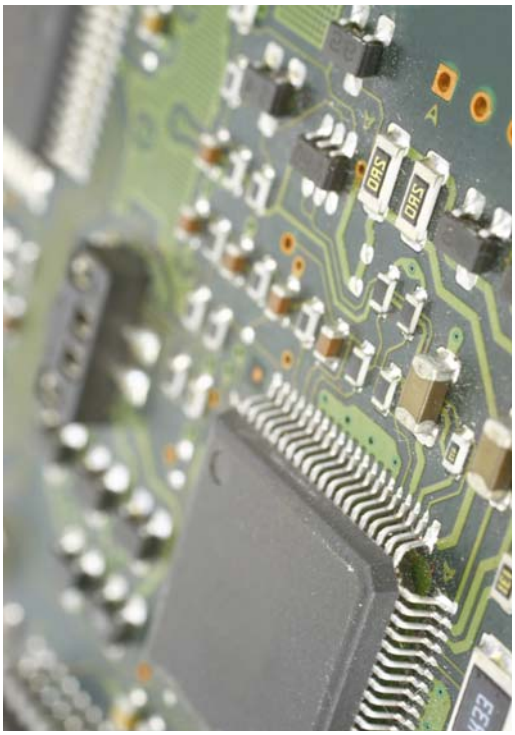
STAC Network Research and "Study A"

Network I/O is crucial to the speed of automated trading applications. This includes network hardware and drivers, as well as the APIs they expose and how applications are designed and threaded.

STAC has long published research and tools to assess network I/O performance in combination with particular feed handlers (STAC-M1) or messaging middleware (STAC-M2). However, many trading groups write applications directly to a network interface. Members of the STAC Benchmark Council have voiced a need for research and tools that focus directly on the network, with no intervening applications.

So that's what we've started to do. Our first project in this area, "STAC Network API Study A," focuses on a question that receives too little rigorous, unbiased treatment: network APIs and their implementations. This project sought answers to several questions:

- What exactly are the **performance, complexity, and resource** tradeoffs of UDP, TCP, RDMA, and RDS under low-latency trading-type workloads?
- What are the tradeoffs of the **different function calls** available within those APIs?
- What are the tradeoffs of **different usage patterns** (for example, adapting an existing Sockets app to use RDMA by copying data into a mapped memory region versus re-writing the app to use RDMA natively)?
- How do performance and resource utilization vary with **different implementations** of those APIs on both QDR InfiniBand and 10 Gbps Ethernet)?



To conduct these tests, STAC established a tools framework that is **neutral** with respect to vendor, API, and network type. Tentatively called STAC-N, it provides a load-generation and time-stamping library that can be bound to any network interface, together with tools that automatically execute test runs, find max message rates, and produce reports containing detailed latency analysis. STAC collaborated with leading vendors on the Council and other experts to bind the STAC-N Library to Sockets, RDMA, and RDS.

The results—a **comparison report plus detailed reports on 23 combinations** of API, physical layer, and usage pattern—are available in the STAC Vault to premium members of the Council. The toolset, including **source code to the bindings** is also available for inspection and use as examples of low-latency coding for each interface.

As the next step, STAC will solicit feedback from Council members on the test methodology, the source code, **what would be useful to test next**, and whether STAC-N should be taken forward as a standard.

To request the materials or indicate interest in participating, please visit:

www.STACresearch.com/network