Enhancing SCI's Fairness Protocol for Increased Throughput

Dan Picker, Ronald D. Fellman, and Paul M. Chau

Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA. 92093-0407

Abstract

The Scalable Coherent Interface (SCI) is a newly established IEEE standard which provides bus-like services to a network of nodes connected by very fast, unidirectional links. SCI provides an efficient flow control protocol that fairly allocates bandwidth to all nodes on the network. When global fairness is not required, it is desirable to relax its enforcement, in exchange for increased network utilization. This paper presents an extension to SCI's fairness protocol that increases attainable throughput by enforcing fairness only among conflicting nodes. We present extensive simulation results and compare them to the standard SCI protocol and to our desired performance, which we derive in the paper. The comparisons show that, on average, the enhancements substantially increase the performance of the original protocol.

1 Introduction

The first VLSI implementations of the Scalable Coherent Interface (SCI) bus standard (ANSI/IEEE 1596) have begun to hit the market less than a year and a half since its approval. Through unidirectional point-to-point communications links, SCI allows high-bandwidth terminated transmission lines or fiber optic technology to construct high-performance multiprocessor systems and distributed computing networks. Dolphin SCI Technology's initial implementation has realized SCI's hardware support for distributed global shared addressing and its potential for high-throughput, low latency multiprocessor communications with a throughput rate of over 500 Megabytes/sec and a node latency of less than 45 nsec. LSI Logic has announced a CMOS SCI implementation that boasts a throughput of over 125 Megabytes/sec. National Semiconductor has announced its Quickring CMOS VLSI interfaces as a low-cost alternative that leverages off of SCI's physical layer to support 180 Megabyte/sec transfer rates with a 40 nsec node latency. These and other SCI implementations promise to jump-start the emergent distributed computing field and greatly enhance the usefulness of massively parallel computers that currently suffer from large communication latencies, limited throughput, and a lack of support for shared memory programming.

Although SCI specifies an efficient set of protocols that guarantee fair access and deadlock-free operation, it was developed primarily for general purpose computing, where fairness is normally desirable. This paper describes an extension to the SCI protocols that significantly enhances performance for distributed computing applications that exploit some degree of pipelining or exhibit regularity in their patterns of data communications. Many numeric intensive applications, such as image and signal processing, share these characteristics. Furthermore, we developed these extensions in close collaboration with key members of the IEEE P1596 SCI Working Group in order to insure compatibility with the existing standard.

1.1 Overview of SCI

The simplest SCI systems will likely consist of a single register-insertion ring. More complex systems will couple many small *ringlets* or use active switches to form very low latency networks of arbitrary topology. Although the standard specifies 2-byte wide data paths, the use of unidirectional links encourages serial fiber-optic implementations. We present here only a brief summary of the operation of SCI and refer the reader to [1,2] for more detailed and comprehensive treatments.

Figure 1 shows the basic queueing structure of an SCI node. Input and output FIFOs buffer the slow, bursty local traffic at each node from the high-speed communications on the links. By providing separate queues, a node can simultaneously transmit and receive packets, effectively doubling communications bandwidth at each node. Incoming packets that are destined for a different node are routed through the bypass FIFO, which is given non pre-emptive priority over a node's own transmissions. The

This work was supported by grants from the Naval Research Labs, Apple Computer, Inc., the MICRO Program of the State of California, and the UCSD ICAS Center (NSF Grant 89-5821).

0-8186-3670-X/93 $03.00 © 1993 IEEE

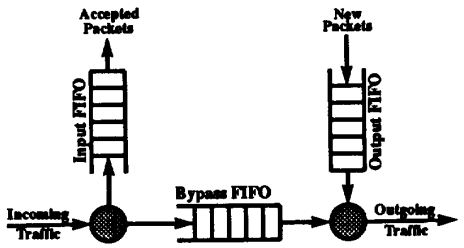

Figure 1: *Basic Structure of an SCI Node*

maximum length of a packet thus determines the size of this queue.

1.2 SCI Transactions and Flow Control

A basic transaction on an SCI ringlet consists of a *source* node transmitting a *send* packet to a specified *target* node on the same ringlet. Upon receipt of the packet, the target node returns an *echo* packet to the source node, indicating whether or not the packet was accepted (i.e. whether or not sufficient input queue space was available). If rejected, the packet is re-transmitted by the source node.

