ADVANCED COMMUNICATIONS IN CLUSTER SYSTEMS. CHALLENGES AND DECISIONS

Stanislav Simeonov

BOURGAS FREE UNIVERSITY - Department for Informatics, Engineering and Natural Science
BOURGAS – 8000, "A'leksandrowska" 101,BULGARIA
e-mail: stan@bfu.bg

Abstract

The new technologies in network computing gave basis for development of cluster technologies. The Virtual Interface Architecture (VIA) specification made it possible to design hardware infrastructures independent to the operation system and at the same time maintains high productivity network solutions at user level. On the other hand, the Scaleable Coherent Interface (SCI) provides means to achieve of extremely low latent times during network access. The combination of these technologies is the key to obtain real scalability in the development of cluster computer architectures with high productivity and at low cost. The aim of the paper is to present some solutions for the improvement of communication within a system.

1. MOTIVATION AND INTRODUCTION

The advanced networks of workstations are becoming the primary computing infrastructure for scientific and engineering from low end interactive computing to applications demanding sequential and parallel computing. **Symmetric Multi-Processor** (SMP) machines are a recent addition to the parallel computing market. They use workstation microprocessor technology but join small number of CPU's (typical 4) with certain level of memory hierarchy to achieve high–end uniprocessor workstation. In a typical simple structure, all processors share a global memory via common bus. As all processors have the same access capabilities to all I/O resources, it is called SMP machine. The cache coherence problem can easily by handled by a pus-snooping protocol. Another advantage of SMP is that all processes have equal access to the architecture. The main drawback of SMP is the restricted scalability due to the limited bandwidth of the common bus, which must be shared by all processors. A variety of parallel computer systems has been offered on the market for some time now. Each one of them has unique hardware and/or software architecture and some of them were announced to have productivity measured in teraflops.

A parallel computer system of high productivity can be realized by two entirely different ways:

- designing a special computer architecture for a certain task which is to be solved by the newest technological principles

- multiplication of standard computer elements communicating between each other

A simple strategy for implementing a scalable and high performance computing facility is to cluster SMP's via very high speed communication system, like **Scalable Coherent Interface** (SCI). The resulting configuration behaves much like a distributed memory multicomputer, except that each node actually has multiple CPU's sharing a common memory. The communication within SMP is carried out via shared memory and between SMP by message passing or shared memory provided by hardware[1].

Usually, each network interface is treated like a peripheral I/O device, and each workstation treats others as foreign agents. The result is long communication latency, low bandwidth, and poor bandwidth utilization. SCI (scalable coherent interface), which was developed by a number of high-performance-bus designers and system architects, is an efficient interface for the interconnection of SMP.

2. COMMUNICATION MECHANISM

SCI is a point-to-point datalink mechanism developed by working group P1596 inside IEEE Computer Science[2], designed to link a large number of processors (up to 64K). It provides high speed bus as a service to nodes in a distributed environment. SCI scales well and avoids bus limitations by using many point-to-point links, as well as hardware embedded packet protocol to provide coherent and non-coherent shared memory to the nodes. The node in SCI may be processor, memory, I/O units or complete workstations connected to SCI.

SCI is packet based. It uses unidirectional lines to configure efficient rings where many nodes can be connected. Forward progress is guaranteed in the rings, and there are buffer capacities to both send and receive packets at full speed in each node. The SCI was originally designed for substituting buses inside computers, and therefore has bus-like functions. Protocols for read, write and atomic operations are supported in hardware together with a cycle redundancy check in each packet, which will be automatically retrieved if error occurs. SCI has a protocol to handle cache coherence through a directory based scheme with double linked lists. The cache lines are defined to be 64 bytes. SCI has a number of fixed packet lengths, ranging from 16 bytes to 288 bytes.

The simplest SCI systems will likely consist of a single register-insertion ring. More complex system will couple many small ringlets or use active switches to form very low latency networks of arbitrary topology. Figure 1. shows the basic queuing structure of a SCI node. Input and output FIFO's buffer the slow, burst local traffic at each node from the high-speed communication on the link. By providing separate queues, a node can simultaneously transmit and receive packets, effectively doubling communication bandwidth at each node. Incoming packets they are routed through the bypass FIFO, which is given non-preemptive priority over a node's own transmission. Thus, the maximum length of the packets determines the size of this queue.

