A PCI-SCI Bridge for Building a PC Cluster with Distributed Shared Memory

Georg Acher, Hermann Hellwagner, Wolfgang Karl, Markus Leberecht

Lehr- und Forschungseinheit für Rechnertechnik und Rechnerorganisation / Parallelrechnerarchitektur (LRR-TUM)
Institut für Informatik, Technische Universität München
80290 München, Germany
Tel.: +49-89-289-28278, Fax: +49-89-289-28232
Email: {karlw|hellwagn|leberech}@informatik.tu-muenchen.de

Abstract

In 1995, the SMiLE (Shared Memory in a LAN-like Environment) project at LRR-TUM started to set up a SCI-based PC cluster with distributed shared memory (DSM). This system will serve as a platform for experimental research and investigations into the efficient use of a DSM parallel machine. For this reason, we decided to develop and implement our own PCI-SCI bridge, which is targeted to plug into the PCI bus of a PC. Pentium-PCs equipped with our bridge will be interconnected to a cluster of computing nodes, the SMiLE multiprocessor. This paper describes the technical features of this PCI-SCI interface implementation and outlines the software developments and research activities within the SMiLE project.

1 Introduction and Motivation


For example, SCI is being used in tightly coupled multiprocessor systems like the Hewlett Packard / Convex Exemplar [2], or in symmetric multiprocessor systems (SMPs) or clusters of SMPs announced from Sequent and Data General. The Local Area Multiprocessor (LAMP) model [8] incorporates workstations, servers, specialized processors or data acquisition systems with the Scalable Coherent Interface as interconnect technology. High-performance distributed and parallel cluster computing can be achieved with SCI hardware that is available now or in the near future. Dolphin’s SBus-SCI adapters [4] [3] or SCI switches can be used to set up a cluster of workstations. Additionally, a PCI-SCI adapter has been announced by Dolphin. At CERN, a PCI-SCI bridge has been developed which is targeted for constructing a demonstrator data acquisition system for the Large Hadron Collider experiments [15].

Exploiting SCI’s flexibility in terms of efficiently supporting both parallel programming models – message-passing and shared-memory – leads to investigations targeting a distributed shared memory multiprocessor system similar to Stanford’s FLASH [12], or MIT’s Alewife [1] machines.

With this background in mind, the SMiLE (Shared Memory in a LAN-like Environment) project [9] at LRR-TUM started in 1995 to set up a SCI-based cluster of PCs with distributed shared memory, and to develop software for this system. The long-term objective of research is to design and implement an appropriate distributed shared memory programming model and to investigate the efficient use of a parallel computer with NUMA characteristics (non-uniform memory access).
The key to efficiency is to provide and exploit data locality and to avoid remote memory accesses as far as possible. Monitoring the behaviour of data accesses at runtime and using the gathered information to move or replicate data and/or migrate processes will improve locality of data accesses and, thus, increase the efficiency of a DSM machine.

Although there exist PCI-SCI bridges (e.g. at CERN [15]) or will become commercially available in the near future, we decided to develop and implement our own PCI-SCI interface within the SMiLE project. This bridge is targeted to plug into the PCI bus of a PC. Pentium-PCs equipped with our bridge will be interconnected to a cluster of computing nodes with distributed shared memory, the SMiLE multiprocessor system. In contrast to the announced competing interfaces, however, our bridge will be equipped with additional monitoring hardware, enabling research on data locality in NUMA systems.

The paper gives an overview of the SMiLE project at LRR-TUM and its objectives. It describes in detail the design and implementation of our PCI-SCI bridge targeted for clustering of PCI-based PCs into a distributed shared memory system. Additionally, preliminary performance results will be presented.

2 The SMiLE Project at LRR-TUM

Networks of workstations (NOWs) have become increasingly popular as widely available facilities for parallel processing. This has been fostered by the availability of public domain packages like PVM [6] or NXLib [17] as well as implementations of MPI [7], which allow a NOW to be utilized as a virtual parallel computer. In particular, small and medium enterprises, universities, and research labs follow this approach, thereby saving their existing investments in desktop machines. On the other hand, parallel computing with NOWs bears the drawbacks of insufficient communications performance over today’s LANs and message passing as the only programming model.