Because the SCI protocols are synchronous, special "spacer words" or *idle symbols* are sent in the absence of packets, and at least one is always sent between consecutive packets. Although SCI uses these idle symbols for a variety of purposes, they are of key importance to the flow control protocol.

Since the bypass FIFO is given priority over the transmission queue, a node cannot transmit a packet unless its bypass fifo is empty. In the absence of flow control, upstream nodes can potentially overwhelm a downstream node so that it cannot empty its bypass FIFO and, hence, cannot transmit packets waiting in its transmission queue.

The design of the SCI flow control protocol assures fair access to all nodes on a local ringlet. It accomplishes this by allowing a node to empty its bypass FIFO by gradually preventing all other nodes on the ringlet from transmitting. However, poor network utilization often results since this affects *all* nodes, regardless of whether or not they were a direct cause of conflict to the overwhelmed node.

To illustrate the problem, consider a 4-node ring in which nodes 1,2, and 3 are continually trying to send transmissions to node 4, while node 4 always sends packets to node 2. For the sake of clarity in this example, we shall assume that all transmissions are *echo-less*, and ignore the small amount of bandwidth used up by the idle symbols which occur between packets. In the absence of flow control, a steady stream of transmissions from node 4 will prevent node 1 from ever transmitting. The lack of transmissions from node 1 allows node 2 to transmit indefinitely, preventing node 3 from ever transmitting a packet of its own. To remedy this situation, SCI's fairness protocol restricts the throughput of each node to a maximum of one third of the bandwidth offered by a link. Although nodes 1, 2, and 3 all share $link_{3,4}$ and should indeed each receive one third of its bandwidth, node 4 only shares bandwidth with node 1. Since the throughput of node 1 is already constrained, node 4 could use two thirds of the bandwidth of $link_{1,2}$ without further degrading the throughput of node 1. Hence, it would be advantageous to *relax* the fairness constraint so that node 4 can utilize all the *available* bandwidth of the links that its transmissions traverse.

1.3 Relaxed Fairness

In the example above, a fairness protocol produced sub-optimal network utilization because it reduced the throughput of a node beyond the point at which its transmissions limited those of other nodes. More generally, we define the optimal throughput of a node under a *relaxed fairness* discipline to be the maximum throughput for which it does not *unfairly* inhibit the *attempted* throughput of other nodes. Hence, if a node uses less of a link's bandwidth than is fairly allocated to it, other nodes sharing that link may increase their bandwidths accordingly.

1.4 Organization of the Paper

In this paper we extend the SCI flow control protocol to accommodate relaxed fairness. This enhancement increases attainable throughput by allowing an overwhelmed node to cause only conflicting nodes (i.e. those currently sending packets through its bypass FIFO) to refrain from transmission. Nodes sending only non-conflicting transmissions remain unaffected.

We begin in Section 2 by presenting the details of the original protocol, and then describe our proposed modifications in Section 3.

To evaluate the effectiveness of the protocol, we present in Section 4 equations for the optimal throughputs of each node on a ringlet governed by relaxed fairness.

In Section 5 we compare the performance of our flow-control protocol to that of SCI's protocol, and to the optimal performance for various ring sizes and traffic scenarios. Finally, in Section 6, we summarize the main results of the paper.

2 The Standard SCI Flow Control Protocol

A description and analysis of the original SCI flow control mechanism was given in [5]. However, the protocol has recently undergone some modifications and, hence, we present a brief description of the updated protocol.

Although SCI's flow control scheme supports a 4-level priority mechanism, we shall, without loss of generality,

describe its operation within the context of a single priority. The same assumption was made in [5]. We note that a similar priority scheme can be applied to the modified protocol presented in this paper.

Figure 2 shows a simplified state transition diagram of the transmitter. The flow control protocol determines the value of the variable *blocked*. This diagram will be referenced throughout the remainder of this section.

Earlier, we explained that idle symbols traverse the ring between and in the absence of packets. The circulation of *go bits* within these idle symbols distributes flow control information. When a node has no packet to transmit, it simply forwards all passing packets and idle symbols downstream (state A). Fundamentally, a node may transmit a packet only if its bypass FIFO is empty *and* it has just emitted an idle containing a go bit (i.e. with go bit field set to 1). This is represented by a transition from state A to state B of the transmitter state diagram of Figure 2. A node that wishes to transmit a packet but cannot, either because its bypass FIFO is not empty (state C) or it has not received a go bit (state D), is said to be *blocked*. While waiting for its bypass FIFO to empty, a blocked node records the arrival of any go bits it receives, but does not forward them downstream. Multiple go bits are, in effect, merged into a single go bit. By removing go bits from passing idles, the node gradually depletes the ringlet of go bits and, hence, forces all other nodes on the ringlet to refrain from transmission. Eventually, the blocked node's bypass FIFO will empty. If at least one go bit has arrived, the node releases this *saved* go bit in the next idle symbol and follows that symbol with its own transmission.

