PowerMANNA: A Parallel Architecture Based on the PowerPC MPC620

P.M. Behr
GMD FIRST
peter@first.gmd.de

S. Pletner
GMD FIRST
samuel@first.gmd.de

A.C. Sodan
University of New Mexico
acsodan@cs.unm.edu

Abstract

The paper presents PowerMANNA - a distributed-memory parallel computer system based on the 64-Bit PowerPC processor MPC620. The PowerMANNA node architecture supports all the sophisticated features of the MPC620 and incorporates important architectural concepts that allow us to exploit the performance of modern superscalar microprocessors in the context of massively parallel supercomputing. The two-way processor nodes of PowerMANNA are embedded in a powerful communication system supporting low-latency communication and maximum connectivity. Processing and communication performance of an eight-node prototype are shown and compared with shared-memory machines and clusters. In the course of the presentation, experience gained with the PowerPC MPC620 processor is discussed.

1. Introduction

The high-performance parallel processing market is currently dominated by symmetric-memory processing (SMP) systems – such as SGI Origin or SUN UE10000. SMP systems with up to 128 processors (more typical are 16) are powerful compute servers. They are easy to program and benefit from the high interaction speed between processors as a result of the tight integration. However, shared-memory systems have the fundamental disadvantage of being of limited scalability. Thanks to an improved understanding of parallel programming, applications in the field of high-performance scientific computing are being increasingly designed to run parallel computers with distributed-memory architectures that are scalable to massive parallelism (MPP).

The best price/performance ratio for distributed-memory machines is provided by massively parallel clusters built from off-the-shelf hardware components of workstations or high-end PCs (even using standard packaging and power supplies) and fast standard networks such as Myrinet, SCI or Gigabit Ethernet.

In the current TOP500 list [1], the most powerful super computers are highly parallel MPP systems. These systems are specifically designed for optimal performance and dense packaging. Using components like microprocessors, memory modules and communication devices produced for the larger market of high-end workstations or servers and standard networks, these systems have a price/performance ratio somewhere between high-end vector machines and PC clusters. The preferred building blocks of these high-end MPP systems will be very powerful SMP nodes, the communication hardware being an integral part of the node or even the processor architecture. By avoiding additional interfaces and protocol conversions like the PCI and IP protocols commonly used in PCs and workstations, minimum latency and maximum throughput is attainable. Achieving better than Teraflops performance from current technology requires up to a thousand powerful SMP nodes and thus a sufficiently scalable communication network.

PowerMANNA is a highly scalable massively parallel computer system. Considerable effort has been made to obtain the best possible sustained performance from its hybrid SMP/MPP architecture. The computing nodes are based on two-way processor nodes with Motorola's MPC620 superscalar processors. Thus, both features – the scalability of the MPPs and the cost-effectiveness of small SMPs – are exploited by the PowerMANNA architecture.

The MPC620 implements the PowerPC architecture as specified for 64-bit addressing. The superscalar processor provides sophisticated support for shared-memory multiprocessing, distinguishing it from other processors. In summary, the PowerPC MPC620 was one of the most promising processors when the PowerMANNA project started. The new Power3 processor from IBM, the successor of the MPC620, demonstrates the viability of its design concepts [2].

The distinguishing features of PowerMANNA are its hierarchical scalable interconnect, its duplicated communication network with two separate link interfaces on each node, and the short message start-up times based on a simplified network interface (NI). In addition, the nodes support shared-memory processing by specifically designed address and datapath switches resulting in multiple point-to-point connections instead of a single shared-bus system.

The topology of PowerMANNA's interconnect is formed by a hierarchy of crossbars. The nodes are connected to the duplicated scalable communication system via two bidirectional communication links, providing a total bandwidth of up to 240 Mbyte/s and less than 4 μs latency for small messages. Nodes connected to crossbars form clusters that, again, can be interconnected in a hierarchical manner to create arbitrary configurations with hypercube communication interconnectivity.

The paper presents important architectural features of the node and the communication system, and evaluates Power-MANNA's performance in comparison with pure SMPs and clusters. In particular, the scalability of the two-way node and the performance of the communication system are explored and compared with other systems. The performance evaluation aims mainly to validate the architectural design. The concrete results suffer to some extent from the fact that the PowerPC MPC620's technology is no longer quite state-of-the-art and the successor Power3 has not been made generally available by IBM.

In Section 2, we present the node architecture and its hardware implementation in more detail. Section 3 describes PowerMANNA's communication system and Section 4 presents the software environment. In Section 5, the test results of various benchmarks are discussed. Related work is considered in Section 6, Section 7 concluding the paper with a summary.

2. Node architecture

The computing performance of PowerMANNA is based on dual-processor nodes with two 64-bit PowerPC MPC620 processors. The MPC620 processor implements the PowerPC architecture as specified for 64-bit addressing. The superscalar processor is capable of issuing four instructions simultaneously. Its six execution units can operate in parallel, and as many as six instructions can complete execution in parallel. The MPC620's rename buffers, reservation stations, dynamic branch prediction and completion unit increase instruction throughput, guarantee in-order completion and ensure a precise exception model.

The MPC620 has separate memory-management units (MMUs) and separate 32 Kbyte on-chip caches for instructions and data, and provides support for demand-paged virtual-memory address translation. The MPC620 has a 40-bit address bus, and it can be configured with either a 64- or 128-bit data bus. The MPC620 interface protocol allows multiple masters to compete for system resources through a central external arbiter. Additionally, on-chip snooping logic efficiently maintains data-cache coherency for multiprocessor applications. The MPC620 supports single-beat and burst data transfers for memory accesses and memory-mapped I/O accesses [3].

On the PowerMANNA node computer each processor has its own 2-Mbyte second-level cache running with the 180 MHz processor clock. The interleaved and pipelined node memory of up to 1 Gbyte uses cheap standard DRAM modules and provides an access bandwidth of 640 Mbyte/s. The PowerMANNA node computer, shown in Figure 1, fits onto a single 29x23 mm^2 board. Each node can, if required, be extended by a PCI (Peripheral Component Interconnect) bridge with two PCI mezzanine slots (PMC-P1386.1) to connect required peripheral devices like disks, 3D graphics or LAN network controllers.