Here SCI’s functionality and performance help to alleviate these problems: up-to-date technology provides high throughput and low latencies while the shared address space and bus-like transactions create more degrees of freedom for programming models.

By exploiting the potential of SCI-based local area multiprocessors, the SMiLE project at LRR-TUM tries to meet the demands for low-cost, technically advanced, and powerful parallel computers.

The SMiLE project is divided into a hardware and a software part:

Development of a SCI-based PC Cluster with DSM

The SMiLE multiprocessor to be built up with industry-standard PCs interconnected by SCI technology will serve as main research vehicle. As already mentioned, since commercial PCI-SCI interface cards are (still) not yet available and often do not suit our research interests, we started our own hardware and system software developments. Aside from the PCI-SCI bridge hardware described in more detail in Section 3, other work in this area comprises the following:

- Development of a device driver for Linux using our own PCI card, providing transparent memory accesses and message passing style communication.
- Development of a network interface device driver for Windows NT using our own PCI card.
- Design of communications monitoring hardware and software to be fitted onto our SCI bridge.

Software Development for SCI-based Multiprocessors

We aim to provide a software basis for parallel processing on the SMiLE hardware. Work that has already been carried out includes:

- Port of PVM direct-routed data transfers over native SCI to support more efficient message passing programming on clustered workstations [10].
- Implementation of an Active Message [18][14] layer on top of SCI shared memory transactions [5].
- Port of a Unix Shared Virtual Memory (SVM) implementation to make use of the SCI interconnect for page transfers among nodes.

Since the SMiLE hardware has not yet been available, research work was carried out on a testbed consisting of two Sparcstation-2 computers connected through Dolphin’s SBus-1 SCI adaptors. Dolphin’s elementary driver-level SCI APIs [3] were used.
Research into Efficient Parallel Programming of DSM Systems

In the long term the SMiLE system will also be used for experimental research into a DSM parallel programming model. For efficiency of DSM programming, detection, and exploitation of data locality is the main issue. The aforementioned monitoring concept will allow us to study data access behaviour at runtime.

An alternative approach to dealing with long access latencies is the concept of multithreading that allows to make use of otherwise idle processor cycles. The Multi-threaded Scheduling Environment (MuSE) [13] is an experimental runtime system, the basic concept of which is to use a specific feature of hardware-supported communication (and thus the SMiLE architecture): buffered writes are exploited to gain execution time that would be lost otherwise when the processor stalls on reads. For this, non-blocking threads have to be scheduled in a dataflow style. Initial analyses indicate that this approach can result in highly efficient execution on the SMiLE DSM system.

3 The PCI-SCI Bridge Architecture

The PCI-SCI bridge serves as the interface between the PC’s I/O bus, the PCI bus, and the SCI network. SCI nodes are interconnected via the input and output links, so that ring-like connections are built with the output of one node providing the input to the input link of the neighbour node.

The SCI standard defines packet switched communication protocols. SCI split transactions require a request packet to be sent from one SCI node to another node and a response packet transmitted from the remote node back to the source node in order to complete the transaction. The PCI-SCI interface generates packets for remote SCI nodes, transmits incoming packets via the output link to the neighbour SCI node or directs them to the local user node. PCI address spaces of the bridge can be mapped into SCI, allowing PCI read/write accesses to any of the mapped SCI resources. The PCI-SCI interface is responsible for potentially required address translations and request/response packet generation. Such a transparent access is the key feature required for implementing shared memory applications.

As depicted in Figure 1, the PCI-SCI bridge is divided into three logical parts, the PCI unit, the Dual-Ported RAM (DPR), and the SCI unit. The PCI unit interfaces to the PCI bus of the local PC host, and the SCI unit interfaces to the SCI network. Data is transferred between the PCI unit and the SCI unit across the Dual Ported RAM (DPR), and control information is passed on the handshake bus. The PCI unit is connected to the DPR via a 32 bit wide internal bus, the DPR bus. The DPR and the SCI unit are connected via the B-Link, a 64 bit wide synchronous bus.