While transmitting a packet (state B), a node again merges arriving go bits and releases the saved go bit in the idle that follows its transmission (as it transitions to state A). If its bypass FIFO has filled up at all, the node enters state C and proceeds as described previously.

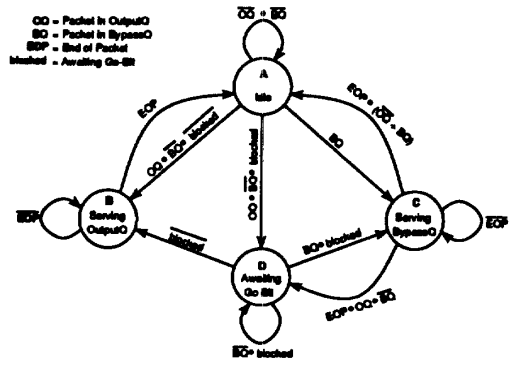

Figure 2: *State Transition Diagram of the Transmitter*

While reading a packet into its receive queue, a node copies the go bit field of the last emitted idle into subsequent idle symbols. This is called go bit extension, and it continues until the arrival of the next packet or idle symbol. Go bit extension can help to quickly replenish the supply of go bits on a lightly loaded ringlet.

Although not shown in the state diagram of Figure 2, an idle symbol is always transmitted when transitioning to states A or D, whereas a data word is transmitted when transitioning to states B or C. This ensures that every data packet is followed by an idle symbol. Note that a node which becomes unblocked while in state D *always* transitions to state B. This reflects the fact that ties are given to the output queue.

3 Extending the Protocol for *Relaxed Fairness*

In this section we outline an extension to the standard SCI flow control scheme which achieves higher overall throughput and, hence, smaller message delay. Because the protocol is rather complex, we present here only a summary of its main features and operation, and refer the interested reader to [4] for a more precise specification. The transmitter state transition diagram of Figure 2 applies to this protocol as well.

3.1 The Modified Scheme

As mentioned in Section 1, when *global* fairness is not required, SCI's flow control protocol often makes inefficient use of bandwidth, due in large part to its unselective blocking of nodes. Our solution to this problem is to assign packets to transmission groups, and enforce fairness between groups containing conflicting transmissions. The source node acquires the group information from a register within the node interface which either remains constant, or changes between packets according to software control. It encodes this group in the packet's header. A blocked node records the groups of passing send packets in a special *Block Register*. The node records the group of the packet at the head of its transmit queue within this register as well.

Idle symbols contain a distinct go bit *for each group*. Normally, a node may transmit a packet only if it has received a go bit corresponding to a group contained within its Block Register. Moreover, while emptying its bypass queue, a node only removes go bits in passing idle symbols that correspond to groups contained within its Block Register. In this way, only transmissions belonging to conflicting groups will be forced to refrain.

The Block Register allows a node to learn and adapt to the current loading and traffic scenario. It is cleared only when a node finishes transmitting a packet and finds the bypass queue *empty*. This usually indicates that a busy cycle has completed. When the next busy cycle begins, the

initialized Block Register will allow the node to adapt to a new, *possibly different*, traffic scenario.

To transmit its own packet, it is sufficient for a node to receive a go bit corresponding to any group contained in its Block Register. It does not, necessarily, have to wait for a go bit corresponding to its own transmission. Otherwise, deadlock could occur if a number of blocked nodes withheld each other's go bits.

Note that a node blocks go bits corresponding to its *own* packet awaiting transmission, even if that group would not have otherwise been represented in its Block Register. This balances the otherwise unilateral nature of most conflicts. To ensure fairness, conflicting nodes should, ideally, block *each other*. Since a unidirectional ringlet contains no feedback mechanism, we force a downstream node to, in effect, block itself on behalf of the nodes whose go bits it is consuming.

