

# ABSTRACT

## Design of Adaptable Computing Cluster\*

Ron Sass      Keith Underwood<sup>†</sup>      Walter Ligon

Parallel Architecture Research Lab  
Holcombe Department of Electrical and Computer Engineering  
Clemson University  
105 Riggs Hall  
Clemson, SC 29634-0915  
E-mail: {rsass, keithu, walt}@parl.clemson.edu

### 1. Introduction

Military and Aerospace applications often place high computational demands on computing systems. Cluster computers based on COTS parts (Beowulf-class machines) offer an outstanding price-performance ratio and work exceptionally well for some applications. However, some applications do not map well to the distributed memory of a cluster computer and even though emerging high-performance commodity network technology provides significantly higher bandwidth, cluster computers fail to meet the desired performance for some applications. Another technology — reconfigurable computing or FPGA-based custom computing machines — also accelerates certain applications but is limited in its general applicability. Thus the question arises, “can combining these technologies offer a significant performance advantage?” We believe that the two are complimentary and, in fact, do enable each other such that they exceed the simple sum of their parts. In this paper, we describe the Adaptable Computing Cluster (ACC) architecture and its key component the Intelligent Network Interface Card (INIC). An experimental system has been built to test the ideas presented. It is a 16-node cluster connected with a Gigabit Ethernet switch. On each node, a reconfigurable computing card with a Gigabit Ethernet mezzanine card is used to emulate our INIC. We discuss the implementation of the prototype and how it differs from our ideal design. A short summary of the some initial performance results are presented to validate our proposition that the ACC architecture provides a cost-effective high-performance COTS system.

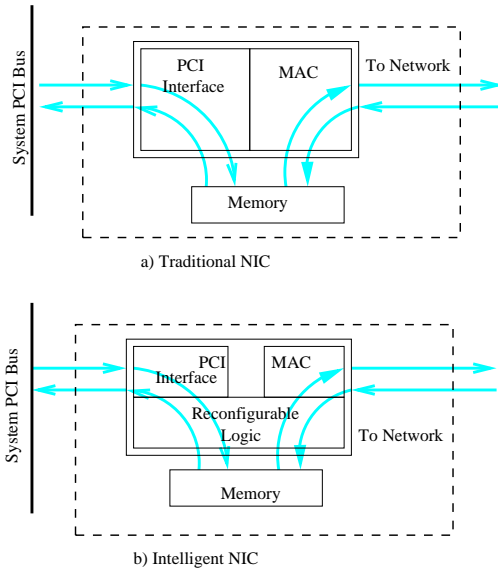
\*This work was supported in part by the National Science Foundation under NSF Grant EIA-9985986

<sup>†</sup>Mr. Underwood is supported by a NASA GSRP Fellowship under grant number NGT5-85

### 2. Adaptable Computing Cluster

Using programmable logic to speed up high-performance computing is becoming more commonplace and indeed others have investigated the use of FPGAs in cluster systems. For example, VPI’s Tower of Power project [2] augments each node in a cluster with an off-the-shelf reconfigurable computing card. Dandalis [1] uses FPGAs in the NIC to implement IPsec. Sepia [3] uses Smart (FPGA-based) NICs to implement a cluster computer for image processing. Others have used FPGAs in the switching fabric [5]. Our objective was to build an integrated architecture that allows each of the two technologies to mask the deficiencies of the other. Specifically, we wanted to put the FPGAs in the flow of data so as not to add load to each node’s I/O bus. We wanted to take full advantage of the 1000 Mb/s link speed of the Gigabit interface (since the COTS PCI systems cannot). And because others have shown that many latency-sensitive applications perform badly on Beowulf clusters, we wanted to improve the message latency. Based on these objectives, we believe that the FPGA resources belong in the network interface side of the system I/O bus and propose the architecture illustrated in [Figure 1](#).