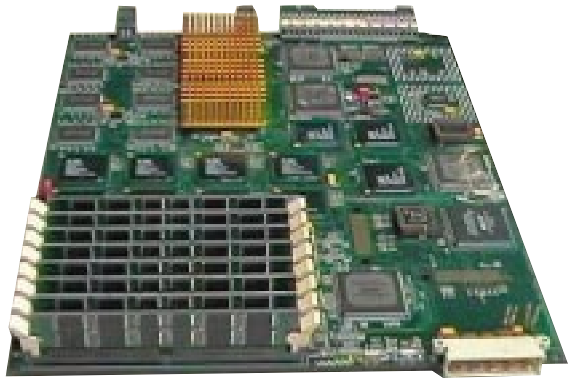


Figure 1. The PowerMANNA node computer

We discuss below how the special features of the MPC620 processor support an efficient implementation of SMP nodes and the architectural solutions to cope with the board complexity of the single-board node computer

The incorporation of small SMP nodes in MPP systems is known to be of advantage (and is currently implemented in many cluster designs) because it reduces overall cost while at the same time offering a high local communication bandwidth. Despite the usually high spatial locality of applications, the communication requirements of SMP nodes increase according to their higher computing performance compared with single-processor nodes. In PowerMANNA this aspect is taken into account by providing two communication links on the dual-processor nodes. This feature not only provides sufficient communication bandwidth, it may also be used to have completely separate communication networks for system- and user-level communication.

In the SMP node design, the shared-memory bus typically becomes the system bottleneck as the number of installed processors increases. By applying standard architectural principles like pipelining and interleaving, and by implementing sufficiently large caches, the required memory bandwidth can easily be provided. However, the saturation on the processor/memory bus is caused not only by the traffic between processors and memory, but also by cache-to-cache transfers and by the data transfers for communication and I/O. Thus, an efficient sharing of the address and data path among the connected resources is needed. Here, the processor bus of the PowerPC MPC620 - supporting split transactions, pipelining, tagged out-of-order transactions and cache-to-cache transfers - provides optimal support to achieve maximum parallelism between the competing transfers. In addition, the MPC620 helps to overcome the limitations imposed by the bus-based snoop protocol, which implies that the address phases of the processors need to be sequentialized. The MPC620 efficiently supports the full MESI cache-coherence protocol and allows several outstanding snoop requests to be queued.

In summary, the features of the MPC620 processor bus provide – along with its efficient implementation of the MESI protocol – optimum support for the implementation of powerful SMP nodes. The current PowerMANNA node includes only two processors. During the design phase we performed detailed simulations which showed that the actual node design would support up to four processors without their significantly hindering one another [4]. We found that the limiting factor is not the bandwidth of the node memory (thanks to its efficient implementation) but the sequentialization of the address phases enforced by the snoop protocol of the MPC620 processor.

The aim of achieving maximum node performance by supporting all MPC620 features requires innovative solutions to handle the resulting complexity. Two major design decisions allowed the PowerMANNA node to be implemented as a single-board computer without sacrificing performance: (i) Instead of conventional address and data buses, the node architecture features an integrated implementation of a multi master bus switch to which all devices are connected, and (ii) a single central control unit – the dispatcher – handles all the complexity of the MPC620's control signals and protocols and provides a simplified interface to all other node devices.

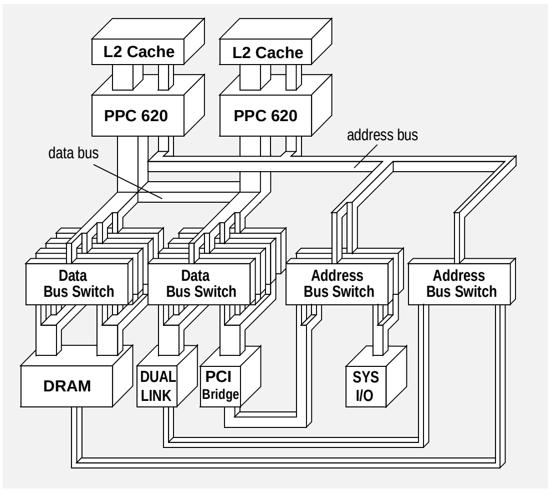


Figure 2. Address and data path architecture

We achieved a dense implementation of the multi master bus switch, despite its wide address and data path, by using a specially designed ADSP gate array. A single ADSP (address data path switch) chip contains a 36-bit slice of a three-way bus switch. Figure 2 shows the address and data path architecture of the PowerMANNA node based on 11 ADSP slices.

The PowerMANNA dispatcher handles the protocol and control complexity of the MPC620 processors. Pipelining, split transactions, intervention, out-of-order bus-transfer completion as well as the snoop protocols are kept transparent to the other units of the node, which simplifies their design significantly. In addition, the concept of the central dispatcher results in point-to-point connections of the control lines, thus avoiding long control buses connecting all the devices. The block diagram in Figure 3 shows the central PowerMANNA dispatcher with its interfaces to all other units of the PowerMANNA node. The unique concept of the central bus dispatcher handling the complex data-transfer protocols of the processors by controlling the ADPS devices is the subject of a patent application (patent pending).

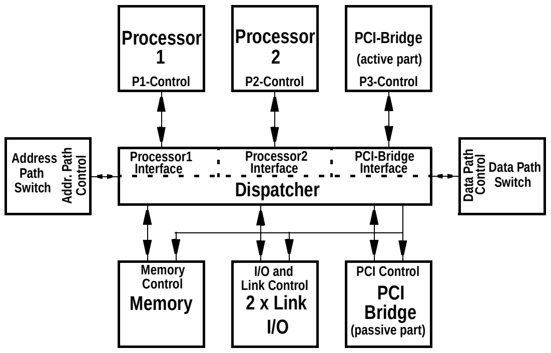


Figure 3. The PowerMANNA dispatcher

3. Communication system

As the performance of parallel computers depends strongly on the performance of the communication system, in the past complex and expensive interconnection topologies such as the hypercube have been employed. Less expensive mesh topologies, however, as used in the PARAGON or Cray T3E systems, exhibit a poor blocking behavior [5]. Communication networks based on crossbars are able to provide the favorable blocking behavior of the hypercube at much lower cost by replacing the link complexity of the hypercube (expensive connectors and cables) with the switching complexity of crossbars (inexpensive silicon) [6].

