A Low Latency Modular Approach to Designing InfiniBand Fabrics Used in MPI Clusters

Stephen S. Fried
President and CTO
Microway, Inc.

Paul G. Howard, Ph.D.
Chief Scientist
Microway, Inc.

Abstract

Microway has developed a family of InfiniBand-based FasTree™ switches that work with conventional HCAs and the PathScale's InfiniPath™ connector. The first switch in the family employs a pair of Mellanox 24-port 4X InfiniScaleIII™ devices. It can be used to build low-latency switches in the 32- to 84-port range or as a building block used to create large distributed non-blocking "fat tree" networks. When building large switched networks, the use of FasTree components for leaf nodes and spines has a number of advantages over the popular monolithic fat tree design. The advantages include modularity, reduced latency, extensibility, and improved cable management.

FasTree™ design

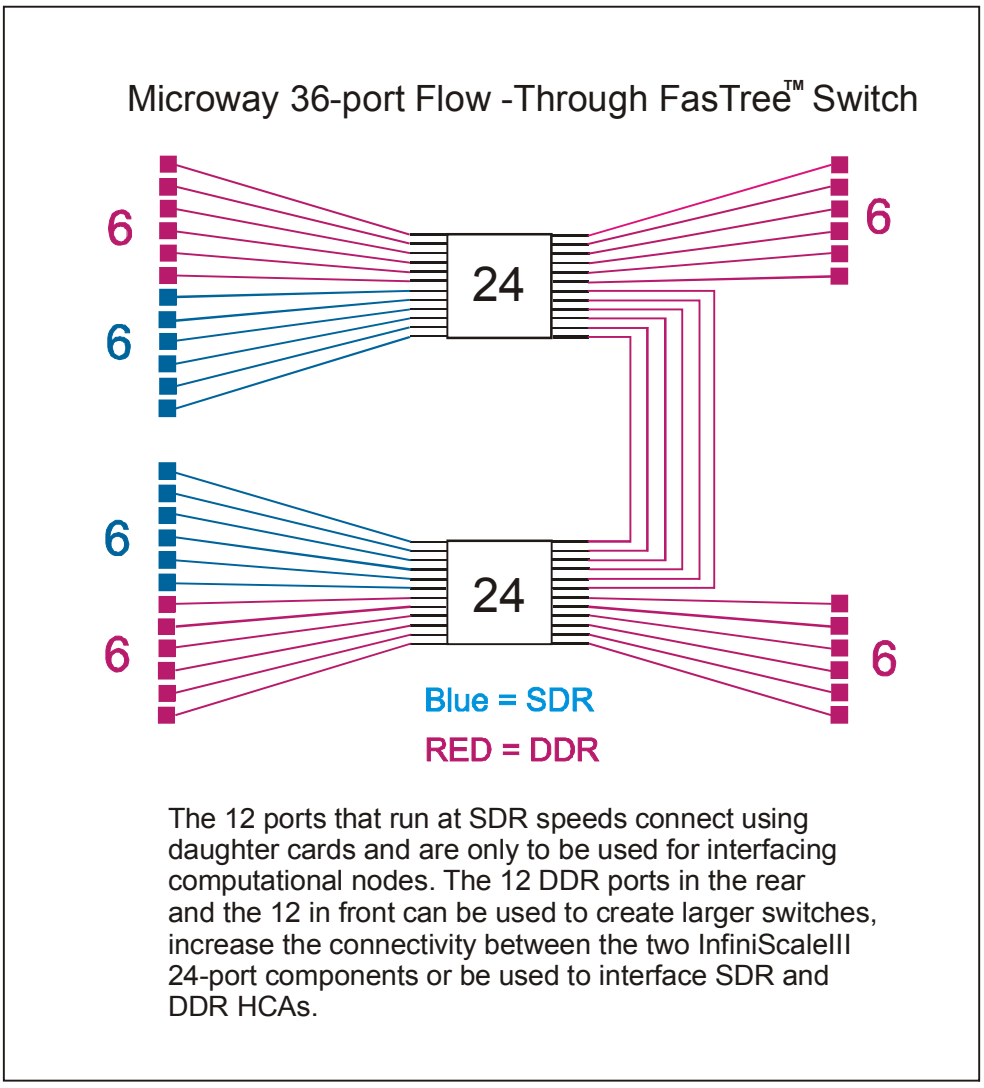


Figure 1: Microway 36-port FasTree Switch

Microway's basic FasTree building block is a 36-port switch that contains two Mellanox InfiniScaleIII components. The InfiniScaleIII is a single chip device that provides 24 ports running at 4X speeds at SDR (2.5 GHz) and DDR (5.0 GHz) rates. The DDR link frequency of 5 GHz provides the equivalent throughput of 20 GigE links. The other useful component in the family is a 24-port DDR switch that employs a single InfiniScaleIII component.

The two InfiniScaleIII switches in our 36-port switch (see figure 1) are connected with six DDR cross links, leaving 24 DDR and 12 single data rate (SDR) ports available for external connections. (The SDR ports are physically located on daughter cards, and must be run slower to preserve signal integrity. They are only used to connect computational nodes to the switch and not to interconnect switches). This topology provides balanced bandwidth across and through the switches when the 24 ports in the front are connected to 24 SDR Host Channel Adapters (HCAs). The 24 SDR input ports are balanced in the bandwidth sense by the 12 DDR links coming out the rear. However, each of the switches are also balanced internally, as the 6 DDR links connecting the two switches are just enough to handle cross traffic generated by the 12 SDR links on either side. Hence the 36-port FasTree switch behaves like a flow-through device while providing low latency connections between the 24 HCAs that are locally connected to the switch.