In essence, our protocol combines conflicting transmissions into a single functional group, and uses SCI's protocol to enforce fairness within this super-group. To promote independence between disjoint sets of super-groups, nodes must pass along, unchanged, go bits corresponding to groups not represented in their Block Registers.

3.2 Implementation Considerations

The protocol just described presents a functionally straight-forward extension to the standard SCI protocol. Indeed, it was designed to easily fit into the framework of the existing standard with minimal added cost. To maintain a high level of compatibility with SCI, the number of groups supported is an important design parameter that must be chosen with consideration of a number of constraints.

The greatest such constraint is available space within an idle symbol. Each group must be represented by a unique go bit field in each idle symbol. SCI currently uses all 16 bits of an idle symbol, of which only 2 are used as go bits (representing high and low priority groups). With minimal impact, several of the remaining bits could be used as additional go bits.

Obviously, an ideal implementation offers as many groups as source-destination (s-d) pairs. This completely eliminates blocking of non-conflicting nodes. Most traffic patterns, however, require far fewer groups, since only a subset of transmissions conflict heavily with each other.

In some cases, such as digital signal processing and other numeric-intensive applications, the load is known a-priori. Then strategically placing processors along the ringlet so as to decrease the average distance data must travel will minimize communications delay. Furthermore, optimizing processor placement will reduce the number of groups that efficient communications would require. An automated group assignment algorithm, based upon a traffic probability matrix is given in [3], and may be used to optimize the performance of the protocol.

Finally, we note that the first word of a packet's header should contain its group identifier. However, SCI reserves the entire first word of a packet for the target address, allowing 65,536 nodes. Since this is far more than would be practical to put on a single ring, several of these bits could be moved elsewhere within the packet header to be extracted and interpreted, off-ring, by a router within a gateway or bridge node. Even if 4 bits were needed to encode the group, the system could still support 4096 nodes per ring.

4 Optimal Throughput Equations

In this section we present an iteration for the optimal throughput of each node on a ring governed by a relaxed fairness discipline, as defined in Section 1.3. This quantity is denoted p_i^{opt}, or p_i^{optE} when echos are used. The results of this section are used in Section 5 to evaluate both the absolute performance of our flow control protocol, and its performance in relation to SCI's scheme. They are also used by the grouping algorithm given in [3]. We include this section for readers who desire a formal specification of the relaxed fairness discipline upon which the results of Section 5 are based. We first present an overview of the iteration, and follow that with the actual equations.

4.1 Overview

The procedure actually consists of two nested iterations. At the start of the first *major* iteration, the available bandwidth, $U_k^{0,0}$, of each link, k, is distributed among all nodes whose transmissions traverse it. Although we initially attempt to apportion the total link bandwidth equally, some nodes may not need this much bandwidth. The residual bandwidth, $U_k^{0,1}$, is equally distributed to nodes that have not yet been granted their desired throughputs across link k. This continues until either all the bandwidth of link k has been assigned, or until each node has been granted the amount of link k bandwidth it desires. After performing this procedure for each link, the resulting optimal throughput allotments are denoted $p_{ik}^{0,opt}$.

The actual throughput of a node i, $p_i^{0,min}$, is then chosen such that its throughput across any particular link k does not exceed $p_{ik}^{0,opt}$. Using the throughputs, $p_i^{0,min}$, the total link utilizations are re-computed. The throughputs, $p_i^{0,min}$, are final for nodes that use links which are maximally allocated, and are hence denoted p_i^{opt}. The throughputs of the remaining nodes across each link are reset to their initial values, $A_{ik}^{0,0}$, and the above steps are repeated. This process continues until p_i^{opt} is

obtained for every node i. Table 1 summarizes the main quantities used in the iteration.

4.2 Optimal Throughputs Without Echo Packets

$$t_{ik} = \begin{cases} \sum_{j \in S} r_{ij} & i \neq \langle k+1 \rangle_N \\ 0 & i = \langle k+1 \rangle_N \end{cases},$$

where $S = \{1,...,i\text{-}1,...,k\text{+}1,...,N\}$, N is the number of nodes, and $\langle\ \rangle_N$ denotes modulo-N.

$$A_{ik}^{m,0} = \begin{cases} t_{ik} p_i^{m-1,min} & \min_{l:\, t_{il}>0} \left[U_l^{m-1,0} - \sum_j t_{jl} p_j^{m-1,min} \right] = 0 \\ & \text{and} \quad (m>0) \\ t_{ik} p_i^{m-1,min} & otherwise \end{cases}$$

where λ_i is the attempted generation rate (in *packets/clock cycle*) of node i, and L_i^s is the mean length of send packets originating at node i.