The topology of the scalable interconnection network of PowerMANNA is formed by a hierarchy of crossbars. A well-known example of crossbar structures is the fat-tree topology of the CM-5 [7]. The main difference between the two networks is the limited routability of the CM-5's 8x8 crossbar. Its input ports can only be routed to output ports of a different level in the tree. PowerMANNA's 16x16 crossbar device allows each input to be routed to all other output channels. This provides an almost unlimited flexibility and scalability and allows the realization of a wide variety of communication structures, as shown, e.g., in Figure 5b. In addition, by using newer technology, PowerMANNA's byte-wide communication links are three times faster and the bandwidth of its duplicated network interface is six times higher than in the CM-5.

3.1. Crossbar

PowerMANNA's crossbar device integrates all the FIFO buffers and the command- and address-decoding logic for each input channel and the arbiters for the output channels into a single ASIC. It implements a wormhole routing protocol and supports soft flow control on each connection. The setup of a logical connection is initiated by a *route* command. If there are no collisions, this through-routing takes only 0.2 microseconds. The route command (one byte), containing the address of the output channel, is consumed within the crossbar. Building up a logical connection across several crossbars therefore requires as many route commands in the message header as there are crossbars involved. Logical connections are closed by sending a single *close* command at the end of the message.

3.2. PowerMANNA Link Protocol

The PowerMANNA link is a clock-synchronous, byte-parallel, bidirectional point-to-point connection operating at 60 MHz. Each port simultaneously supports incoming and outgoing connections at up to 60 Mbyte/s (120 Mbyte/s full-duplex). This full-duplex protocol improves not only the overall bandwidth but also simplifies the communication protocols by excluding deadlocks.

To achieve an efficient implementation of the communication channels (links), an extremely lightweight protocol – suitable for complete implementation in hardware – was defined. Physically, each link direction consists of a 9-bit-wide channel for data and control information from the sender to the receiver, and a *stop* signal in the opposite direction (Figure 4). Together with the FIFO buffers on the receiver side, the stop signal is used for soft flow control.

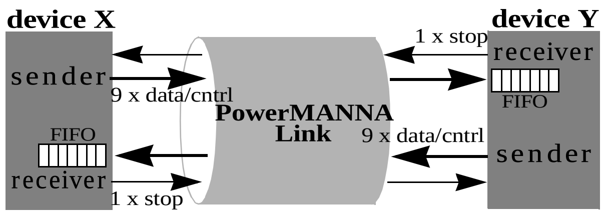
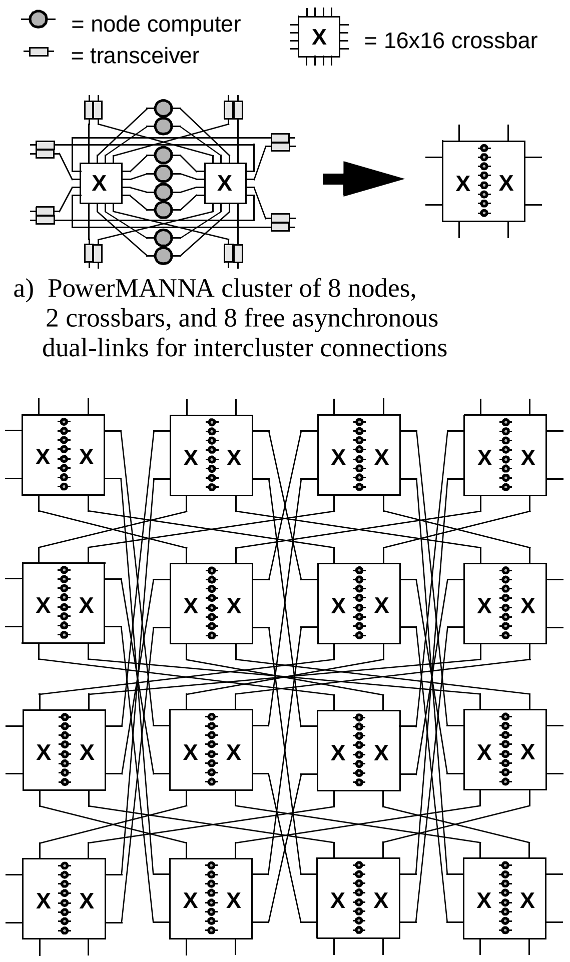


Figure 4. Link protocol

The link protocol is defined to connect the PowerMANNA nodes to the crossbars of the network, but also to connect the links of the crossbars or of the nodes directly to each other. Physically, the clock-synchronous link protocol is limited to short distances, e.g. within a cabinet. To bridge the greater distance between cabinets (up to 30 m) asynchronous transceivers have been implemented. On the input side of the transceivers, there are asynchronous FIFO buffers with 2-Kbyte entries allowing soft flow control over a longer distance.


= node computer
= transceiver
X
= 16x16 crossbar
a) PowerMANNA cluster of 8 nodes, 2 crossbars, and 8 free asynchronous dual-links for intercluster connections

b) PowerMANNA System with 256 Processors

Figure 5. PowerMANNA topologies

Figure 5.a shows the basic interconnect of an eight-node PowerMANNA system assembled into a desk-side cabinet. The crossbars, transceivers and connectors are assembled on the backplane, to which the eight single-board node computers and the cables of the asynchronous links are connected. Figure 5.b gives an idea of how larger systems might be configured from eight-node systems interconnected by asynchronous links. Basically, the topology consists of two permutation networks connecting the clusters of the rows and the columns of the 128-node system. Note that each line in Figure 5.b represents a duplicated link connection providing a total bandwidth of 240 Mbyte/s, and that a logical connection between any two nodes involves at most only three crossbars.