The 24 external DDR ports are what give the FasTree architecture its flexibility. They may be connected to cluster nodes through DDR or SDR HCAs; they may be used as additional links between the two InfiniScaleIII components; or they may be used to create larger switches. The switches are completely user configurable simply by re-cabling.

Fat trees and spine-leaf topologies

Very large InfiniBand clusters typically use a spine consisting of a number of large, fully connected ("fat tree") switches and edge or leaf nodes that interface up to 12 nodes (i.e. HCAs) to the spine using 24-port switches. The fat tree concept is that there will always be enough bandwidth to connect any pair of leaf nodes simultaneously, so communication will never be delayed by contention. In effect, the fat tree guarantees full bisectional bandwidth.

This guarantee comes with a price, however. Since the spine switches are large (96 to 288 ports), all communication through the spine requires three hops, increasing latency. The leaf switches are typically small (24 ports, half connected to nodes and half to the spine), so the amount of localized low-latency intra-leaf communication is limited to 12 computation nodes, with the leafs adding two hops to the latency of the spine.

At the present time, there is a gap in the size of the InfiniBand switches that are commercially available in our modulo 24 InfiniBand world. The starting point is 24 4X ports. The next available sizes run between 96 and 288 ports. What that means is if you want to connect 32 to 64 nodes, the only practical choice is to buy a large switch and under populate. Another disadvantage of the classic fat tree approach when applied to InfiniBand is that it results in a centralized layout that concentrates the rather wieldy, but economic, 4X copper cables at a single point.

Guaranteeing full bisectional bandwidth may be appropriate in the web server world. For MPI-based HPC clusters, the FasTree architecture provides a more flexible, lower latency, convenient and cost-effective solution.

Large-cluster FasTree spine-leaf topologies

A 96-node FasTree cluster can be designed with four leaf nodes and two spine nodes. The four leaf nodes use our basic 36-port DDR switches, each providing 24 SDR links to computation nodes that are balanced by the 12 DDR links to the spine. The

