STATUS OF THE NEXT GENERATION DATA ACQUISITION SYSTEM AT THE NSCL

Eric Kasten

1. Introduction

The status of the design and implementation of the next generation data acquisition system for the NSCL is discussed in this paper. In order to provide an understanding of the requirements and limitations of current data acquisition systems and hardware, an exploration of current research was pursued and performance testing was conducted using drivers either constructed or modified by the author. The results of these explorations and tests are outlined below and a preliminary system design outlined. The working name for this project is Spectrodaq.

2. Background

In order to provide background for the design and implementation of the next generation data acquisition (DAQ) system at the NSCL, prior research was investigated. This investigation revealed that DAQ systems that exploit the aggregate capabilities of distributed computer systems are becoming common [1,3,4]. The use of commodity PC hardware and the Linux operating system is being adopted in several cases [1] demonstrating, that commodity hardware and operating systems are an effective alternative to real-time systems and can be effectively used for data acquisition. High speed networking technologies using standard network protocols [5,6] are replacing in-house, custom networks and protocols.

This preliminary research provided the background needed to outline an initial approach for designing and implementing the next generation DAQ system at the NSCL. First, it is desirable to have a distributed system that uses standard operating systems and network technologies. Such a system is scalable, allowing for the addition of hardware to provide greater aggregate processing power or network bandwidth. Also, the upgrade path for replacing older hardware and networks with newly available technologies no longer requires that custom protocols and hardware be reworked. Second, there needed to be further exploration of VME-to-PCI hardware and current network technologies.

3. Hardware and Drivers

In this section, VME-to-PCI hardware and drivers are discussed and some comparison tests presented. Second is an exploration of gigabit ethernet and a short discussion on the potential impact of network technologies on the DAQ system.

3.1. VME to PCI Hardware and Drivers

One early design conclusion was that the readout controllers (ROCs) for the DAQ system were going to be built using PC style hardware. A number of reasons, including low cost, adequate performance and flexibility of software design made this decision attractive. The low cost of these systems also provides the benefit of being able to dedicate systems to particular tasks. For instance a PC can be a dedicated ROC.

A number of different operating systems are available for these platforms including our two primary contenders: Linux and NT. It was decided to try to stay with a standard operating system kernel as opposed to a real time kernel if adequate performance levels could be maintained. This was principally to avoid the additional overhead and difficulties that arise from using, designing and maintaining real time kernels.

Many of the new generation DAQ systems are distributed. This requires that a ROC both collect data from the system and transmit it to other systems. This type of system requires that both the readout and transmission processes as well as supporting subsystems have adequate priority to complete their tasks in a timely manner. Dedicated hardware with a standard preemptive operating system is likely to provide for these requirements.

There are a number of VME-to-PCI hardware bridges coming onto the market. Our current choice is the Bit-3 Model 617 Adapter produced by SBS Bit-3 Operations. An initial Linux driver was acquired from NIKHEF and modified by the author to support the new Linux PCI interface, stabilize driver operation and providing proc filesystem access for control and monitoring. Several performance tests were done comparing Linux and NT with regard to interrupt and polling latencies. Polling latencies were comparable, both systems demonstrating latencies on the order of 4.6 microseconds. Typical interrupt latencies were 300 microseconds under NT running on 200MHz Pentium II as compared to 43.4 microseconds under Linux on a 400MHz Pentium II. Even when scaled with regard to the CPU speed, Linux demonstrates a significantly lower interrupt latency than that of NT.

Figure 1 is a plot of VME memory bandwidth using memory mapping from Linux using the Bit-3 Adapter. This performance is comparable to that seen under NT. The bandwidths shown during this test are considered adequate for the next generation DAQ system.