Compared to clusters, the backplane-integrated network of PowerMANNA requires only a small number of additional cables to build larger systems, and thanks to the smart single-board nodes, it is easy conceivable that systems with more than a thousand nodes could actually be handled

3.3. Network interface

The main design goal of the network interface was to minimize the communication overhead, especially for short messages. Instead of using a complex network interface controller (NIC), we implemented a simple but fast interface to the network.

The interface between the CPU/memory bus of the nodes and the network is implemented by a special ASIC. As defined by the link protocol, the sending and receiving of messages can be performed simultaneously. For each direction, there is a FIFO buffer of 32 64-bit words to decouple the different transfer rates. The addressing of the FIFOs and the control registers of the two link interfaces in a node is memory-mapped, so the CPUs of the SMP node can provide all the functionality of a powerful NIC by directly accessing the link interface.

We now briefly discuss the arguments supporting our design decision. A standard microprocessor is always cheaper than a specially designed network controller of comparable functionality, and as long as there are no communication requests, the processor contributes to the computing performance of the system. Assuming that the node CPU is able to copy messages between the network and the node memory at least at the speed of the network, the DMA of a separate network controller cannot speed up the transfer of messages. Here, the network interface of the CM-5, which implements a very similar concept, suffers from the slow copy performance of the node CPU. The 32-MHz SPARC processor utilizes less than 25% of the 20-Mbyte/s unidirectional transfer rate of the CM-5's data network [8].

Furthermore, the argument about the possible parallelism of a separate network controller is no longer a crucial factor because there is already sufficient parallelism between the CPUs of an SMP node. In addition, the on-chip MMU of the CPU allows the data to be transferred directly between user address spaces without additional copying. Especially in user-level communication, no system calls are required, either to translate logical to physical addresses or to pin pages used for communication, as is necessary, e.g., in Myrinet-based systems [9]. Also protection can more easily be incorporated because the CPU along with its MMU is involved in all communication. Another issue that has to be considered are the large caches supported by modern CPUs. The data to be sent may have be computed beforehand and may therefore still reside in the cache. Similarly, after a message has been received by the CPU, the data is already available in the cache for further computations. By contrast, a dedicated network controller transfers the data between node memories. The potential benefits from caching communicated data depend, of course, on the specific behavior of the application and cannot be easily quantified by simple benchmarks.

As shown by the communication benchmarks in Section 5.2, the PowerMANNA node fully exploits the bandwidth provided by the communication links, while at the same time requiring minimum setup costs. In addition to the protocol conversion, the link-interface chip performs generation and checking of a CRC check sum, ensuring that communication is not only an efficient but also a reliable.

4. PowerMANNA software

Users' demands with respect to standard software are taken into account by providing the LinuxPPC operating system [10], offering standard software environments for FORTRAN, C and C++ applications. Interprocess communication is supported by both the PVM and MPI message-passing libraries.

To obtain maximum benefits from the low-latency communication system, an optimized implementation of MPI offers user-level communication, which reduces the communication overhead significantly. In a first implementation, one part of the duplicated network is used exclusively for user-level communication, while the second part is reserved for Linux. This straightforward implementation required no modifications for Linux. In future work, we will implement a low-level protocol to coordinate the link access between the operating system and the application so that both links are available for application communication and the communication bandwidth can be fully exploited. However, dispensing with a separate control network for Linux would appear to be acceptable only when a single user is using the whole machine.

5. Performance evaluation

In this section, we evaluate the node and communication performance of PowerMANNA (by real runs on the machine), comparing its performance with other SMP nodes and a PC cluster system. The reference machines are a two-way SUN ULTRA-I and a PC cluster based on two-way Pentium II nodes connected by Myrinet. Table 2 shows the characteristics of all three systems.

Table 1. Configuration of test systems

System Type	**SUN**	**PowerMANNA**	**PC**
Processor Type	UltraSPARC-I	PPC620	PENTIUM II
Processor Clock	168 MHz	180 MHz	180/266 MHz
Bus Clock	84 MHz	60 MHz	60/66 MHz
Processors	2	2	2
Primary Cache	16/16 Kbyte	32/32 Kbyte	16/16 Kbyte
Secondary Cache	512/512 Kbyte	2/2 Mbyte	512/512 Kbyte
Cache line	32 byte	64 byte	32 byte
Node Memory	576 Mbyte	512 Mbyte	128 Mbyte
Operating System	Solaris 2.5	Linux	Linux

To facilitate comparison of the node architecture, we configured the PC board to run – in addition to the original speed of 266 MHz – at the same clock speed as the PowerMANNA node (180 MHz for the processors, and 60 MHz for the board frequency).

5.1. Node Performance

5.1.1 Single-processor performance. To evaluate single-processor performance, we use dual processor nodes of PowerMANNA, the SUN as well as the Pentium PC in order to establish equal constellations for all three machines (i.e., to take into account the slight performance drops potentially resulting from the coordination of the two processors). We show the results of the HINT benchmark and of the NASPAR MatMult benchmark.

HINT [11] measures the performance of both processor and memory hierarchy under the assumption that performance is basically memory-bound. The benchmark performs a simple calculation (Hierarchical INTegral calculation) which is – unlike most existing benchmarks – scalable and linear in computation and memory consumption. The calculation approximates $\int_0^1 (1-x)/(1+x)$ by successively refining intervals (in order of the largest removable error) into an integer power of two equal subintervals. Then, the maximum number of inside (lower-bound) and the minimum number of outside (upper-bound) squares are calculated. The logs describing the intervals and the bounds calculated for them constitute the information potentially cached, and this is accessed in more complex ways than just a consecutive order. The quality of the solution is the reciprocal of the difference between the upper and lower bounds. Because of self-similarity properties, the integration method is linear in its quality improvement. The ratio of operations to storage is almost one to one – which is lower than in many other benchmarks and considered by the HINT designers to be more realistic with respect to real applications' memory access. The quality obtained is order N for order N storage and order N operations. The metric used is progress in work, measured in so-called QUIPS (Quality Improvement Per Second) and is given along the runtime of the benchmark. The left side of the curve shows maximum processor performance (all data in the cache), and the right side of the curve memory bandwidth. The further left the curve starts, the lower the latency of the system. The performance curve indicates when specific parts of the memory hierarchy – i.e., the L1 and L2 cache – are fully in effect (start phase up to maximum of curve) or lose their support effect (the sharp drops in the decreasing phase of the curve) because the current working set becomes too large, main-memory access ultimately dominating. The benchmark can run with different data types, partially revealing a machine's particular orientation toward numerical computation or data processing. (Data is represented as whole numbers, and either floating-point or integer arithmetic can be used for the calculations.)