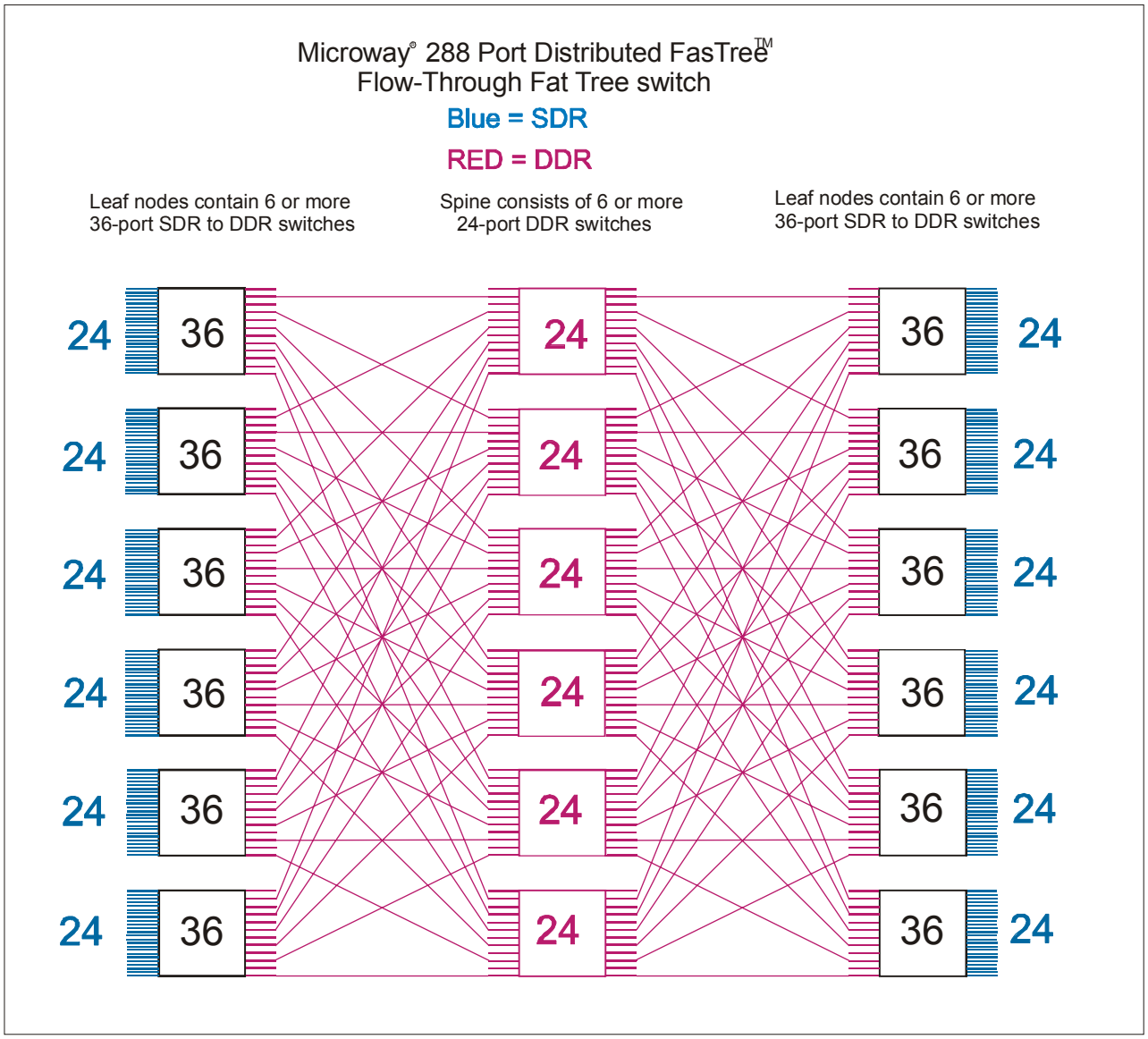


Figure 2: 288-node FasTree cluster

two spine nodes are simple 24-port DDR switches, each containing a single InfiniScaleIII device, with six of its DDR connections going to each leaf node.

As shown in Table 1, this design scales easily to 144, 192, 288, and 576 nodes, using more leaf nodes and spine nodes. The only difference is that the number of DDR links between

leaf nodes and spine nodes is reduced, but there is still enough bandwidth available so that there will be no contention up to 288 nodes and only limited contention at 576 nodes. (Contention at 576 nodes may occur when packets must travel between the two InfiniScaleIII devices within a 36-port switch.) Figure 2 illustrates the FasTree design for a 288-node cluster. Each line in the figure represents a pair of DDR cables.

Table 1: FasTree designs for large clusters

Nodes	Leaf nodes 36 ports	Spine nodes 24 ports	DDR links per leaf-to-spine connection
96	4	2	6
144	6	3	4
192	8	4	3
288	12	6	2 (see Figure 2)
576	24	12	1

Small-cluster FasTree topologies

Smaller FasTree topologies can be created using a small number of basic 36- port switches

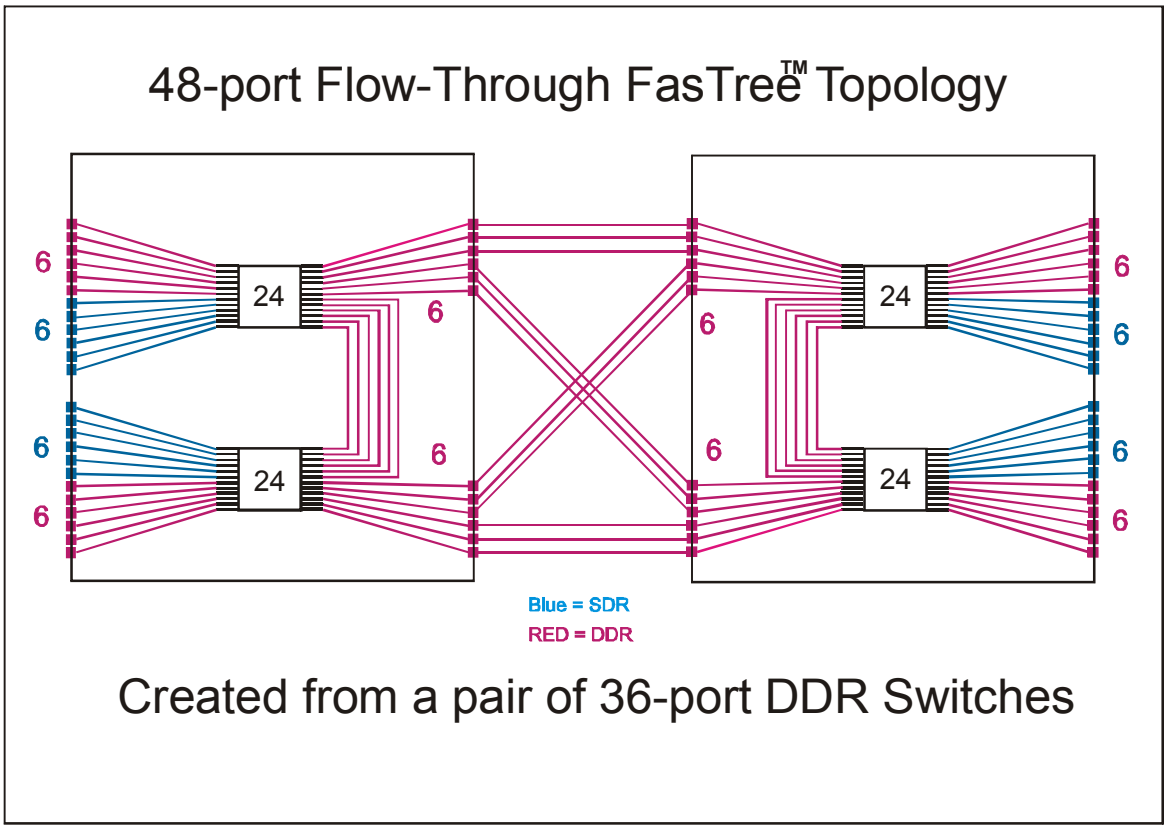


Figure 3: 48-port FasTree switch

connected by their DDR ports. Obtaining guaranteed non-blocked bandwidth is possible for 48 and 54 nodes using 2 and 3 switches respectively. However, by relaxing the full bandwidth requirement, it is possible to design a 54-node fabric with two switches, a 72-node fabric with three switches, and an 84-node fabric with four switches with only a very small risk of contention. Figure 3 shows a non-blocking 48-port FasTree switch consisting of a pair of 36-port switches connected by 12 DDR links. The big advantage of this approach is the latency, which is always better than an underpopulated Fat Tree and often employs single hop transactions in situations where nearest neighbor access predominates.