$$A_{ik}^{m,n}\big|_{n>0} = A_{ik}^{m,n-1} - \Delta_{ik}^{m,n} .$$

$$U_k^{m,n} = \begin{cases} \min\{1, \sum_i A_{ik}^{m,0}\} & n = 0 \\ U_k^{m,n-1} - \sum_i \Delta_{ik}^{m,n} & n > 0 \end{cases}$$

Quantity	Description (TP = throughput, BW = bandwidth)
r_{ij}	Pr{packet sent by node i is destined for node j}
t_{ik}	Pr{packet sent by node i traverses link k}
$A_{ik}^{m,n}$	Residual node i TP assignable to link k after iteration (m,n)
$U_k^{m,n}$	Link k BW remaining for allocation after iteration (m,n)
$p_{ik}^{m,n}$	Link k BW allocated to node i after iteration (m,n)
$W_k^{m,n}$	Total attempted TP across link k due to nodes in $V_k^{m,n}$
$\Delta_{ik}^{m,n}$	Link k BW allocated to node i during iteration (m,n)
$p_{ik}^{m,opt}$	Optimal TP of node i across link k after iteration m
$p_i^{m,min}$	Minimum TP allocation to node i after iteration m.
A_{ik}^{E}	Total node i TP assigned to link k, including echo packets.
p_i^{opt}	Optimal TP of node i without echo packets.
p_i^{optE}	Optimal TP of node i with echo packets.

Table 1: : *Iteration Variable Definitions*

$$p_{ik}^{m,n} = \begin{cases} 0 & n = 0 \\ p_{ik}^{m,n-1} + \Delta_{ik}^{m,n} & n > 0 \end{cases}$$

$$W_k^{m,n} = \sum_{i \in V_k^{m,n}} t_{ik} ,$$

where $V_k^{m,n}$ is the set of all nodes i for which $A_{i,}^{m,n} > 0$.

$$\Delta_{ik}^{m,n} \equiv \min\left(A_{ik}^{m,n-1}, \frac{t_{ik}}{W_k^{m,n}} U_k^{m,n-1}\right)$$

$p_{ik}^{m,opt} = p_{ik}^{m,q}$ for the smallest q satisfying $U_k^{m,q} = 0$.

$$p_i^{m,min} = \min_{k:\, t_{ik}>0} \left[\frac{p_{ik}^{m,opt}}{t_{ik}} \right]$$

Finally, the optimal throughput of a node i (in *words/cycle*) is given by $p_i^{opt} = \alpha p_i^{q,min}$ for the smallest q satisfying

$$\min_{k:\, t_{ik}>0} \left[U_k^{q,0} - \sum_j t_{jk} p_j^{q,min} \right] = 0 ,$$

where $\alpha = \left[\frac{L_i^s}{L_i^s + 1} \right]$ accounts for the postpending idle.

4.3 Optimal Throughputs With Echo Packets

When echo packets are used, the throughput of each node will be lower due to the additional network load. The added load is quantified by the following expression:

$$A_{ik}^{E} = \frac{p_i^{opt}}{\alpha} \left[T_{ik} + \left(\frac{L^e + 1}{L_i^r + 1} \right) (1 - T_{ik}) \right] ,$$

where α and L_i^r are as defined previously, and L^e is the length of an echo packet.

The optimal throughput of a node i (in *words/cycle*) when echos are used is then finally given by

$$p_i^{optE} = p_i^{opt} \min\left(1 , \min_{k:\, t_{ik}>0} \left\{ \frac{1}{\sum_j A_{jk}^{E}} \right\} \right) .$$

5 Results and Comparisons

5.1 Evaluation Models

In this section we compare the performance of our enhanced flow control protocol to that of the standard SCI protocol and to the optimal performance, as defined in Section 1.3 and, more formally, in Section 4. All results were obtained through a detailed network simulation, which we have verified by an analytical model [4]. Although developed independently, the results produced by our model are consistent with those presented in [5].

Although SCI supports a variety of packet sizes, current VLSI technology will most likely constrain the first practical implementations to a maximum packet size of 80 bytes, or 40 16-bit words or *symbols*. Since an SCI packet contains 8 words of header, an 80-byte packet carries 64 bytes of data.