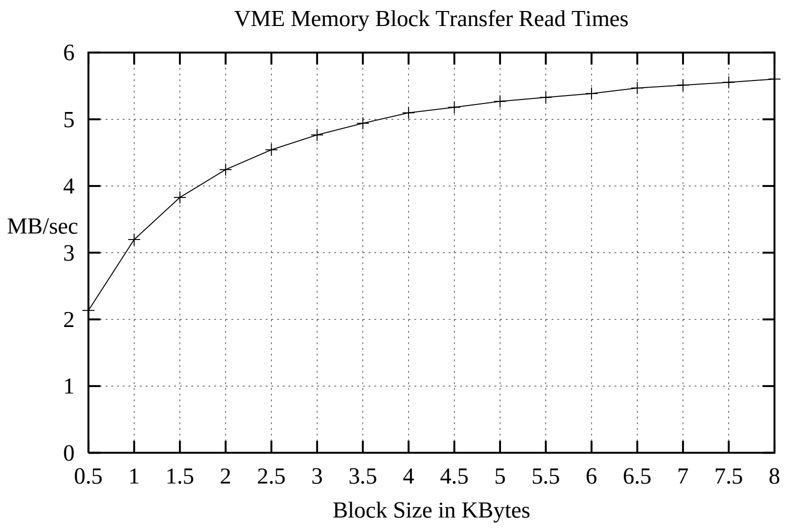

Figure 1: VME Memory Mapped Bandwidth

Linux also has a number of other advantages over NT. Linux provides easy remote access in the form of telnet, ftp and exported X displays among others. NT also provides some types of remote access, but these are typically available only with the NT Server installation or as add on products. These options are more costly and are often less mature than many of the comparable Linux/Unix counterparts. Linux provides a cleaner more direct interface for constructing device drivers as well as providing complete kernel source in the event that the operating system should need to be customized.

Considering both the advantages and the improved interrupt latencies, Linux was chosen for the the ROC operating system.

3.2. Networking

It is anticipated that network bandwidth and latencies may also impact the overall performance of the DAQ system. With this in mind experimentation using gigabit ethernet technology was pursued. Using a FDR-12 repeater and Hamachi gigabit ethernet cards provided by Packet Engines, Inc., client-to-server latency and bandwidth experiments were conducted between a 400MHz Pentium II with 64MB RAM and a 450MHz Pentium II with 128MB RAM. The preliminary driver for the Hamachi card was acquired from CESDIS and further modified to increase stability and throughput by the author. Additionally, test programs were constructed to provide performance testing.

Figure 2 shows how bandwidth varies with kernel buffer size and application message size. This reveals that bandwidth increases with increased kernel buffer sizes and that application messages sizes above 40 KBytes improve bandwidth only slightly. A maximum client-to-server throughput of approximately 320 MBytes/sec was witnessed during this experiment.