A cluster built with this Intelligent NIC at each node, provides several ways of improving performance. Fundamentally, we foresee several modes of operations. The cluster can operate on its own with a default configuration of the FPGAs. In this way, some of the message-passing protocol can be implemented on the card which improves acknowledgement latency and offloads some work from the processor (similar to [1]). The FPGAs could be dedicated solely to improving the computation performance and not interact directly with network. This mode simply improves the computational power of a single node (as in the [2]). A final

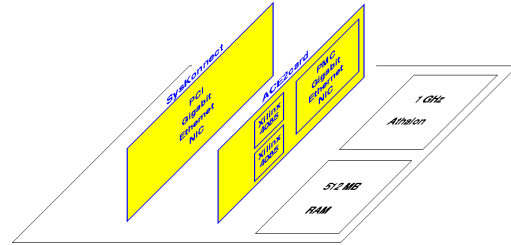


**Figure 1.** proposed enhancement to a network interface

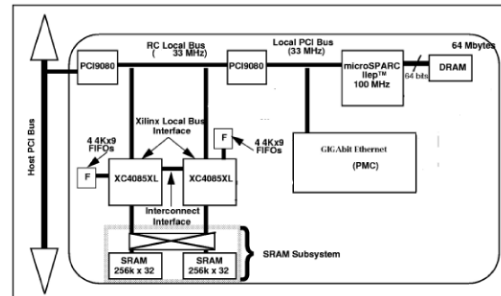
mode of operation, which we believe makes the architecture especially beneficial, has the FPGA resources involved in computation and communication. When the cluster can be used in this mode, an application transmits data to the network interface where the FPGAs can further operate on the data and then inject messages into the network. This simultaneously addresses several deficiencies. There is no overhead in using the reconfigurable computing (since the data already has to travel through the NIC). The processing does not add demand to the node's I/O bus. Since the bandwidth between the FPGAs and the network is not limited to a COTS bus, the FPGAs have the opportunity to fully exploit the bandwidth and latency advantages of modern network technology. Furthermore, this beneficial mode distinguishes our work from many of the other cluster/FPGA projects in progress.

### 3. Clemson Prototype

To test the ideas behind an Adaptable Computing Cluster, we have developed a prototype cluster. The experimental architecture is a 16-node Beowulf running the Scyld Linux distribution. Each node (as illustrated [Figure 2](#)) is a 32-bit PCI motherboard with a 1GHz Athlon and 512 MB of RAM. On the PCI system bus is a SysKconnect PCI Gigabit Ethernet NIC, a Fast Ethernet NIC, and an ACEII card. The ACEII has an onboard PCI bus attaching a  $\mu$ SPARC processor and PCI Mezzanine (PMC) slot to the FPGAs.



**Figure 2.** one ACC node



**Figure 3.** Block diagram of an ACEII card

The PMC slot is populated with an Alta Technologies PMC Gigabit NIC based on the Packet Engines Hamachi chipset. Note that the two Gigabit Ethernet cards use identical components — their chief difference is only the form factor.

The ACEII is a second-generation reconfigurable computing board from TSI TelSys designed to do ground station satellite communications processing. A simplified block diagram of the board is shown in [Figure 3](#). Specifically, it has two Xilinx XC4085XLA-07 FPGAs, 2 MB of SRAM, two FIFOs, and a  $\mu$ SPARC embedded processor. The node's PCI bus is bridged to the i960 bus where the FPGAs are located. Another bridge connects the i960 bus to a local PCI bus that the  $\mu$ SPARC and PCI Mezzanine use. Although not ideal, the board is a sufficient experimental testbed for prototyping an Intelligent NIC. Furthermore, because it was an off-the-shelf design, we were able to contain costs. The PMC connector offered a standard interface for which a variety of network interfaces are commercially available.

Current architectural deficiencies prevent the card from achieving its full potential as an Intelligent NIC; nonetheless, it suffices to demonstrate the concepts we are addressing.

### 4. Performance

The price-performance advantages of Beowulf clusters is well understood. However, for some classes of applica-

tions the commodity networks are insufficient. For example, applications that frequently calculate the 2-D Fast Fourier Transform (FFT) suffer because of the communication-intensive transposes required by 2-D FFT. Similarly, integer sort is fast on distributed memory machines but by doing some of the sorting in the network interface — as the data is being transferred — we can further improve the performance. The current implementation of the experimental system prohibits us from running the full tests (four of the ACE2 boards have not been delivered and the complete communication protocol, with flow control and retransmission, has not been implemented in the VHDL). However, the VHDL for the application components of the FFT and integer sort tests are operational and from these results we can compute application times and extrapolate to sixteen nodes. The FFT application and the full explanation of its performance has been published in [4] and the integer sort work is still partially under development.