Networks supporting only a single packet size are becoming increasingly popular as higher bandwidth demands force simpler implementation circuitry. This is especially true when the traffic is largely numerical data, such as in a digital signal processing system. Indeed, this type of application inspired the protocol extensions described in this paper, and we therefore expect it to perform best when primarily a single packet size is used. We shall, however, show that the enhanced protocol yields good performance for mixed packet sizes, as well.

The results in this section correspond to 4, 8, and 16-node systems containing either exclusively 80-byte send packets or mixed 80-byte and 16-byte send packets, with relative proportions of 40 and 60 percent, respectively, as assumed in [5]. Each send packet is acknowledged by an 8-byte echo packet.

Our simulator supports 8 transmission groups. Although we simply assigned each node in the 4 and 8-node simulations to its own group, we used the *Maximal Conflict Grouping Algorithm* of [3] to assign groups for the 16-node simulations.

All results assume that the attempted throughput of each node is 1 *word/cycle*. Heavy and uniform loading provides the best basis for comparison in this case, since we are concerned primarily with providing the highest possible throughput without sacrificing fairness, where it is required.

5.2 Performance Metrics

Table 2 defines the performance metrics used in this section. When a node gets more bandwidth than it should optimally receive, one or more other nodes necessarily receive less than their optimal throughputs. Hence, the *absolute deviation* metric tends to overestimate the actual magnitude of the deviation. To more accurately reflect the true magnitude of the deviation, we introduce the *adjusted* deviation, which registers only negative throughput deviations. All results herein will thus be given in terms of *adjusted deviation*.

Metric	Description
Absolute Deviation	%Dev. between Optimal and Obtained Throughputs
Adjusted Deviation	Throughputs greater than optimal are not considered
Ave. Ring Deviation	Ave. over node devs. on a ringlet
Max. Node Deviation	Max. over node devs. on a ringlet
Ave. Max-Deviation	Ave. over max. node devs. in several cases

Table 2: *Performance Metrics*

5.3 Best-Case Performance Improvement

Because the enhanced protocol is designed to increase the throughput of nodes that conflict minimally with other nodes, we expect it to perform best for highly pipelined traffic scenarios in which several nodes usually transmit to their immediate downstream neighbors, but *at least one* node transmits through one or more nodes. In general, the worst-case scenario under SCI flow control occurs when $f \in [2..N-1]$ consecutive nodes all transmit to the same node, while the remaining N-f nodes all transmit to their immediate downstream neighbors without any conflict. We refer to f as the *fan-in*. It can be shown [3] that the enhancement is expected to yield the largest improvement when $f = \sqrt{N}$. This is the worst-case fan-in under an ordinary fairness protocol.

Table 3 shows the average and maximum deviations for 4, 8, and 16-node ringlets with worst-case fan-in. Note that in each case, the performance of the enhanced protocol is, on average, nearly optimal. Although the maximum adjusted deviation is somewhat high for the 16-node ringlet with mixed packet sizes, it is much lower than the deviation under the standard protocol.

Ringlet Size	Flow Control	Single Packet Size		Mixed Packets Sizes	
		Ave Dev	Max Dev	Ave Dev	Max Dev
4 Nodes	None	24.97	99.70	25.04	99.27
	SCI	21.73	43.47	25.08	54.65
	Enhanced	0.19	0.28	5.66	19.76
8 Nodes	None	24.64	98.26	24.73	99.46
	SCI	33.14	53.02	27.23	54.10
	Enhanced	0.18	0.82	4.74	26.54
16 Nodes	None	18.06	96.28	17.82	98.74
	SCI	39.79	53.50	18.37	98.74
	Enhanced	0.01	0.17	1.91	16.98

Table 3: *Flow Control Performance on Semi-Pipelined Ringlets with Worst-Case Fan-In*

5.4 Overall Performance

The performance of adaptive flow control protocols, such as those considered here, tends to vary widely with the traffic scenario. Hence, to accurately judge a protocol's performance, we must consider it under a wide range of traffic scenarios. We thus take two approaches to choosing cases for comparison.

First we view the performance of a 4-node ring under all possible *deterministic* traffic scenarios. Deterministic traffic refers to the case in which a given source node always transmits to the same destination. Considering only deterministic cases, for which there exist only 81 possible traffic matrices for a 4-node ring, allows us to average our results over every possible class of traffic. In fact, due to symmetry, only 24 of these patterns are unique.