We used the data types DOUBLE and INT in our tests. Figure 6 shows the results obtained for the PowerMANNA node, the SUN and the Pentium cluster node. In the case of the Pentium, we present performance for both its original and its reduced clock rate. For data type DOUBLE, HINT performance is slightly better for PowerMANNA than for the reduced-clock-rate Pentium PC as long as caches are in effect, but the reverse is true for the memory-access region. The main reasons for this are the missing load/store pipeline and lower benefits from the cache (explained below). For data type INT, PowerMANNA and the Pentium PC perform almost equally well, both outperforming the SUN. Generally speaking, PowerMANNA and the Pentium PC perform even better for INT than for DOUBLE, the SUN's performance being lower. On the basis of the results, PowerMANNA (and the PC cluster, too) can truly be considered machines for both numeric and nonnumeric processing.

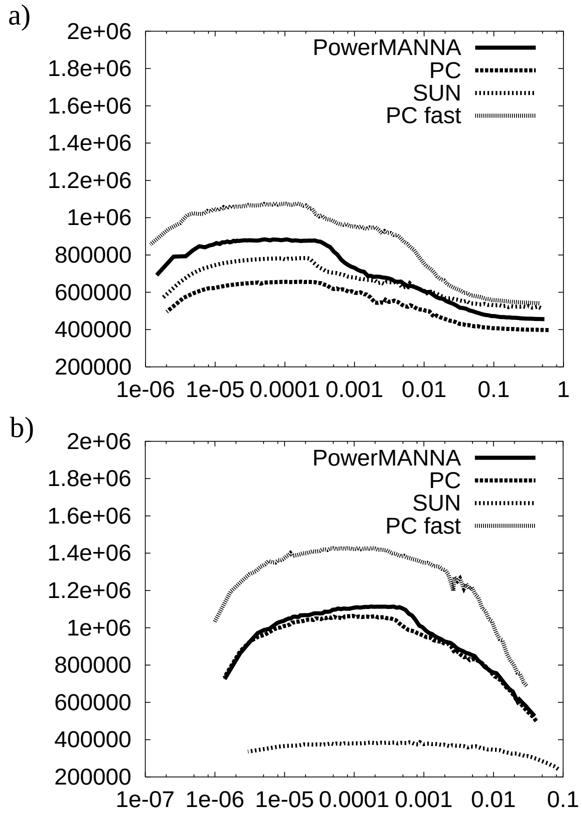
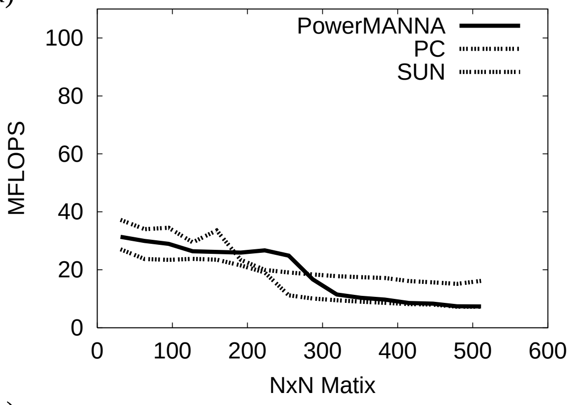


Figure 6. Performance of the three machines' nodes for HINT benchmark and data types a) DOUBLE and b) INT. Performance in QUIPS along time in seconds.

As HINT is unable to differentiate subtle features like floating-point instruction sequences, memory-access patterns or cache-line sizes, we use the MatMult matrix-multiplication benchmark to evaluate the processor and cache features in more detail. MatMult is run in two versions: a) naive matrix multiplication with both matrices allocated in row order, and b) transposition of the second matrix (to allocate it in column order) and multiplication of the first by the transposed second matrix (runtime includes the transposition). These two versions show the specific features of the PowerMANNA caches (see Figure 7).

a)



b)

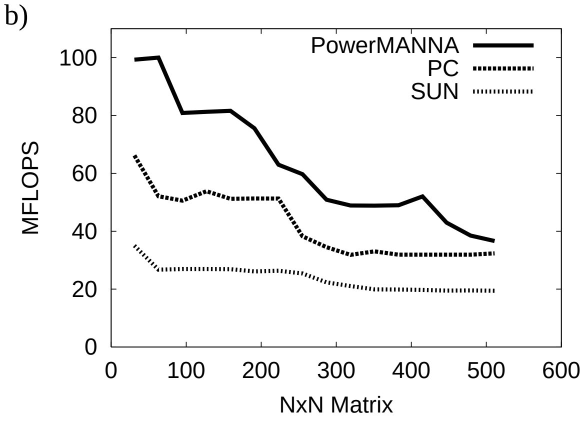


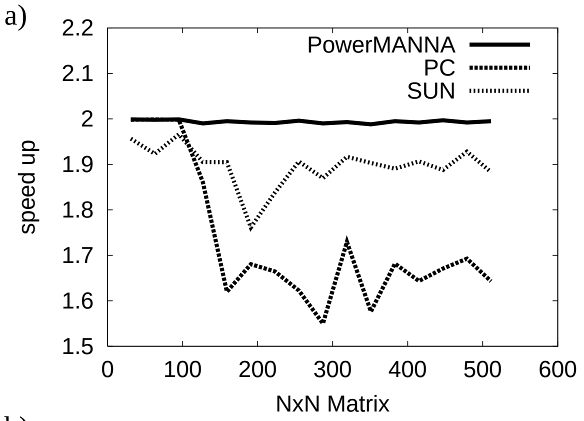
Figure 7. Single-processor performance of the three machines' nodes for MatMult benchmark, odd strides. a) naive version and b) transposed version. Performance is measured in MFLOPS along increasing matrix sizes.