#### 4.1. FFT

Figure 4 compares the scalability of FFTW on Fast Ethernet, Gigabit Ethernet, and our Intelligent NIC.<sup>1</sup> The data associated with Fast Ethernet in Figure 4 indicates that the bandwidth of Fast Ethernet severely limits scalability for all of the problem sizes of interest. To even improve upon the performance of a single processor, a minimum of eight nodes is required! Furthermore, with 14 nodes, the Fast Ethernet cluster barely doubles the performance of a single processor. Results using Gigabit Ethernet are also shown in Figure 4. Gigabit Ethernet gives a speedup of two with 8 processors and is able to approach a speedup of four in some cases. This is better, but would hardly be considered scalable. Finally, the data associated with the INIC performance indicates that the INIC offers both a significant speedup over and better scalability than the standard Gigabit Ethernet solution, although both use the same network hardware.

#### 4.2. Integer Sort

Figure 5(b) compares the speedups of an INIC and a Gigabit Ethernet implementation of integer sorting. The superlinear speedups achieved by the INIC implementation is attributable to the elimination of the time for bucket sorting the data (over 5 seconds in the serial implementation). The INIC exhibits much better speedups than the Gigabit Ethernet solution by eliminating bucket sorting time and imple-

<sup>1</sup>Intelligent NIC speedups are conservative estimates. These estimates combine measurements of actual times to transfer data to and from the card while performing the local transpose and final permutation based on the design reported in [4] and preliminary bandwidth measurements between two cards.

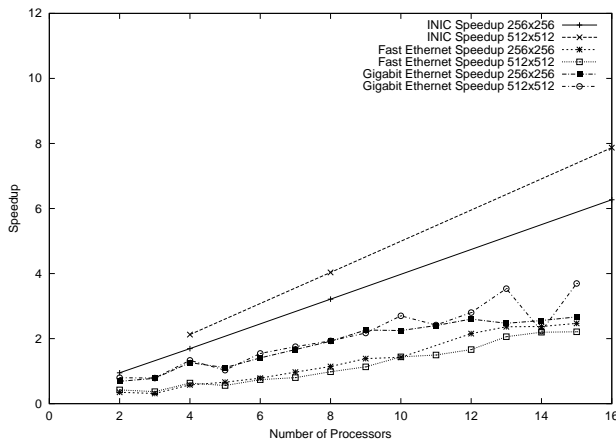


Figure 4. The graph shows the parallel speedup for a 2D-FFT for matrix sizes of  $256 \times 256$  and  $512 \times 512$  on three technologies: Fast Ethernet, Gigabit Ethernet, and an Intelligent NIC

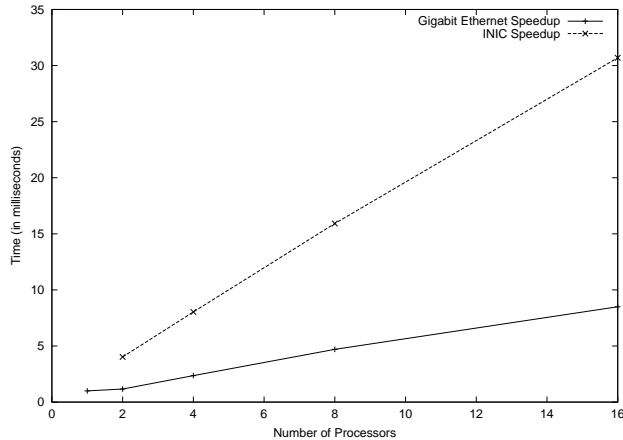
menting a better protocol for the relatively small data transfers caused by larger numbers of processors.

Figure 5(b) compares the speedup<sup>2</sup> of our prototype INIC and Gigabit Ethernet implementations. Our prototype INIC does not achieve the full potential of the INIC limited both by the bus bandwidth on the card and the need to perform a second stage bucket sort on the receiving host. It does, however, have significant speedup.