We have investigated the blocking behavior of small FasTree switches. In simulations assuming completely random communication patterns, we measured the percentage of patterns that resulted in blocking, that is, in at least one link receiving more traffic than its capacity. (In the simulations we always use the same number of links for each switch-to-switch connection. Some ports remain unused if the number of switches does not evenly divide the number of available links; our results are conservative in those cases.) The simulation results are shown in the Table 2.

Switches	Total nodes	Percent of patterns blocking	
2	48	0	(guaranteed nonblocking)
	50	0.01 %	
	52	0.08 %	
	54	4.8 %	
	56	14.8 %	(potentially blocked)
3	54	0	(guaranteed nonblocking)
	57	0.00 %	
	60	0.00 %	
	63	0.02 %	
	66	0.08 %	
	69	3.0 %	
	72	5.9 %	
	75	44.1 %	(potentially blocked)
4	56	0	(guaranteed nonblocking)
	60	0.00 %	
	64	0.00 %	
	68	0.00 %	
	72	0.00 %	
	76	0.47 %	
	80	0.96 %	
	84	1.74 %	
	88	11.6 %	(potentially blocked)

Table 2: Blocking results for small FasTree clusters

In real problems, blocking is much less likely than the simulation would suggest. Except for the latest HTX and PCI-E buses, HCA bandwidth is limited by the memory interface.

Often bandwidth contention is not an issue because of localized communication patterns, temporarily distributed communication patterns, or algorithms that are limited by computation or latency rather than bandwidth. For many problems (e.g., computational fluid dynamics) a nearest-neighbor design is more suitable than a fully connected graph. When processes are properly mapped to computation nodes, at least half of the communications in nearest neighbor problems are local. Similar arguments can be made for clusters shared by multiple users in which many problems are run at the same time within small sub domains in the cluster that are completely connected by a single leaf node. What we will discover below is that even for most fine grain problems, 24 nodes connected by 4X InfiniBand scale linearly, enhancing the value of our 36-port leaf node concept. The same can not be said for problems with 48 or more nodes, in which we start to approach the fine grain limit in CFD problems working with small data sets. For fine-grain parallel problems with many short messages, bandwidth is not an issue at all. Hence for many real clusters and HPC problems, the technically over-subscribed topology will perform just as well as a theoretically optimal design.

Single-switch FasTree designs

Even a single FasTree switch provides some design flexibility. When the 36-port switch is used in a 32-node cluster, there are 4 unused DDR ports. The two InfiniScaleIII devices are internally connected using 6 DDR links, and the unused ports can be used to provide two more DDR cross links, giving 16 SDR equivalent available bandwidth, which exactly balances the 16 external SDR ports on each InfiniScaleIII device; hence this design is theoretically nonblocking.

We ran a simulation similar to the one described above using two 18-port switches connected by 6 DDR links, effectively simulating the internal behavior of our 36-port switch. About 3.6 percent of all random connection patterns result in some blocking. When we ran MPI benchmarks (NAS benchmarks, described below) on a single switch configured as a 32 and 36 port device and then compared those results with a 48 port switch that was under subscribed (i.e. was non-blocking), no significant difference could be detected between the two cases for either the 32 or 36-port runs. These results lend further credence to our assertion that designing for guaranteed full bandwidth in the case of HPC MPI use patterns, is often overkill.

Benefits of FasTree technology

The main benefits of FasTree technology are modularity, reduced latency, distributivity, and extensibility. In addition, there are both physical and electrical cable management benefits.

The FasTree architecture enables modular design. Users are free to experiment with different technologies that might be better matched to their problems. With a set of FasTree switches, a user can take advantage of a topology with less-than-complete connectivity that is easily good enough to solve their problem at 100% efficiency.

The FasTree architecture enables a reduced-latency design in small clusters with 48 to 72 nodes. In a fat tree, all non-local communication requires four hops: node-to-switch-to-node, plus 2 hops within the switch. Using a FasTree, only three hops are needed: node-to-switch-to-switch-to-node.

This makes the FasTree topology ideal for use with products like PathScale's InfiniPath, where switch latency has to be kept low to take advantage of PathScale's very low latency HCA-like NIC. The latter can achieve 1.3 microsecond latency communicating through a single InfiniScaleIII component. In a typical device that uses low voltage differential signalling (LVDS) pairs, (e.g., an InfiniBand switch, the HyperTransport bus, the PCI-E bus), it typically takes 200 to 300 nanoseconds to transverse each device in the LVDS path. Cutting out a single hop in the InfiniPath case reduces the total latency by close to 20 percent.

A FasTree design is easily extensible. As a user's needs grow, the same switches can be used to turn a small, directly-connected cluster into a larger spine-leaf design, and the spine-leaf design can be easily expanded from 96 nodes to up to 576 nodes simply by adding switches in the obvious regular pattern.

