building small-scale machines, which avoided a number of problems. Two exceptions were the C.mmp and Cm* machines built at Carnegie-Mellon University in the 1970s. These systems included many of the features found in more recent large-scale machines. However, simpler structures, adequate for the smaller systems being built at the time, became more popular.

In the early 1980s, the first commercially successful multiprocessors became available. Almost all of these designs were bus-based, shared-memory machines. Their development was enabled by two key factors: the incredible price-performance advantages of microprocessors and the extensive use of caches in microprocessor-based systems. These technological factors made it possible to put multiple processors on a bus with a single memory, relying on caching to reduce the memory bandwidth demands of the processors to an acceptable level. Coherency among the caches was maintained through a snoopy protocol, which, while simple, employed broadcast, making these systems fundamentally unscalable.

In the mid 1980s, the interest in scalable multiprocessors grew quickly. The bus-based, cache-coherent, shared-memory machines were clearly not scalable, so designers focused on two other approaches. One approach gave up on coherence but maintained shared memory. Three important designs of this genre were the BBN Butterfly, the IBM RP3, and the NYU Ultracomputer. As in earlier experiments, the absence of caching significantly increased the cost of shared access—each shared access had to traverse the network—and made it hard to obtain good performance on such machines. The Cray T3D is the major machine in this camp at the present. In practice, it appears that such machines are most effectively used as message-passing machines, with the shared address space providing a convenient naming mechanism.

The second approach to scalability relied on message passing rather than shared memory. The problem of memory latency on the non-coherent shared-memory machines, together with the impossibility of scaling the snoopy approach, led some designers to conclude that message passing was the only viable approach for scalable multiprocessors. However, message-passing machines have proved very difficult to program.

This book explores a new class of machines supporting both cache-coherent shared memory and scalability to a large number of processors. The first part of the book looks at the general concepts involved in building scalable multiprocessors, especially scalable shared-memory machines. The authors begin with a description of the challenges in designing scalable architectures. They then discuss the trade-offs between shared memory and message passing, as well as the challenges of scalable cache coherency. In considering the scalability of performance of any particular scheme for shared memory, the results are substantially affected by the communication and sharing characteristics in the applications. Chapter 2 presents extensive data on this important topic. Designing a large-scale, shared-memory machine requires attention to minimizing memory latency and hiding that latency whenever possible. These two topics provide the key focus of Chapter 3. Chapter 4 examines a variety of design alternatives that arise in the actual implementation of such systems. These first four chapters are valuable to both designers and programmers of scalable machines who wish to understand the performance of various architectures, as well as the variety of design alternatives. Finally, Chapter 5 surveys historical and proposed scalable shared-memory machines, including the Stanford DASH multiprocessor.

The DASH project began in 1987 to explore whether it was possible to build scalable shared-memory multiprocessors that efficiently support cache coherence. By using a distributed-memory model, explored in both the Cm* and RP3 projects, DASH allows memory bandwidth to be scaled, unlike that in a bus-based system. To implement scalable cache coherence, DASH uses a distributed directory protocol. In 1991, DASH became the first operational shared-memory multiprocessor with a scalable hardware cache coherence mechanism. Kendall Square Research and Convex became the first companies to market scalable shared-memory multiprocessors commercially. Such machines have become known as Distributed Shared Memory (DSM). The interest in DSM architectures is growing quickly, both because of the scalability advantages and because of the increasing difficulty of building bus-based multiprocessors as processors increase in speed.

The second part of this book is a retrospective examination of the design of the DASH multiprocessor. It provides not only a detailed description of the design, but also a rationale for the design and an exposition of the challenges arising in any such machine. For people interested in scalable machines, this portion of the book is a must-read. Unlike the documentation of industrial projects, which rarely criticizes their products, this book supplies many frank assessments of the strengths and weaknesses of the DASH design, providing readers with extensive input on how the design might be improved upon in the future.

Although the largest multiprocessors seem to be moving toward shared memory and away from message passing, it remains a point of debate whether cache coherency will play a major role in very large machines. The final section of the book explains how a cache-coherent design like DASH could be extended to the range of thousands of processors and TERAOps of performance. In addition to examining the structure of such a design, this section estimates the performance that might be achievable. The analysis shows that building large-scale parallel machines does not mean abandoning the efficient and easy-to-use shared-memory programming model.

I found this book extremely interesting. The first portion of the book lays a solid foundation, showing why it is possible to build a shared-memory scalable machine. Then, in the second portion, the description of the landmark DASH machine by one of its original designers is both fascinating and informative. Finally, the third portion gives a glimpse at the not-so-distant future of scalable shared-memory multiprocessors. Such scalable, cache-coherent machines appear to be major candidates for next-generation multiprocessors. Anyone involved in the design of such machines, in software development for them, or even in using such machines will find this book useful and enlightening.