3.1 The PCI Unit

The PCI unit (Figure 2) has to translate PCI read/write transactions into bridge internal bus operations and vice versa. For this task, the PCI9060 chip from PLX Technologies [16] is used, which provides a PCI bus master interface for adapters. The chip’s local bus follows the Intel i960 microprocessor’s bus protocol. Using the i960 bus as the PCI unit’s local bus simplifies the design of the control logic considerably. No complex PCI compliant control logic is necessary. The PCI9060 supports both multiplexed and non-multiplexed local buses. Operating with the multiplexed bus mode for addresses and data can be exploited for the generation of SCI packets. The PCI9060 can act both as target and as initiator on both of its sides, thus supporting bidirectional transactions forwarding. Two independent bi-directional DMA channels are integrated, allowing direct memory transfers between PCI and SCI initiated by the PCI-SCI bridge. The PCI9060 local bus interface runs from a local TTL clock and generates the necessary internal clocks. This clock runs asynchronously to the PCI clock. This feature sim-
plifies the implementation, because the internal bridge bus can operate at a different speed than the PCI bus.

Figure 2: Block diagram of the PCI Unit

The SCI Upload and Packet Encoder Management Unit (SUPER\_MAN) is responsible for controlling and coordinating the translation of read/write accesses into B-Link packets and vice versa. It comprises several functional units, which are shown in Figure 3.

The functional units of SUPER\_MAN are controlled by the Packet MANagement unit (PAC\_MAN). It contains a microcode sequencer and a writeable control store (WCS) with up to 64 microinstruction words. Each microinstruction word is divided into 4 bit wide fields to directly and independently control the actions of every functional unit. In each microinstruction word, the WCS address of the next microinstruction to be executed is also encoded. This implementation allows all functional units to operate in parallel.

For example, microprograms exist for translating a PCI address into a SCI address and writing them into the DPR, for generating SCI commands from addresses or byte enables, and for transmitting of data to be written into the DPR.

The Local Bus MANagement unit (LB\_MAN) recognizes whether there is a request on the local bus. Depending on the state of the read/write and DMA signals, it determines which microprogram has to be executed. The LB\_MAN sends the appropriate control information to the TRANSAction Management unit (TRANS\_AM). This unit checks its transaction queues for outstanding requests whether there exist free entries and forwards the received control information to the PAC\_MAN. If a SCI packet arrives, a control command for the PAC\_MAN will be generated from the SCI command using a translation table.

The TRANS\_AM determines the DPR base address for the appropriate packet frame. It also coordinates between local bus and SCI requests and communicates with the corresponding unit within the SCI unit (the B-link Access and Transaction Management Unit).

The Data MANagement unit (D\_MAN) of the SUPER\_MAN generates SCI commands and performs the arithmetic operations for lock transactions. The Port, Access and Enable MANager (PAN\_AM) generates special DPR addresses and the chip-select and write-enable signals.

The SUPER\_MAN unit is implemented on a FPGA chip.

The ATC RAM of the PCI unit will be used for PCI to SCI address translations. Because of the multiplexed PCI unit’s local bus, data buffering is necessary (BUF). The PCI unit is connected to the DPR via a 32 bit wide bus, the DPR bus.
3.2 The Dual Ported RAM

The Dual Ported RAM (DPR) is used as data transfer buffer between the B-Link and the DPR bus. It also serves for queueing of SCI packets. The DPR contains 16 packet frames for outgoing `writesb`- and `nwrite64`-requests respectively, 32 for outgoing responses and one packet frame for `readsb` and `locksb`, respectively. Composing a SCI packet only requires writing variable data like addresses or data to be written into the appropriate fields. Additionally, 64 incoming packets can be buffered. Figure 4 shows the partitioning of the DPR.

![Figure 4: Dual Port Memory Partitioning](image)

The DPR consists of 4 IDT7124 chips which are organized as 4 K x 16 bytes per chip.

3.3 The SCI Unit

The SCI Unit (Figure 5) is connected to the DPR via the B-Link. The B-link Access and Transaction MANager (BAT_MAN) controls B-Link arbitration, reading or writing the DPR (on SCI unit side), and drives the B-Link control signals. It is also implemented on a FPGA chip.