The FasTree architecture is also future-friendly. Because InfiniBand components are interoperable even at different speeds, it will be possible in the future to use faster switches to aggregate the data collected by the leaves and spines of the existing InfiniBand fabric. With most other high-bandwidth, low-latency solutions, you cannot connect prior generations of switches and NICs to future generations. In the not-too-distant future, InfiniBand HCAs will run at DDR speeds and the connecting switches will run at quad data rate (QDR) speeds. The output lanes of Microway's DDR 36-port switch could be used in the future to feed a QDR spine, effectively doubling the bisectional bandwidth of our current generation spine nodes.

Another advantage of the distributed FasTree architecture is that the switches are physically spread out within the cluster, which spreads out the cabling and avoids the cable management issues that characterize centralized fat tree switch designs when used with bulky, heavy copper 4X InfiniBand cables. Note that the high cost of a 4X InfiniBand fiber cable ($300 more than copper SDR because of the cost of electrical to optical transceivers; even more for DDR) makes the use of fiber cables impractical, unless there is a need to move packets more than 15 meters. Fiber is expensive for InfiniBand, simply because of the amount of bandwidth that must be supported. Typical high performance switches in the past could get by with a single fiber channel pair running at bandwidths of 250 MB/sec. Increasing the bandwidth to 1,000 MB/sec (4X SDR) requires the addition of multiplexors to break the 4X signal into four parts, resulting in four times as many transmitters, fiber channel lines and receivers. Until the cost of these 4X components comes down, fiber channel in DDR InfiniBand solutions will be practical only for long distance communication.

In addition, at DDR speeds, the use of cables to create network connections instead of etch running within printed circuit boards (PCBs) makes it possible to produce switched networks with much better signal integrity, resulting in longer transmission lengths. The losses in 8 inches of PCB material at 5 GHz are roughly equivalent to the losses in a 10-meter equalized copper cable.

Benchmark results

We ran several tests from the NAS benchmark suite. We ran the Lower-upper symmetric Gauss-Seidel (LU) and Scalar Pentadiagonal (SP) tests at two different problem sizes, "A" (small) and "C" (large). We ran the tests for various numbers of processors, powers of two

from 2 to 64 for the LU tests, and square numbers from 4 to 81 for the SP tests. We ran FasTree InfiniBand tests with a number of different sized clusters, including 24, 32, 36, and 48 nodes. These included a 48-port non-blocked switch that was backed off to 32 and 36 nodes so we could compare its performance with a single switch running with 32 and 36 nodes. For comparison, we ran the same benchmarks using Gigabit Ethernet on the 48-node cluster with three 24-port switches. All the nodes were dual Opteron 248s with 4 gigabytes of memory.

The results are summarized in the four graphs below, where both the x-axis and the y-axis are logarithmic scales. The x-axis is the number of processes (we never oversubscribed the 96 processors available in the 48-node dual Opteron cluster.) The y-axis is work, that is, the product of wall clock time and the number of processes.

The interpretation of the graphs is the following. If the graph is a horizontal line (constant work), the problem scales linearly with the number of processes. If the graph goes down from left to right, the problem scales superlinearly on the given fabric. If the graph goes up, the problem is not scaling well on the fabric. Many curves are plotted on each graph, corresponding to different configurations; most of them are not individually labeled to avoid cluttering the graphs with descriptions of curves that are mostly close together and do not show any significant differences.

On all of the graphs, there is one curve that increases much more than any other for more than 36 processes: 36 nodes on one 36-port InfiniBand switch. After the run, it was found that during this run the InfiniBand cable between one of the nodes and the switch had a loose connection. Note that the benchmarks ran successfully, just much slower than all the other tests, by a factor of between 3 and 30. This points out the serious but silent performance loss that can occur when a seemingly minor cabling problem is present in a cluster. We were able to find the offending node in just a few seconds using Microway's MPI Link-Checker ™diagnostic software, but unfortunately only after the run.

There is an unexplained irregularity in the results for the SP "C" benchmark. All configurations show a peak at 36 processes in that test, with no indication of abnormality at 25 or 49 processes, the next smaller and larger numbers of processes.

For all four tests, performance using GigE was comparable to that of InfiniBand for small numbers of processes. But above 16 processes for small problems, GigE does not scale well, showing performance losses of a factor of 3 to 6 relative to InfiniBand. For large problems, the differences begin at about 32 processes, and are smaller but still noticeable, up to a factor of 3 for the SP "C" benchmark.

The FasTree InfiniBand fabric shows excellent scalability for these problems. For both small benchmarks it shows nearly linear speedup out to 64 processes and beyond, and for the LU "C" benchmark the speedup is superlinear. We surmise that in the LU "C" problem, as the number of processes increases, the subproblems on each processor become smaller, allowing better cache utilization. For the SP "C" problem there is a modest loss of efficiency for 64 processes, but much less severe than for GigE.

Benchmark conclusions

We had several questions in mind when we ran the NAS benchmarks. We were trying to discover at what point a typical fine-grain parallel problem starts to suffer from

communication latency and bandwidth issues, or in other words, at what point does it stop scaling well. In particular, we were trying to find how the scaling pattern for InfiniBand, a low-latency, high-bandwidth interconnect solution, compares with that for a standard Gigabit Ethernet implementation. We found that for small data sets, GigE quickly starts to have problems above 8 processors; more so for the SP test than the LU test. For the problems we ran, we did not see any tendency for the InfiniBand results to stop scaling until we were running more than 64 processors.

As many others have seen before us, we observed that fine-grain problems with small data sets show scalability issues long before large data set versions of the same problem. This is expected, since smaller data sets are less computationally intensive, which means that the computation stage of each iteration completes faster, increasing the impact of communications. We also discovered for LU benchmarks that the speedup is faster than linear in the number of processors; we speculate that this is a result of the data fitting into the cache.