Note that the PowerMANNA's cache line is twice as long as the cache lines of the SUN and the PC. Here, we used the reduced-clock-rate Pentium PC. It is important to mention, too, that the PowerPC MPC620 is specially designed to support floating-point pipelining, but it does not support load pipelining (the follow-up processor Power3, however, incorporates this). Thus, the available memory bandwidth of PowerMANNA cannot be fully exploited.

In case b), the cache and the cache-line prefetching can be fully exploited, accesses then being to a large extent to data in the cache. Here, PowerMANNA clearly outperforms the other machines. In case (a), accesses are spread across memory and caches are able to offer much less support. In particular, the large amount of prefetching resulting from the longer cache lines of PowerMANNA is not effective here and transfers superfluous data. Performance for all machines is significantly worse than in case (b). However, the difference is largest for PowerMANNA: a factor of approx. 2.5 for small matrices and a factor of approx. 6 for large matrices. The Pentium PC performs best here. However, as long as caches are effective, the difference between the Pentium PC and PowerMANNA is small in case (a), while the absolute performance advantage of PowerMANNA in case (b) is much larger.

5.1.2 SMP performance. Next, we evaluate the SMP scalability of the three architectures and investigate potential conflicts of the two processors on memory access. We use, again, the MatMult benchmark and measure it when started on both processors. Figure 8 shows the speedups obtained.

a)



b)

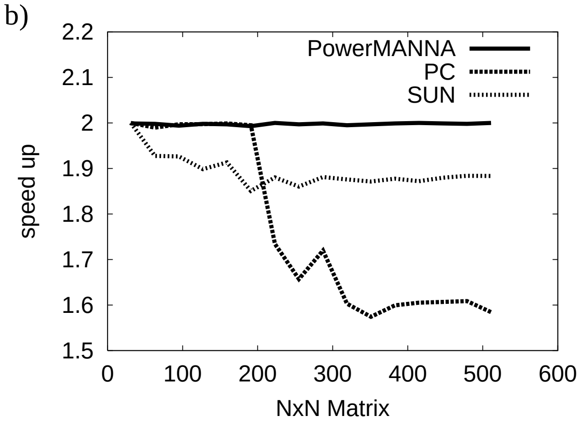


Figure 8. Dual-processor speedup of the three machines' nodes for MatMult benchmark, odd strides. a) Naive version and b) transposed version.

As expected from the split-phase transaction feature of the MPC620 processor, performance for PowerMANNA exactly doubles when running the benchmark on both processors of the node. In other words, there is no memory-access contention. For the SUN, there is a loss of about 5%, i.e., speedup is about 1.9 for nontrivial matrices. For the Pentium PC, the loss is 15% (naive version) and 20% (transposed version), i.e., the speedup is about 1.7 or 1.6, respectively, for nontrivial matrices.

5.2. Communication performance

We compared the communication performance of PowerMANNA and the PC cluster in terms of set-up times, one-way latencies, unidirectional and bidirectional throughput on a two-communicating-node basis. For the PC cluster, we employed the user-space communication libraries BIP and FM (cf. [9] and [12]). BIP (Basic Interface for Parallelism) is a minimal library that aims to provide raw hardware performance to its users. FM (Fast Messages) provides software flow control. Performance data for BIP and FM are taken from [9] because the data obtained from our Linux 2.2, for which only the GM library currently offers support, were too slow for a fair comparison. Data taken from [9] is measured under the same test conditions, i.e. on a cluster with Myrinet interconnect and Pentium Pro processors running at 200 MHz.

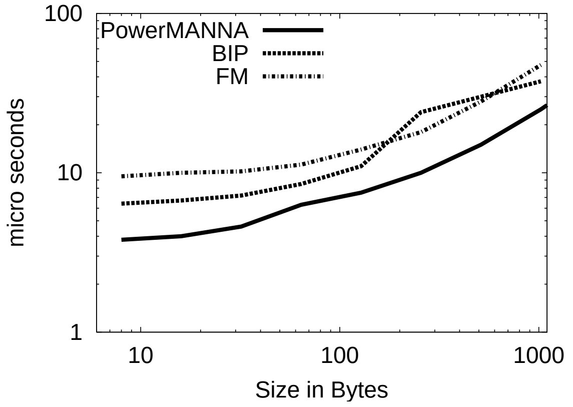


Figure 9. One-way latencies for PowerMANNA and for BIP and FM on the PC cluster.

Figures 9 and 10 show one-way latencies and message-sending times at the network saturation point, corresponding to half of the ping-pong time and the gap parameter in the LogP model [13]. As can be seen, PowerMANNA clearly outperforms the other systems for short messages. For instance, 8 bytes are transferred in 2.75 µs, whereas BIP takes 6.4 µs and FM 9.2 µs. For larger messages, however, PowerMANNA's performance is limited by its current network technology to 60 Mbyte/s unidirectional single-link bandwidth (see Figure 11).

Figure 12 shows the bidirectional bandwidth, when both nodes are simultaneously sending and receiving messages. For short messages, the bidirectional bandwidth is similar to BIP and Myrinet. For longer messages, however, we did not obtain the expected bandwidth. Apparently, PowerMANNA suffers from too small FIFOs in the link interface. The communication driver can send at most 4 cache lines to fill the send-FIFO. Then the driver has to test the receive-FIFO and possibly receive the incoming data. After a maximum of 4 cache lines, the receive-FIFO is emptied and the driver must switch directions again. This overhead could be significantly reduced if larger FIFO buffers were implemented.

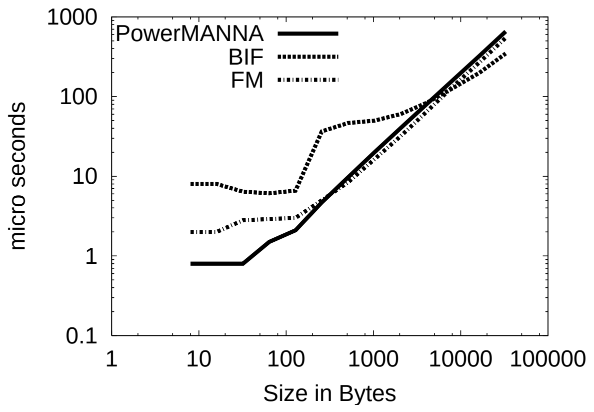


Figure 10. Message-sending times at the network saturation point for PowerMANNA and for BIP and FM on the PC cluster.

