The ParaStation Project

Joachim M. Blum, Thomas M. Warschko, and Walter F. Tichy

University of Karlsruhe,Dept. of Informatics
Postfach 6980,76128 Karlsruhe,Germany
email:{blum,warschko,tichy}@ira.uka.de

Abstract. ParaStation is a high speed communication subsystem which enables a cluster of workstations or PCs to offer performance comparable to a dedicated parallel machine. Each node is still usable as a regular workstation or PC. Speedups of 7.4 on a 8 node cluster have been achieved even on communication intensive programs. The ParaStation software supports a real multitasking environment and combines the advantages of workstation clusters and of parallel machines.

1 Introduction

The basic concept of the ParaStation approach is to build a scalable and efficient parallel processor from off-the-shelf workstations and PCs. In ParaStation this is done with a second interconnection network dedicated to parallel computation together with well-known programming interfaces such as UNIX sockets and PVM. By taking advantage of mass production, the resulting system is considerably less expensive than traditional parallel machines.

The ParaStation system [?] is based on the retargeted MPP network of Triton/1 [5, 6]. The goal is to offer MPP-like communication performance while supporting a standard, but efficient programming interface. The communication hardware is a PCI-bus interface card which can be used in many platforms including Intel PCs, Digital Alphas, and Motorola PowerPCs. Other platforms, such as HP, Sun, and SGI, will follow.

2 Existing Platforms

ParaStation already supports two different platforms. The initial realization was based on Digital Alpha workstations running Digital Unix (OSF/1). Recently we have ported the software to Intel Pentium PCs running Linux. Both systems support multiple programming interfaces and show high performance on all layers. The port of Windows NT on Alphas and PCs is scheduled to be completed during summer 96.

The ParaStation network provides a high data rate, low latency, scalable topology, flow control at link level, minimized protocols, and reliable data transmission. It is dedicated to parallel applications and is not intended as a replacement for a common LAN. This restriction allow the use of specialized network

features, optimized point-to-point protocols, and controlling the network at user-level without operating system interaction.

The ParaStation system library (see figure 1) consists of three building blocks: the hardware interface layer, the central system layer, and the standardized user interface (sockets).

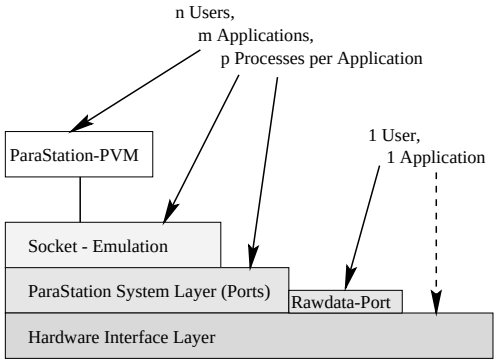

Fig. 1. ParaStation system library

The hardware layer provides the necessary abstraction of the underlaying hardware to maintain a transparent and system independent interface to the upper layers.

Since messages at this level are addressed to nodes rather than individual communication channels, message headers simply contain the address of the target node, the number of data words contained in the packet, and the data itself. When sending a message, data is copied directly from user-space memory to the interface board and the receiving function does the same vice versa, eliminating all intermediate buffering.

The ParaStation system layer establishes multiple communication channels between applications and supports a multiuser/multiprogramming environment. To meet our primary design goal of efficiency we have reassembled operating system functionality at user level.

To support individual communication channels (called *ports* in ParaStation), the system layer maintains a minimal software protocol which adds information about the sending and receiving port in each packet. This concept is sufficient to support multiple processes by using different port ids for different processes. Critical sections while sending and receiving messages are locked with semaphores, which are also implemented in user space by using processor-supported atomic operations.

As a result, the implementation of these concepts does not need system calls at all. Furthermore, we provide a zero-copy behavior (no buffering) whenever possible. This leads to high bandwidth and low latencies.

The socket layer provides an emulation of the standard UNIX socket interface (TCP and UDP connections), so applications using socket communication can be ported to the ParaStation system with little effort. For connections outside a ParaStation cluster, regular operating system calls are used. `Send/recv` calls, which can be satisfied within the ParaStation-cluster, do not need any interaction with the operating system.

ParaStation implementations of standard programming environments like PVM [1, ?], P4 [2], TCGMSG [4], or MPI (`mpich`) [3] use ParaStation sockets for high-speed communication. This approach allows us to easily port, maintain, and update these packages. We use the standard workstation software.