To verify our assumptions about communications patterns and their impact on system performance, we ran our benchmarks on a number of different switch topologies, some of which in theory should have performed better than others. What we found is that the stochastic nature of parallel processing problems washed out the differences in topology. Carefully examining the data did not reveal any consistent results showing that a particular topology worked better than the rest. For some tests, in fact, topologies that we expected to be slightly inferior gave the best results. Another contributing factor here is the fact that most MPI users, never really take full advantage of the available InfiniBand bandwidth, simply because they are not computer scientists. To attain the 730 MB/sec that MPI Link Checker identifies as available, we had to properly align the data on page boundaries and when working with large packets, start hundreds of MPI sends at the same time. If we didn't, the numbers we measured fell to 580 MB/sec!

We have confirmed that parallel benchmarking is difficult. For any benchmark it is necessary first to get the system under test into a known state. For a serial benchmark running on a single uni-processor system, CPU designers know that for cycle counts to be meaningful one must start with flushed registers, caches and memory. For a parallel benchmark running on a cluster, not only every processor but also the communication fabric must be in a known state. While probably not an impossible task, it certainly was one we didn't have the time to tackle. The bottom line is that large networks of communicating processors are complex systems that display complicated and non-repeatable behavior. Aside from that, the choice of compiler and MPI implementation, and even the choices for compiler and MPI options and parameters, can have huge effects on benchmark results, overwhelming any small differences caused by different network designs. The bottom line is that it is very difficult to design parallel benchmarks accurate enough to measure performance differences as small as 5 percent between network designs.