This page intentionally left blank

Contents

Preface

Scalable shared-memory multiprocessing (SSMP) systems provide the power of hundreds to thousands of high-performance microprocessors all sharing a common address space. SSMP is appealing because it addresses a number of problems with existing parallel systems. Relative to large-scale message-passing machines, SSMP provides a much simpler programming model, which makes the system easier to use. The shared-memory paradigm also reduces communication overhead, making finer-grained work sharing possible. Compared with traditional bus-based shared-memory machines, SSMP does not have the bus bandwidth problem, which limits the number of processors that can be combined in a single system.

SSMP has been a popular research topic since the development of the IBM RP3 and NYU Ultracomputer in the early 1980s. The problem with these early SSMP machines was that the performance of individual processing nodes was limited by memory latency and by the modest performance of the highly integrated processors (i.e., microprocessors) available at the time. The performance of the latest microprocessors has eliminated the internal performance limits, but the challenge of keeping memory latency low enough so that performance scales when the processors are combined remains.

This book describes the architecture and hardware design of SSMP systems that combine high-performance processors with a scalable cache coherence mechanism. Caching helps to keep the effective memory latency low and processor utilization high. The material from the book is drawn from experience with the DASH project at Stanford University. In early 1991, this work resulted in the first operational SSMP machine with a scalable cache coherence mechanism. The prototype machine combines up to 64 RISC processors, with an aggregate performance of 1.6 GIPS and 600 MFLOPS, all sharing a hardware-supported cache-coherent memory address space.

We wrote this book with three audiences in mind: (1) designers and users of commercial SSMP machines (the first generation of SSMP systems such as the Kendall Square Research KSR-1 and Convex Exemplar are already in the market), (2) researchers of parallel machine architecture, and (3) anyone studying advanced computer architecture and parallel processing.

The book is broken into three major parts. Part I introduces general SSMP concepts and the design issues that all SSMP systems face. Part II is a detailed study of the DASH prototype machine and its performance. Part III concludes with an outline of a very large-scale machine based on the DASH design that could be built with today's state-of-the-art technology. It combines up to 2,000 processors to deliver a peak performance of over one trillion instructions per second and 400 GFLOPS.

The general SSMP concepts presented in Part I are broken into five chapters. Chapter 1 relates the base SSMP design to other major classes of parallel machines. Chapter 2 analyzes the characteristics of a number of parallel applications and their implications for system design. Chapter 3 discusses the important performance issues that every SSMP system must address. Chapter 4 covers design issues related to scalability of the system. Chapter 5 surveys the major SSMP systems that have been proposed and highlights their unique features.

Part II of the book details the design and performance of the DASH prototype machine. Chapter 6 gives an overview of the DASH architecture and a description of the coherence protocol. Chapter 7 describes the detailed hardware design of the system and calculates the overhead of the scalable coherence mechanism quantitatively. Chapter 8 summarizes performance measurements taken from full applications run on the prototype. It also uses stress tests to demonstrate the effectiveness of extensions made to the standard memory protocol.

Part III addresses the future of SSMP systems. Chapter 9 outlines the design of a very large-scale system based on the DASH prototype. It gives solutions to the major problems not addressed in the smaller DASH prototype and an estimate of the performance of the system. Chapter 10 concludes the book with an overview of the trends in SSMP systems and an outlook for this class of system.

The DASH project at Stanford was an exciting research project to which many individuals contributed. We are proud to have been part of such an inspiring group. In particular, we would like to acknowledge the guidance of our advisors, Anoop Gupta and John Hennessy, and the hard work of our DASH hard-core teammates (Jim Laudon, Dave Nakahira, Truman Joe, and Luis Stevens) that made DASH a reality. The project also received contributions from a much wider audience of Stanford faculty (especially Mark Horowitz and Monica Lam) and students (too many to mention). We would also like to acknowledge the generous support from DARPA and Silicon Graphics, without which DASH would not have been possible.

For help with this book, we would first like to thank those who reviewed the manuscript: Rick Bahr (Silicon Graphics), Andreas Nowatzyk (Sun), J. P. Singh (Princeton), Jim Smith (University of Wisconsin), Len Widra (Silicon Graphics), and others. Their input greatly improved the presentation and accuracy of the book. We would especially like to thank Jim Smith, whose careful reading and insightful comments led to a major restructuring of the material. The leaders of the projects surveyed in this book also reviewed the manuscript to verify that their projects were discussed accurately. We'd like to thank the following for their input: Anant Agarwal (MIT), Mike Beckerle (Motorola), David Cheriton (Stanford), Bill Dally (MIT), Allan Gottlieb (NYU), Erik Hagersten (Sun), Dave James (Apple), Kai Li (Princeton), Burton Smith (Tera Computer), and Steve Wallach (Convex). We would also like to thank the folks at Morgan Kaufmann for making the book a reality. In