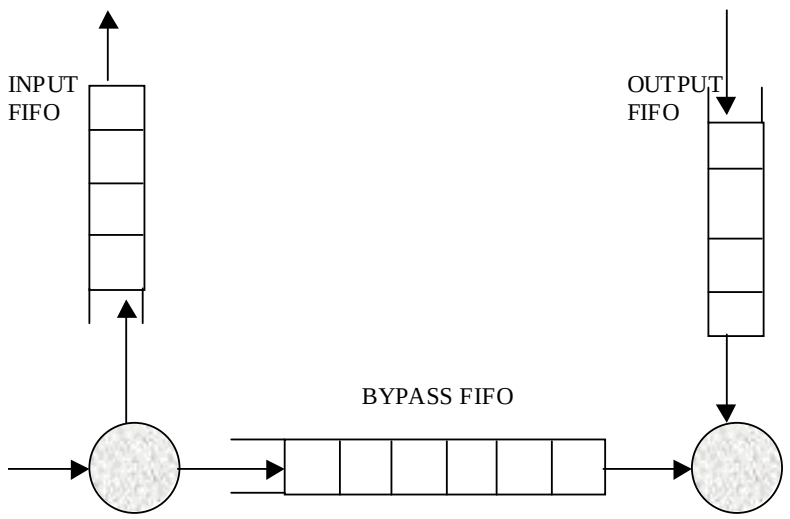

Figure 1 Basic structure of a SCI node

3. KEY CHARACTERISTICS OF PARALLEL SCALEABLE COMPUTERS

Scalability can be defined in different ways. It could be stated that the scalability of a parallel system is the ability to increase the performance of a cluster computer system, defined as a function of the number of its elements (personal computers for a cluster of PC's). In recent years the scalability of the parallel system is of special consideration in the development of modern architectures. The increase of the number of resources increases system overhead necessary for interprocessor communication and synchronization. That is why the system performance is not proportional to the number of processors and amount of memory involved.

The parallel work of the system could be defined as a combination of a parallel machine and the corresponding algorithm and can be formalized as **C=(A,M)**, where

A is the parallel algorithm implemented in a parallel computer system M. Two basic parameters affect the performance of a parallel system - decrease of execution time and the size of the problem to be solved. The analysis of system performance, and particularly the analysis of system scalability, is based on the time for parallel execution and the size of the problem. The latter can be defined as all the basic operations (e.g. floating point operations during calculations) executed by the algorithm **A** on the machine **M**. It is determined by the amount of effective work of the parallel system (number of operations) done to solve the problem. This is the amount of work by parallel computing **C** and will be denoted as **W**. Using **T(W,P)** to denote the time for execution of the algorithm **A** on **P** computers working within a cluster, the speed **S** by parallel computing **C** can be defined as the ratio:

$$S(W,P) = \frac{W}{T(W,P)} \tag{1}$$

Execution time can be divided into two parts: **T0** - time for real calculation and **T'** - overhead time.

To predict the scalability of a parallel system, $\bar{S}$ should be introduced as:

$$\bar{S} = \frac{W}{P \bullet (T_0(W,P) + T'(W,P))} \tag{2}$$

The increase of the number of computer systems within a cluster at constant amount or work, does not increase the speed proportionally because of the increase of the overhead. To keep the scalability of the combination parallel machine - algorithm by the increase of the number of machines involved from **P** to **P'**, the amount of the work should be increased from **W** to **W'**. Besides, to retain the same time characteristics at acceptable accuracy, the following ratio should be obeyed:

$$\alpha(P,P') = \frac{P' \bullet W}{W' \bullet P} \tag{3}$$

This ratio gives how the amount of work should be increased by the change of the parallel system resources in order to keep speed scalability. Any optimizations aimed to retain the scalability of the parallel cluster systems inevitably require new approaches in the realization of the intercomputer communication.

4. VIRTUALIZING THE HARDWARE

When multiple processes want to access the hardware directly, this must be handled very careful. The **Network Interface Controller** (NIC) has to keep the context for every supported **Virtual Interface** (VI). The NIC must be able to assign control access to each of its hardware registers through its unique context. Figure 2 shows how this requirement is realized with the **Virtual Interface Architecture** (VIA)[3][4].

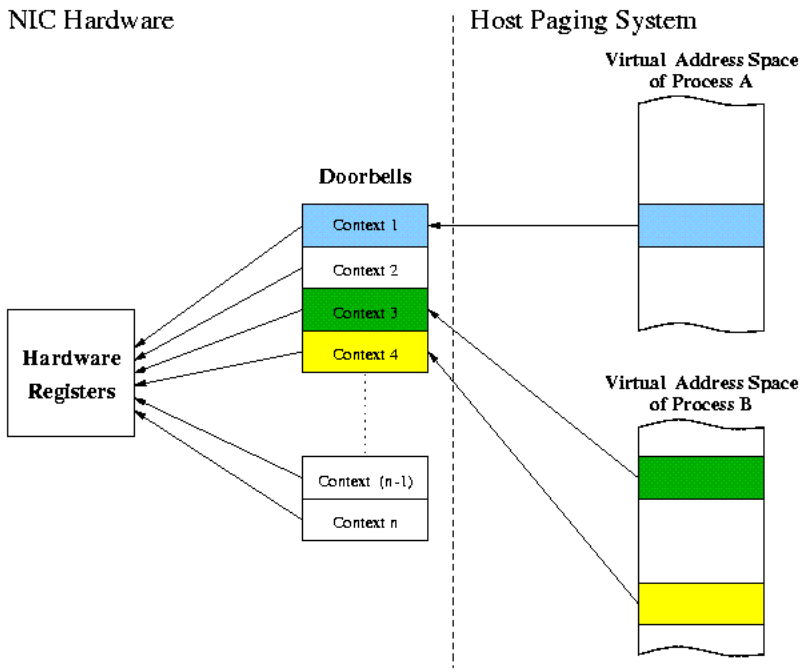

Figure 2: Memory Mapped Control Register via “Doorbells”