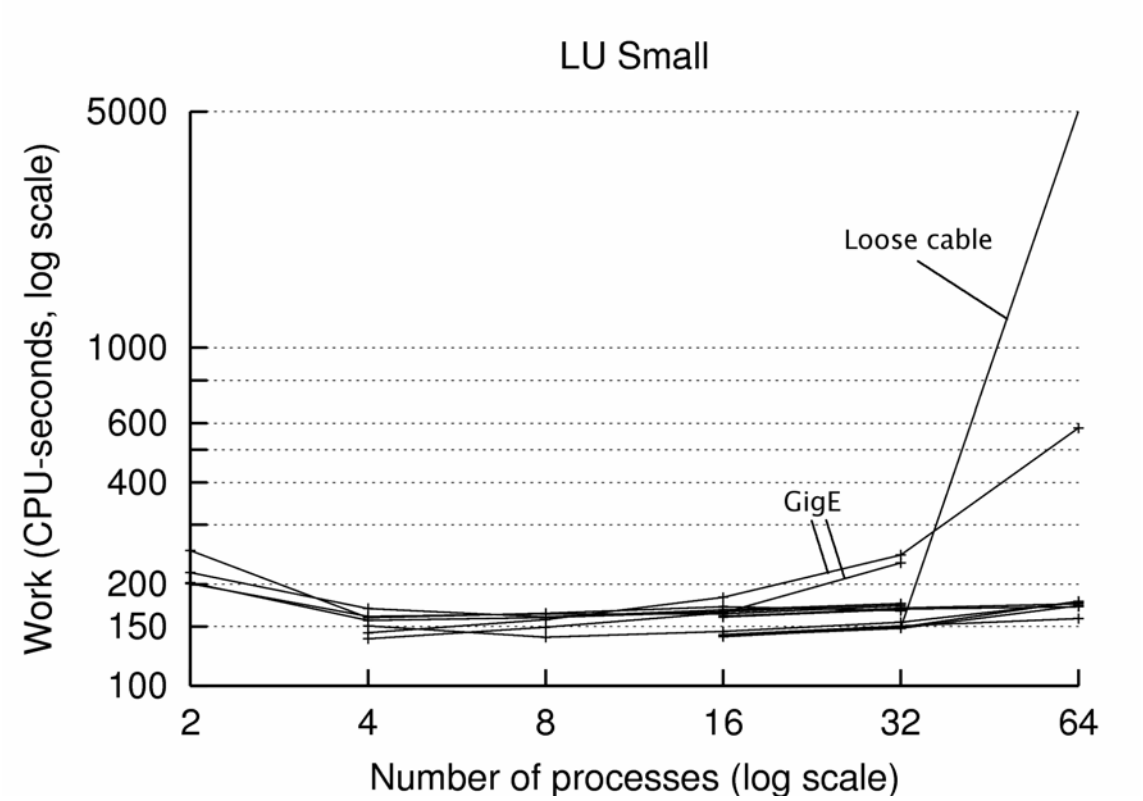
Conclusion

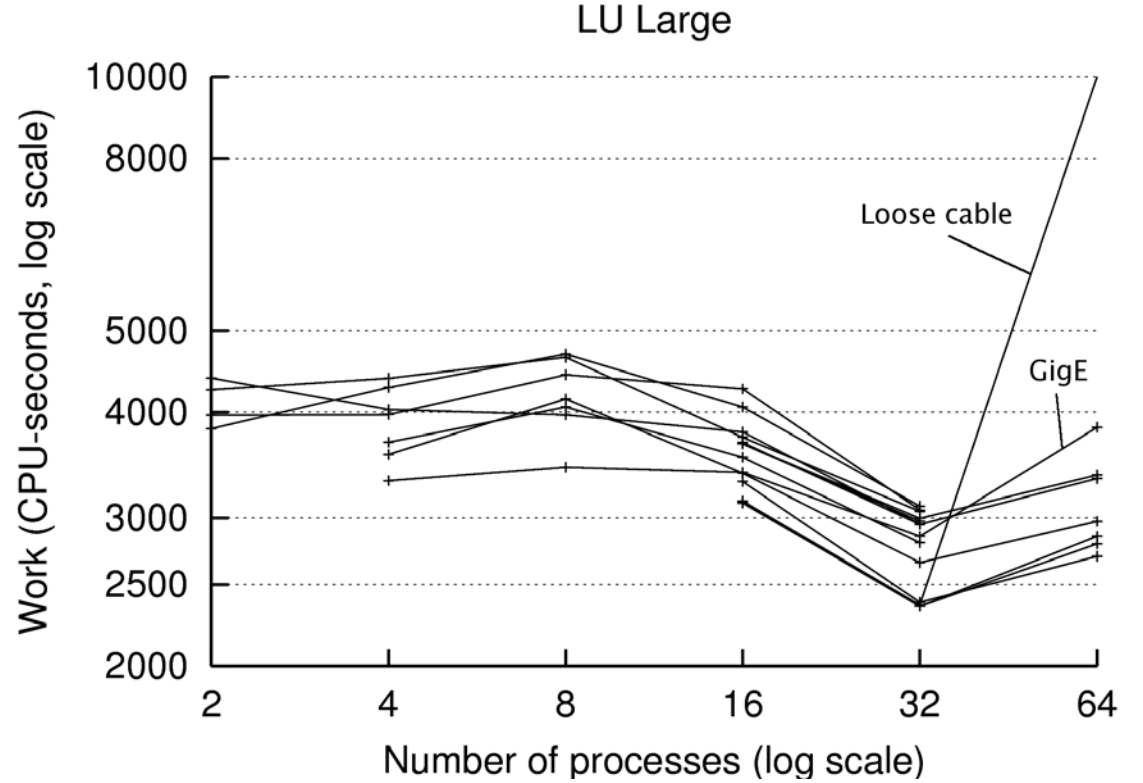
We have shown the Microway's FasTree architecture is a useful and interesting design for both large and small InfiniBand clusters. It is modular and easily configurable by users, allowing a fabric design suited to any particular problem. It lends itself to economical

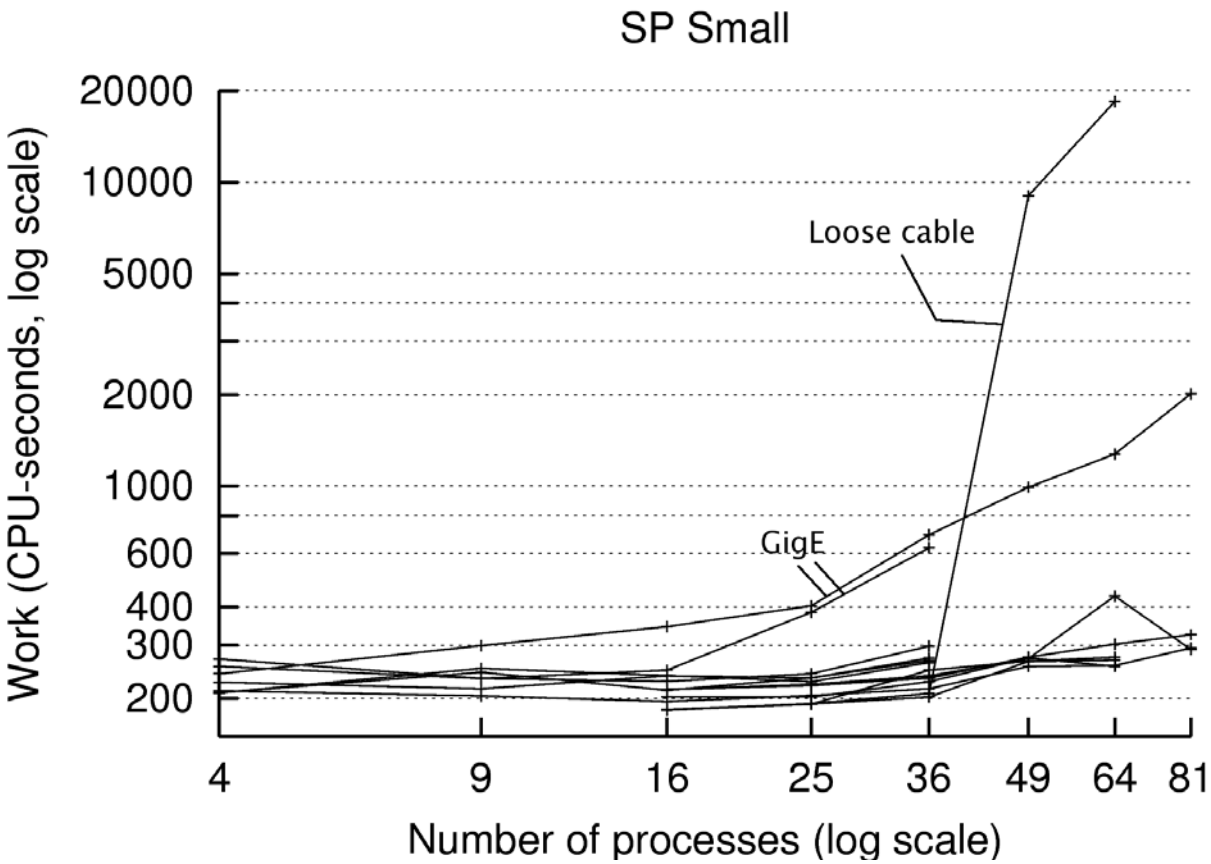
topologies with less than full connectivity, and is easily extended as the user's needs grow. Furthermore, the FasTree family is interoperable with next-generation InfiniBand technology; it will not be made obsolete, for example, by the emergence of quad data rate components. Overall, it is much more flexible than the traditional fat-tree-spine architecture.