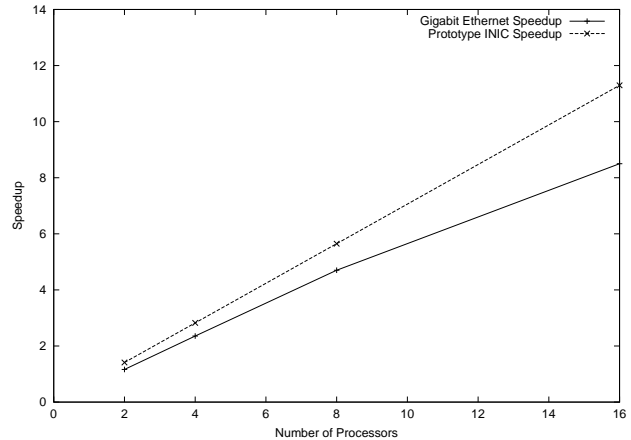
## 5. Cost Assumptions and Conclusion

A important advantage to Beowulf clusters is that they use commodity off-the-shelf components. With multiple vendors in competition and larger production runs, Beowulf clusters are able to contain system cost. Our experimental architecture uses off-the-shelf components, but the reconfigurable computing card is  $10\times$  the cost of a standard NIC (even before adding Gigabit Ethernet). Furthermore, some may argue that Gigabit Ethernet is not yet commodity-priced. While we cannot predict the future, we have reason to believe that these limitations will become insignificant in the future. With the advent of Gigabit over copper standard, it seems likely that it will be widely adopted (if for no other reason, it uses existing infrastructure). Our architecture uses unmodified Gigabit Ethernet switches and so that aspect will follow the market. The INICs we propose could be built less expensively than our prototype. For example, several Gigabit Ethernet cards come with embedded

<sup>2</sup>Speedups for the INIC are estimates based on the equations derived earlier.



(a)



(b)

**Figure 5.** (a) a comparison of parallel speedups for INIC and Gigabit Ethernet implementations of integer sorting; (b) a comparison of parallel speedups for a prototype INIC and Gigabit Ethernet implementations of integer sorting

processors on them. According to Xilinx, a future version of the Virtex II will have the option of an embedded processor. Based on predicted prices from Xilinx, this chip could be substituted for the embedded processor on the NIC. While this would allow the NICs to be used in our ACC architecture, it would be useful in a broader market because of its ability to adapt to evolving protocol standards (such as IPsec in [1]). Under such circumstances it is believable there may be multiple vendors with large production runs.

Accepting these assumptions, the proposed architecture becomes a particularly potent computing resource for Military and Aerospace applications. The cost of ground station telemetry processing and image processing for large amounts of data is reasonable on this type of architecture and the preliminary performance tests suggest that the architecture can be scaled up to meet the computational demand.

## References

- [1] A. Dandalis, V. K. Prasanna, and J. D. P. Rolim. An adaptive cryptographic engine for IPsec architectures. In *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 132–141, Napa Valley, CA, April 2000.
- [2] M. Jones, L. Scharf, J. Scott, C. Twaddle, M. Yaconis, K. Yao, and P. Athanas. Implementing an API for distributed adaptive computing systems. In *Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines*, pages 222–230, Napa Valley, CA, April 1999.
- [3] L. Moll, A. Heirich, and M. Shand. Sepia: scalable 3d compositing using PCI Pammette. In *Proceedings of the IEEE*

*Symposium on Field-Programmable Custom Computing Machines*, pages 146–155, Napa Valley, CA, April 1999.

- [4] K. Underwood, R. Sass, and W. Ligon. Acceleration of a 2d-fft on an adaptable computing cluster. In *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines*, April 2001.
- [5] C.-C. Yeh, C.-H. Wu, and J.-Y. Juang. Design and implementation of a multicomputer interconnection network using FPGAs. In *Proceedings of the IEEE Symposium on FPGAs for Custom Computing Machines*, pages 30–39, Napa Valley, CA, April 1995.