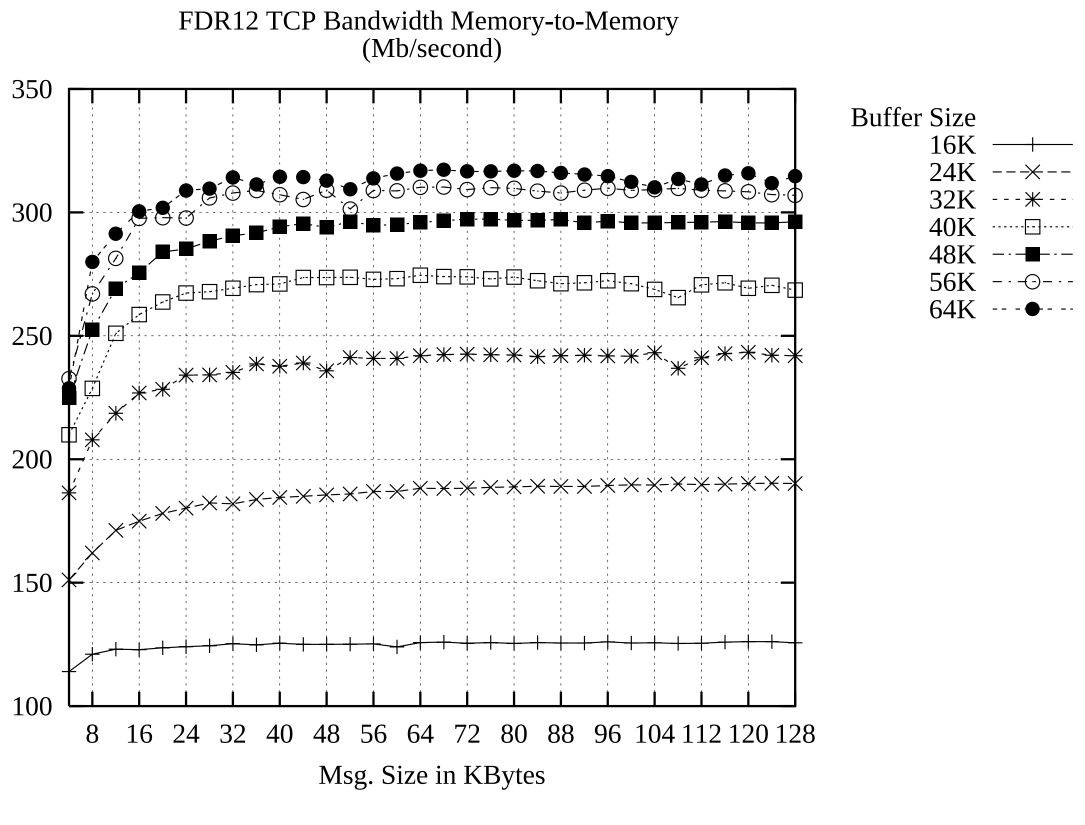

Figure 2: Hamachi Bandwidth vs. Message Size

Figure 3 is a plot of latencies using a ping-pong test measured from the start of the transmission to the start of reception. This experiment shows how latency varies with small message sizes and with different interrupt mitigation settings. The Rx values represent increasing minimum delays between interrupts generated by the Hamachi card for receiver events. This shows that for message sizes between 8 and 128 bytes latencies remain near constant for each Rx value.