The NIC supports a number of VI Contexts (n in this example). One context belongs to one VI endpoint. An access to hardware register must pass a so called Doorbell. The doorbells are page aligned memory windows, so that a process can get these mapped into its virtual memory space. As shown in the figure, one process may also get multiple doorbells corresponding to multiple VIs. The multiple processes may also share one doorbell. Finally, this depends on the application and on the management software. A VI context contains multiple stuff. From the point of view of the hardware, the most important thing is the Protection Tag. Typically, unique protection tag is assigned to each doorbell. The protection tag can’t be changed by a task, but by the system kernel. The protection tag is valid only for the local node. Protection tags are not sent over the network. The VIA NIC make use of VI context

also for incoming transaction. So it is possible to assign the right protection tag for incoming VIA send/receive transactions and NIC is able to qualify the memory access.

Likewise to a process, the VIA NIC itself has a virtual address space. The space is divided into pages (typically 4kB in size). A page table is used to translate NIC virtual addresses into physical addresses. Beside the physical address, a page table entry contains also a protection tag and some other access right bits. By checking the protection tag before a memory access can take place, the NIC is able to detect whether the process initiating this transfer is allowed to access this page or not. The protection tag assigned to this page equals the protection tag assigned to the doorbell, where the transfer was initiated. Then the initiating process is allowed to perform such a operation.

Before a process can invoke a transfer, it has parts of its memory within the VIA NIC. The VIA specifies a so called **Memory Handle**. The memory handle is used in a kind of translation of the process virtual memory to get the corresponding NIC virtual address.

The VIA NIC makes no difference between memory pages or those containing descriptors. In both cases the same translation and protection model is used. This means also that pages containing descriptors have to be registered within the VIA NIC before they can be used. On memory registration, no difference is made between data and descriptor pages.

Figure 3 shows an example where a process has registered some pages within the VIA NIC[5]. The blue pages are used for data and the red ones for descriptors. The VIA NIC doesn’t see any difference between the red and blue pages. They all are equal for the NIC.

5. CONCLUSION

To set a good starting point for development of communication modules for cluster systems, a profound analysis of the scalability and the factors affecting it should be carried out. A suitable decision of the communication problems in this case is the use of SCI-PCI bridges. They provide low enough latency in the interprocess communication between the applications. The ideas presented in this paper aim at the development of a bridge combining the SCI with VIA. At the same time, a concept for

future merging of SCI and VIA technologies into a single module was presented, which enables flexible memory management. This results in hardware with substantially new characteristics.

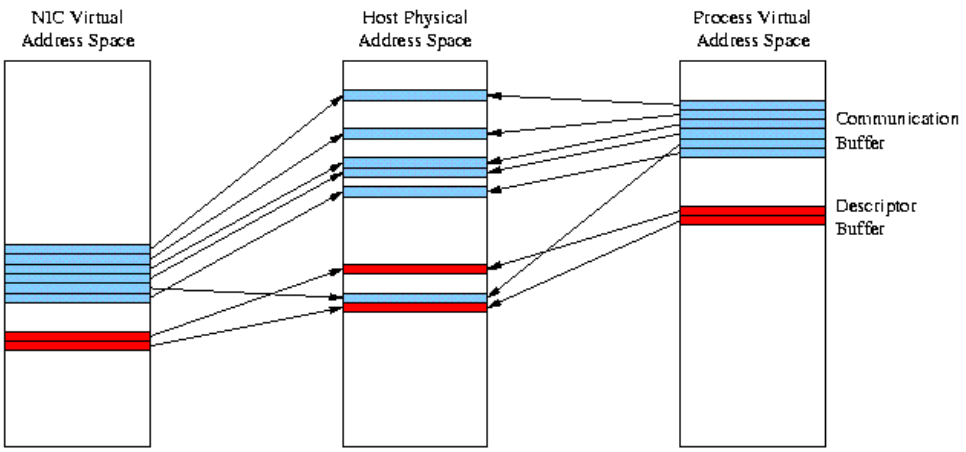

Figure 3: Mapping of Data and Descriptor Buffers

REFERENCES

[1] MESSAGE PASSING INTERFACE FORUM: *The MPI message--passing interface standard Rev. 2.0*, May 1998.

[2] IEEE: *Standard for Scalable Coherent Interface (SCI)* IEEE Std. 1596-1992.

[3] INTEL, COMPAQ AND MICROSOFT: *Virtual Interface Architecture Specification V1.0*. See also Virtual Interface Architecture.

[4] BUONADONNA PH., A. GEWEKE: *An Implementation and Analysis of the Virtual Interface Architecture*. University of California at Berkeley, Computer Science Department, Berkeley, 1998.

[5] TRAMS, M. Design einer systemfreundlichen PCI-SCI Bridge mit optimiertem User-Interface, TU Chemnitz, 1998.