Figure 3 compares the performance of both protocols on a 4-node ringlet for all 24 traffic patterns.The samples are arranged in order of increasing improvement offered by the enhanced protocol over the original. The numbers along the x-axis represent the ratio of the adjusted deviation under standard SCI to that under the enhanced scheme. A ratio of 1 indicates that the protocols perform the same under that traffic scenario. The number on top of each set of bars indicates that pattern's relative occurrence within the 81 possible patterns. Patterns with higher relative occurrence rates have a greater impact on the average deviation overall.

Note that the relative occurrence is high in all cases for which the enhanced scheme provides significantly improved performance, as compared to only one case in which, for constant-length packets, it provides significantly worse performance. It is also important to note that in the case for which the enhanced scheme has the greatest performance degradation over SCI, the deviation is only 2.5% for the constant-length packet case, and 6% for mixed packet sizes. We note that in these cases, the performance of SCI's protocol can always be obtained by placing all nodes into a single group.

The results in Figure 3 suggest that the enhanced scheme will, on average, perform significantly better than the standard SCI scheme, at least on a 4-node ring. As predicted, the results are most significant for the system containing constant-length packets. This is because varying transmission lengths, in turn, cause a large variance in the latency of flow control information. When the traffic load is heavy, the latency in circulating flow control information can actually help both protocols to enforce round-robin access. However, when this latency varies widely and traffic is asymmetric, both protocols tend to bias certain nodes, allowing them to transmit more often than others.

Although many more traffic patterns are possible on 8 or 16-node rings, they fall into the same classes as those covered in the 4-node case. Even the relative occurrences remain the same. This suggests that the results in Figure 3 are likely to be similar to those which we would obtain for 8 and 16-node rings. The next series of graphs indeed confirms this to be true.

Our second approach to covering a wide range of traffic scenarios is to randomly generate a number of traffic matrices, and average over the results for each case. Figure 4 shows the average adjusted deviation overall for 4, 8, and 16-node ringlets. The results for 4 nodes include the cases represented in Figure 3, as well as randomly selected non-deterministic cases. In all cases, the enhanced protocol performs significantly better than the standard protocol, with the greatest improvement for the larger ring sizes.

Table 4 shows the maximum and average maximum deviations for the same cases as Figure 4. Note that the enhancement offers significantly improved performance in

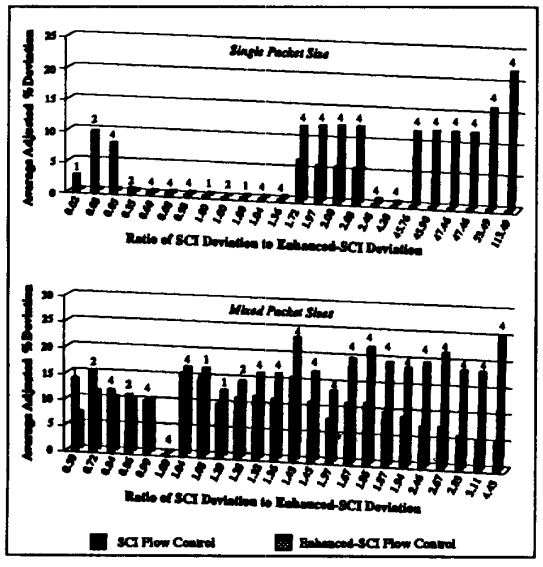

Figure 3: ***Performance Comparison for all Possible 4-Node Deterministic Patterns***

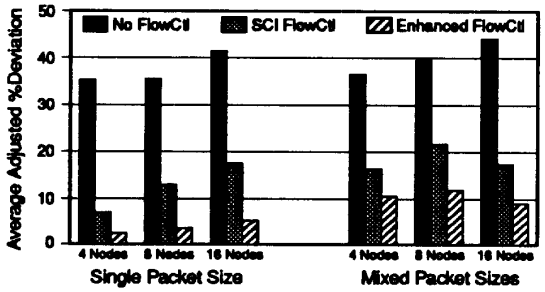

Figure 4: ***Average Deviation Overall***

Ringlet Size	Flow Control	Single Packet Size		Mixed Packets Sizes	
		Max Dev	Ave Max Dev	Ave Dev	Ave Max Dev
4 Nodes	None	99.75	82.65	99.85	87.65
	SCI	63.52	25.01	87.43	38.12
	Enhanced	25.49	6.96	47.23	27.28
8 Nodes	None	99.43	74.64	100.00	83.44
	SCI	83.66	40.69	77.01	51.20
	Enhanced	26.79	4.95	32.92	27.37
16 Nodes	None	99.43	79.26	100.00	91.62
	SCI	92.40	58.43	98.74	69.98
	Enhanced	19.66	8.97	38.94	25.79