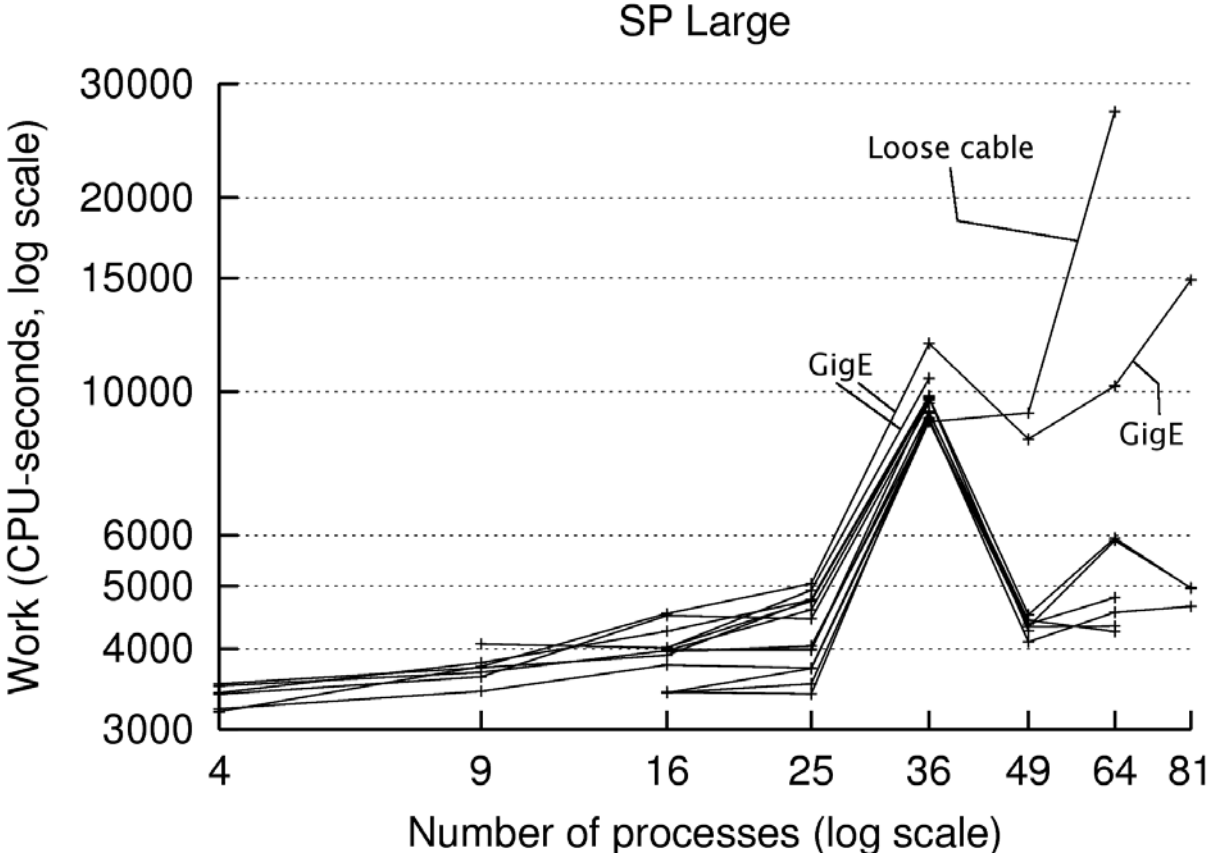
Our benchmark tests confirm that InfiniBand wins over Gigabit Ethernet for all but the smallest clusters. GigE begins to scale badly above 8 processes, while InfiniBand scales well up to 64 processes. We also have seen that for real problems most topologies work equally well up to the precision of the NAS benchmarks, so the economical InfiniBand designs made possible with the FasTree family can be recommended for any medium or large cluster.

Pair of 36 Port Switches configured as a 48 Port Switch using their DDR rear ports


LU Small
Work (CPU-seconds, log scale)
5000
1000
600
400
200
150
100
Loose cable
GigE
2
4
8
16
32
64
Number of processes (log scale)


LU Large
Work (CPU-seconds, log scale)
10000
8000
5000
4000
3000
2500
2000
Loose cable
GigE
2
4
8
16
32
64
Number of processes (log scale)


SP Small
Work (CPU-seconds, log scale)
20000
10000
5000
2000
1000
600
400
300
200
Loose cable
GigE
4
9
16
25
36
49
64
81
Number of processes (log scale)


SP Large
Work (CPU-seconds, log scale)
30000
20000
15000
10000
6000
5000
4000
3000
Loose cable
GigE
GigE
4
9
16
25
36
49
64
81
Number of processes (log scale)