3 Performance

ParaStation achieves high communication performance on all existing platforms. Process to process latencies as low a 1.6 μs (Linux PCs) and 2.5 μs (Alphas) have been demonstrated. On the level of the well known Unix socket interface (TCP/IP functionality), latencies of 15 μs and 20 μs have been achieved, respectively. The bandwidth can be as high as 15 MBytes/s (10.5 MBytes on Alphas). On the socket level a throughput of 11.7 MBytes/s (PC) and 8.8 MBytes/s (Alpha) is possible.

Focusing on latency and throughput only is too narrow for a complete evaluation. It is necessary to show that a low-latency, high-throughput communication subsystem also achieves a reasonable application efficiency. Our approach is twofold. First, we took a *heat diffusion* benchmark to test application performance on our proprietary interface. This benchmark requires both high bandwidth and low latency. In each iteration the edges of the distributed partitions have to be exchanged and every 20 iterations all data is gathered by one node to display the result. A speedup of 7.4 is achieved on a 8-node cluster (DEC-Alpha).

Second, we installed the widely used and publicly available ScaLAPACK math library, which first uses the BLACS package and then PVM as communication subsystem. Thus, the complete protocol hierarchy as presented in the previous section is involved. The application benchmark for ParaStation (*xslu*) is an equation solver for dense systems. Remarkable is a speedup of 6.2 on an eight-node cluster (Alpha 21064A,275MHz) with a aggregate performance of one Gigaflop. Since ScaLAPACK is available for several platforms, this result is directly comparable to other systems.

4 Future Platforms

Due to the widesread availability of PCI-bus systems, ParaStation is not limited to DEC-Alpha and PC platforms. IBM PowerPC as platform is possible and Sun (Ultra-Sparc), SGI and HP have announced PCI-based systems.

Beside the support of various platforms we plan to improve the performance of the communication hardware. The next generation hardware will use fibre optic communication links to get a process to process bandwidth of 100 MBytes/s.

Fibre optic links also enable us to extend the distance between two distinct nodes to more than 1 km.

5 Conclusion

ParaStation proves that efficient parallel computing is possible on a cluster of workstations. A ParaStation network performs similar to parallel machines. Message passing standards such as PVM and MPI are available, simplifying porting of existing parallel programs. Each node is still usable as regular workstation and the parallel application can share processor resources whenever requested.

References

1. A. Beguelin, J. Dongarra, Al Geist, W. Jiang, R. Manchek, and V. Sunderam. *PVM 3 User's Guide and Reference Manual.* ORNL/TM-12187, Oak Ridge National Laboratory, May 1993.
2. Joachim M. Blum, Thomas M. Warschko, and Walter F. Tichy. PSPVM: Implementing PVM on a high-speed Interconnect for Workstation Clusters. In *Proc. of 3rd Euro PVM Users' Group Meeting*, Munich, Germany, Oct.7-9, 1996.
3. Ralph Buttler and Ewing Lusk. *User's Guide to the p4 Parallel Programmimg System.* ANL-92/17, Argonne National Laboratory, October 1992.
4. William Gropp, Ewing Lusk, and Anthony Skjellum. *Using MPI.* MIT-Press, 1994.
5. R. J. Harrison. Portable tools and applications for parallel computers. *International Journal on Quantum Chem.*, 40:847–863, 1991.
6. Christian G. Herter, Thomas M. Warschko, Walter F. Tichy, and Michael Philippsen. Triton/1: A massively-parallel mixed-mode computer designed to support high level languages. In *7th International Parallel Processing Symposium, Proc. of 2nd Workshop on Heterogeneous Processing*, pages 65–70, Newport Beach, CA, April 13–16, 1993.
7. Michael Philippsen, Thomas M. Warschko, Walter F. Tichy, and Christian G. Herter. Project Triton: Towards improved programmability of parallel machines. In *26th Hawaii International Conference on System Sciences*, volume I, pages 192–201, Wailea, Maui, Hawaii, January 4–8, 1993.
8. Thomas M. Warschko, Joachim M. Blum, and Walter F. Tichy. The ParaPC/ParaStation Project: Efficient Parallel Computing by Clustering Workstations. Technical Report 13/96, Universität Karlsruhe, Fakultät für Informatik, 1996.

This article was processed using the LaTeX macro package with LLNCS style