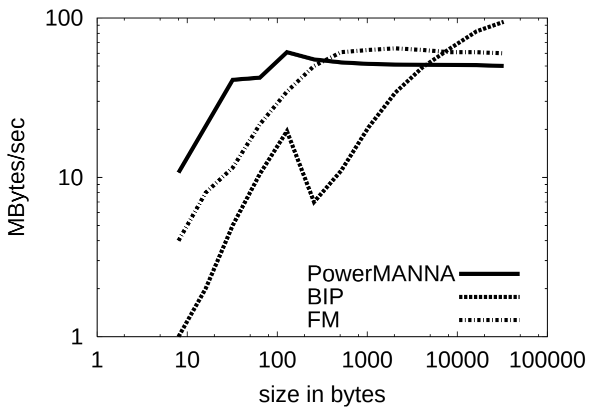


Figure 11. Unidirectional bandwidth for PowerMANNA and for BIP and FP on the PC cluster.

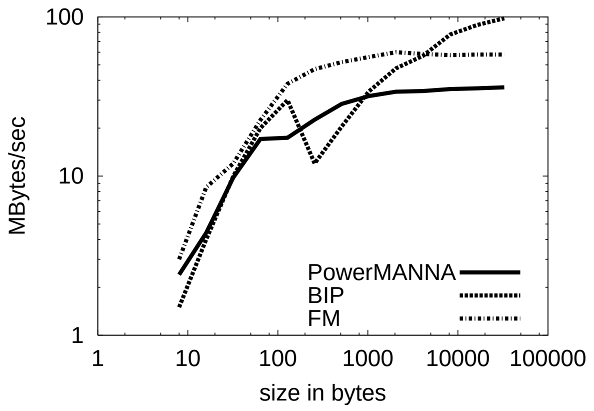


Figure 12. Simultaneous bidirectional bandwidth for PowerMANNA and for BIP and FP on the PC cluster.

6. Related Systems

An architectural design very similar to PowerMANNA, and also based on the MPC620 and a hybrid SMP/MPP concept, is START-NG [14]. The processing nodes contain 4 processors, each with its own network interface. The high-performance and low-latency communication system consists of 4x4 crossbars implemented into the Arctic switch fabric as a central unit to which all nodes are connected. The memory-mapped communication interface exploits the MPC620 L2-cache/coprocessor bus, resulting in a lightweight PIO-based communication comparable to PowerMANNA. However, owing to the delay in the delivery of the PowerPC MPC620 processors, this machine was never build.

In terms of the CM-5's topology and its lightweight communication interface, PowerMANNA takes a similar approach. In addition to its data network, the CM-5 is equipped with a sophisticated control network [7].

The processing nodes of the Intel Paragon are SMP nodes based on up to 3 Intel i860 XP processors. One of the processors is used as a dedicated communication processor, supported by a DMA unit. To communicate, the compute processors of the nodes have to initialize the communication processor, which in turn has to initialize the DMA unit. This results in large set-up times and prevents the easy implementation of user-level communication [15].

IBM's SP series also boasts several features similar to PowerMANNA. The two-way processor nodes are based on either the Power2, the PPC604, or – not so long ago – the Power3 processors. The communication system is based on the SP switch, a permutation network built from 4x4 crossbar components. The nodes of the SP series require several boards: for the processors, memory and the network interface. The network interface board includes a dedicated controller based on an embedded processor [16].

The Cray T3E [17], currently used in many of the Top100 best-performing systems, demonstrates the performance benefits obtainable from integrated parallel distributed-memory machines. The T3E provides single-processor (DEC Alpha EV5) nodes, sophisticated network-interface controllers, a very-low latency communication network (3D Torus) yielding a transfer rate of 600MB/s, an additional synchronization network (barrier and Eureka), as well as special multicast scatter/gather hardware (vector masks on the network interface) and software support. Thus, the T3E specifically supports communication and synchronization in SPMD programs, whereas PowerMANNA is not geared to a specific programming model and can also perform well with multithreaded software.

Building large PC clusters – e.g., of Pentium-based SMP nodes connected by a SAN such as Myrinet with several hundred nodes – requires substantial effort to achieve a reliable assembly as regards cabling, cooling and power supply. In addition the cost of the network is at least in the same order as the nodes themselves. Thus, the price/performance ratio of PC clusters as compared with parallel distributed-memory machines may not be as superior as expected if the system has to meet the reliability and maintenance requirements within an industrial computing center.

Myrinet is a widely used standard communication system for cluster computing. Its 1.2 Gbyte/s transfer capability is exploitable up to 132 Mbyte/s in view of the PCI interface of the network interface controller (NI). Myrinet uses 8x8 crossbar switches and involves extra NI boards. However, Myrinet incorporates some features previously found only in super computers, such as variable-length packets, wormhole routing and hardware backpressure. Messages have to be additionally transferred between the processor and the NI which can be performed either via DMA or PIO, but in any case involves extra setup cost. Transfers from NI to NI always require setting up a DMA unit because of the slow copying performance of the NI processor. Separation of processor and NI also creates the address-translation problem mentioned earlier. Though dynamic pinning of pages can be used, it is less flexible and requires an initial system call, some kind of virtual-address translation and potential NI caching [9]. However, close-to-optimum throughput for large data transfers can be achieved with user-level communication, as performed, e.g., by BIP, FM and PM [9], [12].

7. Summary and Future Work

We have described the architectural features of PowerMANNA, a distributed-memory parallel computer system based on the PowerPC MPC620. Many features of the PowerPC MPC620 processor discussed in relation to PowerMANNA are typical of modern microprocessors that support the straightforward design of powerful SMP nodes. Performance evaluation showed ideal scalability of the PowerMANNA SMP node. Single-processor performance is superior where the long cache lines can be exploited, being otherwise comparable to Pentium Pro for same clock rate. With respect to communication, PowerMANNA outperforms clusters in terms of setup time and latencies for short data transfers.