Figure 6 shows the functional units of the BAT_MAN unit. The central unit is the Ram Operation and B-Link INterface unit (ROBIN), which controls the B-Link and reads or writes the DPR. The FIFOs are used to buffer the DPR start addresses of outgoing request or response packets. The LC_INIT module initializes the Link-Controller chip.

The physical layer and part of the logical layer of SCI are implemented by Dolphin’s Link Controller (LC). The LC provides the SCI input and output links on the SCI side. On the “back side” (non-SCI link side) of the chip is the B-Link.

![Figure 5: Block diagram of the SCI Unit](image)

![Figure 6: Block diagram of the BAT_MAN unit](image)
3.4 An Example

The following example describes the actions during a remote write.

The PCI9060 transforms the PCI write transaction into the PCI unit’s local bus operation. SUPER_MAN recognizes that a SCI write transaction has to be composed. With the help of the ATC, the PCI address will be translated into the address for the remote SCI node. The translated address will then be written into the DPR. Subsequently, the buffered data will also be written into the DPR. The number and the position of the data are used to generate the SCI command for single-byte writes.

SUPER_MAN informs BAT_MAN about the completion and the position of the B-Link request packet within the DPR. Simultaneously, a timer is started which stops upon receipt of the response packet. This feature allows lost packets to be identified.

BAT_MAN starts the transmission of the packet to LC via the B-Link. The LC directs the packet to its output link, thus sending the packet to the remote SCI node.

On the receiver SCI node, the LC directs the packet onto the B-Link. BAT_MAN writes the packet into an incoming packet frame within the DPR and informs SUPER_MAN about the arrival of a packet. SUPER_MAN extracts the SCI command from the packet in order to determine the actions to be done. The address will be extracted from the packet’s address field and put on the bridge internal bus. Afterwards, the data will be put on the internal bus and written into the remote memory by the PCI9060. After completion, a response packet will be generated and sent back to the requester, where the timer will be stopped and the packet frame will be released.

3.5 Development Environment

For the design and implementation of the PCI-SCI bridge the VHDL development environment of Synopsys and the FPGA software tools from Xilinx have been used. The SCI Upload and Packet Encoder Management Unit (SUPER_MAN) and the B-Link Access and Transaction Management Unit (BAT_MAN) were implemented for the Xilinx 4000E FPGA series.

3.6 Status and Initial Performance Results

The development of the PCI-SCI bridge has been completed and the prototype is now being tested.

The example in section 3.4 describes a remote write. This example will be used to illustrate the performance of the PCI-SCI bridge implementation. For the measurements, the PCI-SCI bridge operates in loop-back mode. The timings have been measured with an oscilloscope, monitoring the appropriate signals. The system clock and the B-Link currently run at 18 MHz, while the LC-1 runs at 100 MHz (which implies, that the SCI net runs at 50 MHz).

Figure 7 shows the intervals and the trigger events (signals).

The latency for the transaction (from the trigger event PCI write until the memory write) is 3.1 μs. The PCI9060 needs 400 ns to transform the PCI transaction into a PCI unit’s local bus operation, and vice versa, respectively. Packet generation and decoding lasts 550 ns, respectively. The transmission of the packet from the DPR to the LC via the B-Link takes a period of 275 ns. For the rest of the time (650 ns), the packet is buffered within the LC and on the way through the SCI ring.

The bridge is being tuned to run at a system clock of 25 MHz eventually. Thus, we expect performance to improve by about 40 %.
4 Summary and further Work

In this paper, we have presented an overview of the SMiLE project at LRR-TUM, and described in detail the PCI-SCI interface. A first prototype of the PCI-SCI bridge exists, for which we could present the first preliminary performance results. Concurrently, we are developing a device driver for Linux, as well as a network interface device driver for Windows NT.

The second version of the PCI-SCI bridge will integrate a communications monitoring hardware. Equipped with this monitoring hardware, the SMiLE multiprocessor will be an adequate platform for experimental research into distributed shared memory systems.

References