Table 4: *Maximum Flow Control Deviations Overall*

all cases. With mixed packet sizes, the maximum deviations of the enhanced scheme, though still far less than the standard scheme, consistently increase. This is due to the variation in the circulation time of flow control information, as discussed earlier. The enhanced scheme is more sensitive to this latency because large distances may separate members of a particular transmission group. Hence, this effect is most pronounced when groups contain only a single member.

5.5 Symmetric Traffic

In the case of *symmetric* traffic, in which every node transmits to all other nodes with equal probability, flow control is not required and, in the case of *both* protocols, tends to *degrade* the attainable throughput somewhat. This is demonstrated in Figure 5. Note that the average deviation under the enhanced protocol is actually slightly *worse* than for the standard SCI protocol in this special case.

6 Summary and Conclusions

This paper presented an enhancement to the SCI fairness protocol that increased throughput while maintaining fairness among conflicting nodes, providing what we defined as a *relaxed fairness* service discipline.

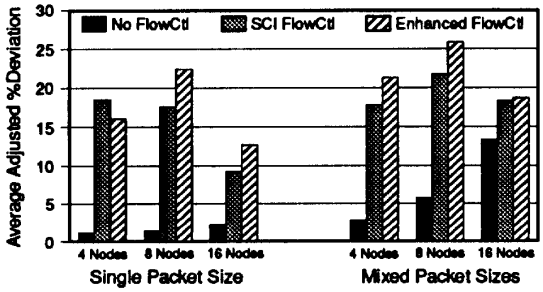

Figure 5: *Average Deviation for Symmetric Traffic*

We presented exact equations for the optimal throughput of each node on a ringlet governed by *relaxed fairness*, and used them to evaluate both the absolute performance of the enhanced protocol, and its performance in relation to the standard SCI protocol.

The goal of the enhanced protocol is to provide improved throughput over the standard protocol when possible, while guaranteeing performance at least as good as the original in other cases. In Section 5, we showed that our protocol indeed achieves significantly increased throughput in most of the cases studied. The performance decreased somewhat in the presence of mixed packet sizes.

In a few cases the enhanced protocol performed worse than the standard protocol. We attribute this to latency in the circulation of flow control information. Because our protocol often increases the throughput of several nodes on a ring, the idle symbols that travel between packets are subject to a higher latency in circulating the ring. Although our protocol acts to compensate for this, in some cases low bandwidth or unfair bandwidth allocation results. We stress, however, that the performance of the enhanced protocol can always be made to equal that of SCI's by placing all nodes in a single group.

In conclusion, we have shown that our enhancement to the SCI fairness protocol can significantly increase the already unsurpassed performance promised by SCI. Because the enhanced protocol simply builds upon SCI, it could easily be incorporated into future extensions of the specification.

Acknowledgments

We gratefully acknowledge the support and encouragement of Y.S. Wu of the Naval Research Labs, Dave James, Mac MacDougal, Kathy Nichols, and Glen Stone of Apple Computer, Inc., and Dr. Walter Ku, Director UCSD ICAS Center.

References

[1] David B. Gustavson, "The Scalable Coherent Interface and Related Standards Projects," IEEE *Micro*, Vol. 12, No. 1, Feb. 1992, pp. 10-22.

[2] IEEE Std. 1596 Scalable Coherent Interface (SCI). For copies call the IEEE Publications Office, 1-800-678-4333.

[3] D. Picker, R.D. Fellman, and P.M. Chau, "An Extension to the SCI Flow Control Protocol for Increased Network Efficiency," Submitted, IEEE/ACM *Transactions on Networking*.

[4] D. Picker, "Interprocessor Communication in a Ring-Based Heterogeneous Multiprocessor System for DSP", Ph.D. Dissertation, To be submitted.

[5] S.L. Scott, J.R. Goodman, and M.K. Vernon, "Performance of the SCI Ring," *19th Intl'l Symp. on Computer Architecture*, May 1992.