The presented architectural principles of PowerMANNA are independent of the actual technology and will be able to benefit from future advanced CPUs, memories and networks. Thus, the follow-up processor, Power3, could be immediately plugged in (provided it were made commercially available). Current performance rates for the PowerMANNA promise excellent results provided the latest processor and network technology is used to implement the design.

A distinctive feature of the PowerMANNA architecture is the smart network interface of the nodes. Its complexity lies below that of an elaborate communication assist – as in the Cray T3E – or that of a dedicated message-processing unit that is based on an embedded processor and interfaced to the node via a standard I/O bus – as in Myrinet clusters. By contrast, PowerMANNA benefits from a lightweight communication protocol that is directly driven by the node CPUs. This arrangement avoids the cost of a communication assist and the software overhead of an interface protocol between the node and the network controller. Furthermore, it allows advantage to be taken of ever-increasing processing power and data-cache size of next-generation CPUs. In addition, driving the communication protocols directly with the node CPUs supports the implementation of pure user-level communication without any involvement of the operating system. Compared with clusters, the communication system of the PowerMANNA design is better scalable because communication cost can be kept lower and cabling effort is minimal.

The design of lightweight hardware communication systems should be accompanied by lightweight communication software in order to fully exploit the special communication-cost and scalability benefits. Thus, for the forerunner MANNA machine, the EARTH system was shown to offer low communication cost close to the hardware limits [18]. In a cooperation project with the University of Delaware, EARTH is currently being ported to the PowerMANNA machine.

The performance of PowerMANNA was evaluated on the basis of micro benchmarks. To show real-application performance, we have to port the SMP version of Linux (which has only recently become available for the PowerPC) and either use EARTH or optimize the MPI user-level communication protocols. Then, we can, in particular, investigate to what extent application performance can benefit from caching communicated data and from the short set up times and low latencies provided by the lightweight communication protocol.

8. Acknowledgments

The authors would first of all like to thank W.K. Giloi, the PowerMANNA project's principal investigator. Our thanks also go to the PowerMANNA research team, B. Brüning, R. Hänisch, E. Jara-Rios, F. Kujawsky, F. Mousa, M. Nguyen-Luong, B. Oestmann, K. Otter and N. Pieth, for their contributions to the PowerMANNA project.

9. References

[1] http://www.top500.org/lists/1999/06/top500.list.html

[2] Power3 Whitepaper: http://www.rs6000 .ibm.com/resource/technology/power3wp.pdf.

[3] IBM/Motorola. PowerPC 620: Red October Microprocessor Implementation Definition, Book IV, Version 1.0, 1996

[4] Wolfgang K. Giloi, Christoph Lindemann, and Samuel Pletner, "Performance Modeling of Novel Node Architectures for Multiprocessor Systems", Proc. 11th Int. Symp. on Computer and Information Sciences, Antalia, November 1996.

[5] Govindan Ravindran and Michael Stumm, "A Performance Comparison of Hierarchical Ring- and Mesh-connected Multiprocessor Networks", Proc. Int. Symp. on High Performance Computer Architecture, Feb. 1997.

[6] Wolfgang K. Giloi and Sergio Montenegro, "Choosing the Interconnect of Distributed Memory Systems by Costs and Blocking Behavior", Proc. of the 5th International Parallel Processing Symposium (1991), IEEE Catalog No. 91TH0363-2, pp. 438-444

[7] Charles. E. Leiserson, et al., "The Network Architecture of the Connection Machine CM-5", 4th Symp. Parallel Algorithms & Architectures, 1992, pp. 272-285.

[8] E.A. Brewer and B.C. Kuszmaul, "How to Get Good Performance put of the CM-5 Data Network", Proc. of the 1994 Int.Parallel Processing Symp., 1994, pp 858-867.

[9] Raoul Bhoedjang and Tim Rühl and Henri E. Bal, "User-Level Network Interface Protocols", IEEE Computer, Vol. 31, No. 11 (Nov. 1998), pp. 53-60.

[10] Focus: Linux. Collection of Articles. IEEE Software, January-February 1999; Linux software and documentation, available at www.linux.org.

[11] John L. Gustafson and Quinn O. Snell, "HINT: A New Way To Measure Computer Performance", HICS'95, Jan. 1995.

[12] Soichiro Araki, Angelos Bilas, Cezary Dubnicki, Jan Edler, Koichi Konishi, and James Philbin, "User-Space Communication: A Quantitative Study", Proc. Supercomputing, Orlando, FL, USA, ACM/IEEE Nov. 1998.

[13] David Culler, Richard Karp, David Patterson, Abhijit Sahay, Klaus Erik Schauser, Eunice Santos, Ramesh Subramonian, and Thorsten von Eicken, "LogP: Towards a Realistic Model of Parallel Computation", ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, San Diego, CA, May 1993.

[14] Derek Chiou, Boon S. Ang, Arvind, Michael J. Beckerle, Andy Boughton, Robert Greiner, James E. Hicks, and James C. Hoe, "START-NG: Delivering Seamless Parallel Computing", CSG Memo 371, MIT, Laboratory for Computer Science, July 1995, and Proc. Europar'95, Stockholm, Sweden, 1995.

[15] Rudolf Berrendorf, Heribert C. Burg, Ulrich Detert, Rüdiger Esser, Michael Gerndt, Renate Knecht, "Intel Paragon XP/S - Architecture, Software Environment, and Performance", Internal Report KFA-ZAM-IB-9409, Forschungszentrum Jülich GmbH, May 1994

[16] IBM POWERparallel Technology Briefing: Interconnection Technologies for High-Performance Computing (RS/6000 SP) http://www.rs6000.ibm.com/resource/technology/

[17] Wilfried Oed, "Massiv-paralleles Processorsystem CRAY T3E", Technical Documentation, Cray Research, München, 1996.

[18] Herbert H. J. Hum , Olivier Maquelin, Kevin B. Theobald, Xinmin Tian, Guang R. Gao, and Laurie J. Hendren, "A Study of the EARTH-MANNA Multithreaded System", Internat. Journal of Parallel Programming, Vol. 24, No. 4, 1996.