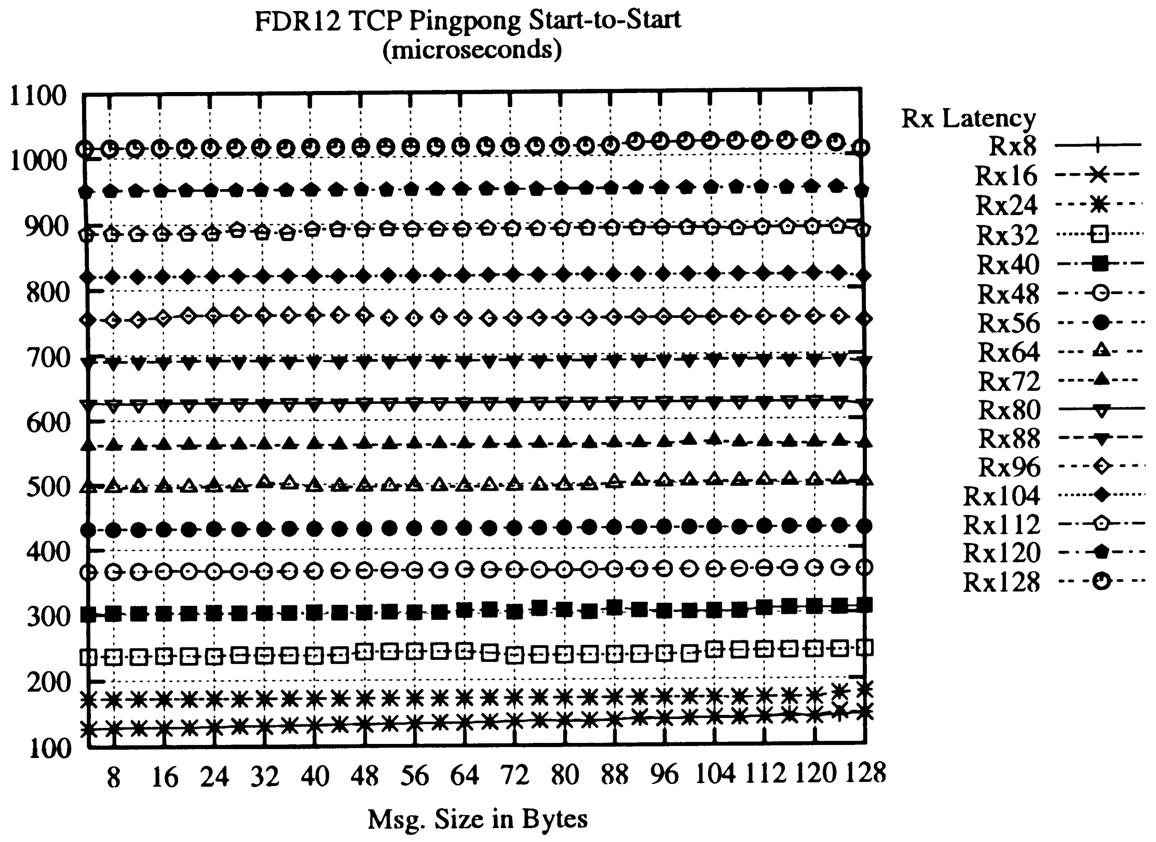

Figure 3: Hamachi Ping-Pong Latencies

Further experimentation should be done using other networking technologies, such as ATM, to provide a basis for comparison. This comparison can then be used to help determine the best networking solution for the different segments of the DAQ system. It is expected that there may be different requirements for certain segments of the system than for others. For example, greater bandwidth requirements may be necessary between the systems that participate directly in the acquisition of data than those that monitor the system.

4. DAQ System Design

The general design of the DAQ system is both that of a distributed and a client/server system. The system should be distributed in that it could support multiple PCs running as both dedicated ROCs and processing nodes. Processing nodes could be used to provide secondary massaging of data as read and transmitted by ROCs or other nodes. This structure helps provide for scalability in that it is possible to add additional ROC and processing nodes to help provide the aggregate CPU power or network bandwidth to handle larger data loads or additional processing. A distributed system also allows for geographic distribution of readout devices while allowing for the coherent assembly of event data.

The system would be client/server in that control, monitoring and analysis clients could access the distributed core of the system through client/server interfaces. A client/server interface makes it possible to have clients that run on a wider variety of operating systems and platforms even though the core system may run atop a particular class of platforms such as Linux/Unix. This helps simplify the design and implementation of the core system while allowing users access through a wider variety of desktop systems.

Figure 4 is a simplified diagram of the basic DAQ system design. In this figure, it can be seen that the ROC and slow control systems inject data into the system that can be further processed or written to storage by filter or processing elements. A status and control pipeline provides access to the system for client machines.

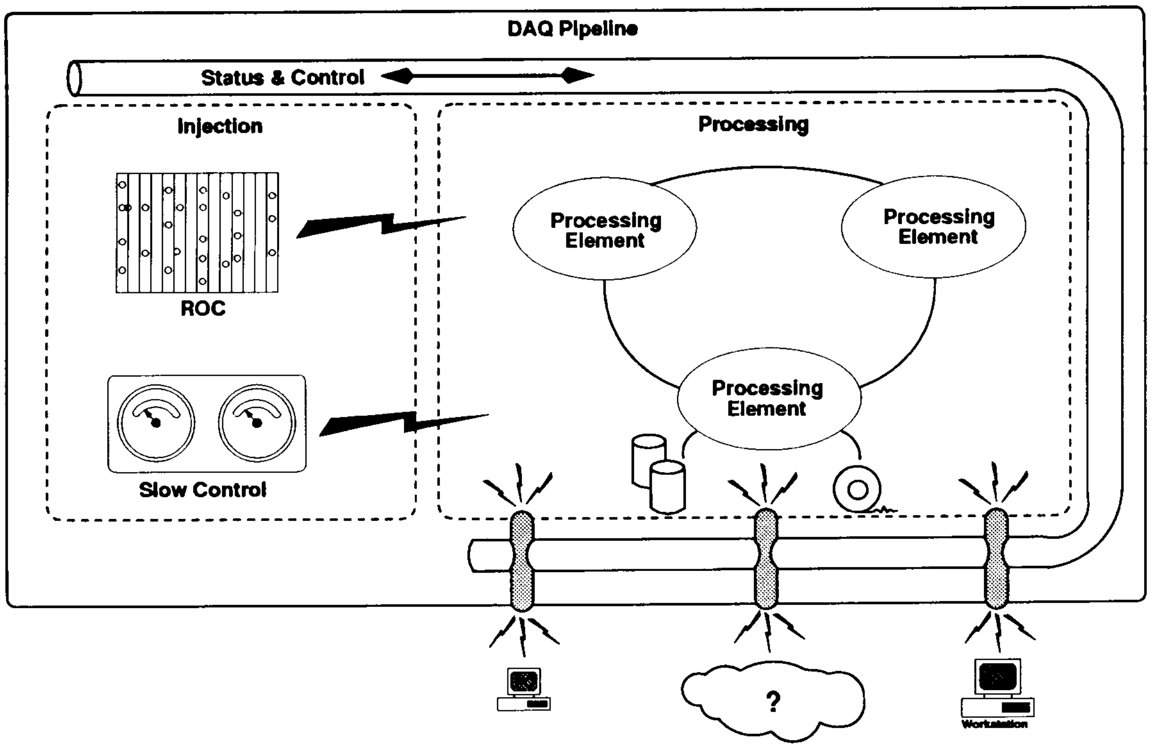

Figure 4: Basic DAQ System Diagram

The system is being coded in C++. It is anticipated the the object oriented nature of this language can be used to create abstractions, such as buffers and pointers, that can be learned and used easily while hiding the more complex system implementation from the user.

5. Conclusion

In this paper a discussion of hardware, networking and software issues and explorations, and how they impact the preliminary design and implementation of the next generation DAQ system at the NSCL has been presented. A summary of the explorations and experiments conducted that have helped guide this initial design was provided. It is anticipated that this design will undergo further revisions as implementation progresses.

References

1. D. Abbott and P. Banta and J. Chen and G. Heyes and E. Jastrzembski and C. Timmer, Implementation and Performance of the CLAS Spectrometer On-line System using CODA Version 2, Proceedings 1997 Xth IEEE Real Time Conference, Sept. 1997, Beaune, France,

2. C. Moore, The Fermilab DART Data Acquisition System at Running Experiments, Proceedings 1997 Xth IEEE Real Time Conference, Sept. 1997, Beaune, France, pages 359--362,
3. M. Rochner and W. Erven and P. Wustner and K. Zwoll, The Second Generation of DAQ-Systems at COSY, Proceedings 1997 Xth IEEE Real Time Conference, Sept. 1997, Beaune, France, pages 241--247,
4. Y. Tanaka and M. Haseno and Y. Nagasaka and Y. Sakamoto and Y. Yasu and H. Fujii and K.H. Tanaka, Performance of CAMAC Data Acquisition System Under Linux, Proceedings 1997 Xth IEEE Real Time Conference, Sept. 1997, Beaune, France, pages 359--362,
5. C. de Laat and P.G. Kuijer and Peter Olthuis and W. Lourens, A Distributed Data Acquisition System By ATM Networks a Pilot Study, Proceedings of the 1995 IEEE Conference on Real-Time Computer Applications in Nuclear Particle and Plasma Physics, May, East Lansing, MI, pages 1--5,
6. M.Campanella and T. Ferrari and A. Ghiselli and C. Vistoli and C. Battista, TCP-UDP/IP Performance at High Speed Over ATM, Proceedings of the 2nd International Data Acquisition Workshop on Networked Data Acquisition Systems, Nov. 1996, Osaka, Japan, pages 3--6