FERMILAB-CONF-08-520-CD

A Software Solution for the Control, Acquisition, and Storage of CAPTAN Network Topologies

Ryan A. Rivera, Marcos Turqueti, Alan Prosser

***Abstract*–The Electronic Systems Engineering department of the Computing Division at the Fermi National Accelerator Laboratory is developing a data acquisition system that is flexible and powerful enough to meet the demands of pixel and strip detectors for high energy physics applications, but also capable of far broader application utility. To facilitate data acquisition and processing as well as system configuration and control, a stacked system was devised with support for gigabit Ethernet networking. The individual unit within the system is known as a Compact And Programmable daTa Acquisition Node, or CAPTAN. The platform's flexibility is achieved through the ability to stack specialized boards, both vertically and horizontally, to create an integrated system that can be optimized for each user and application. The CAPTAN's main interface is via the Universal Datagram Protocol of the Internet Protocol (UDP/IP). The software solution presented in this paper is at the other end of the UDP/IP interface, and must orchestrate communications. The software must take a modular approach to its command library to give the user a unique and extensible vocabulary with which to communicate with ever advancing readout chip technologies and varying configurations of the specialized boards within a CAPTAN. The software must also handle multiple CAPTANs, each producing gigabits of data per second, thus the solution presented here provides an option to employ distributed computing for CAPTAN network topologies involving large amounts of data.**

I. Introduction

PRESENTED in this paper are design decisions made to achieve a complete software solution for interfacing with a network of Compact And Programmable daTa Acquisition Nodes (CAPTANs) [1]. A network of CAPTANs could mean a single CAPTAN working in a test stand system, or it could mean a farm of CAPTANs handling the controls and data acquisition for an entire experiment.

There are several requirements necessary for a CAPTAN software solution to be complete. First, a complete software solution must be designed to adjust to the wide range of configurations within a CAPTAN stack. This means adapting to the inclusion or exclusion of specialized CAPTAN boards and to the future addition of new firmware and hardware.

Second, the software must facilitate changes to the user interface to configure, control, and interpret readout chips that are unknown a priori. Users need to have the ability to define their own unique interpretation of data within the system.

Finally, the system must be able to handle multiple CAPTANs, each producing gigabits of data per second. Data throughput of this magnitude cannot be managed by a single processor alone, so multi-process and distributed computing techniques must be employed.

The solution presented in this paper is a C++ application designed for Microsoft Windows. Note, however, that there are plans for the addition of a web-based interface that would make the system accessible from any platform.

II. The Components

The software solution can be broken into independent blocks that each serve a special function. Replicating a block, replicates its function. The role of each block is defined considering the smallest and largest possible CAPTAN network topologies.

It was decided to design the blocks as stand-alone applications giving complete freedom to their physical location as long as all processes share a common Ethernet network. This gives the user the flexibility to use more than one computer without additional software. Distributed computing is necessary to accommodate the potentially large amount of data in a CAPTAN network.

The four building blocks of the system are the Global Master, the CAPTAN Controller, the Storage Controller, and the Graphical User Interface. They will be referred to interchangeably throughout this paper as the GM, CC, SC, and GUI respectively.

A. The Global Master

The Global Master or GM is at the center of the CAPTAN software solution. The GM is the only one of the four functional components that can not be replicated within a CAPTAN network. All communication between the users and the rest of the system flows through the GM.

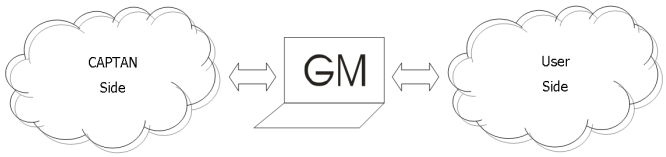

Fig. 1. This figure depicts an overview of the CAPTAN software system and highlights the role of the Global Master (GM) within the solution. The GM passes commands from the user on to the rest of the system, and in the other direction, it forwards data from the CAPTANs to the user for interpretation .

Manuscript received November 10, 2008. This work was supported by Fermi National Accelerator Laboratory operated by Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359 with the United States Department of Energy.

R. A. Rivera is with Fermi National Accelerator Laboratory, Batavia, IL 60510 USA (phone: 630-840-3844; e-mail: rrivera@fnal.gov).

M. Turqueti is with Fermi National Accelerator Laboratory, Batavia, IL 60510 USA (phone: 630-840-2244; e-mail: turqueti@fnal.gov).

A. Prosser is with Fermi National Accelerator Laboratory, Batavia, IL 60510 USA (phone: 630-840-6599; e-mail: aprosser@fnal.gov).

The majority of the GM's processing time is consumed by message forwarding, but the GM must also manage the roles of the system components and enforce system security. The GM is responsible for guaranteeing that there is only one GUI with master permissions connected to the system. The role of the master GUI will be further defined in Section II-D.

When the GM starts, it opens a socket to listen for incoming TCP/IP connection requests. TCP/IP connection requests arrive when new blocks attempt to join the system. When a TCP/IP connection request is received, the GM validates the request with a handshake and determines which type of block is attempting to connect – the type must either be a CAPTAN Controller, Storage Controller, or GUI. Once verified, a new socket is opened over which all future communication will take place between the newly joined block and the GM.

The GM maintains a list of all blocks that are connected to the system by storing the type of the process and the process' communication socket handle. Once connected, the new block remains in the GM's connection list until the process disconnects. There is theoretically no limit to the number of blocks that can join the system.

By maintaining the list of active blocks in the system, the GM has access to every application at all times. The GM constantly checks the open connections for packets arriving on the connected sockets. Upon reception of a packet, the packet is interpreted and handled appropriately.

Information in the GM's connection list may also be requested by users to understand which resources are available in the system and how many other users are connected.

B. The CAPTAN Controller

The CAPTAN Controller or CC resides between the GM and the CAPTAN. There are as many instances of the CC as there are CAPTANs in the system. The CC is in control of a single CAPTAN and some amount of permanent storage. It forwards commands from the GM to the CAPTAN, and it routes data from the CAPTAN either to disk, to a Storage Controller, or to the GM. The ultimate location of the CAPTAN's raw data is configurable by the user with master permissions. The CC is responsible for maintaining the list of files that it saves to disk, and must present the saved content list to the GM when prompted.

Of all the blocks, the CC encounters the highest data throughput because the raw data from the CAPTAN's gigabit Ethernet link runs directly to a single socket controlled by the CC. To best utilize processing power in handling this potentially large volume of data, the CC creates two threads and allocates a block of shared memory. One thread is dedicated to receiving data from the CAPTAN and storing it directly to the shared memory. The other thread is responsible for handling communication between the GM and forwarding commands to the CAPTAN. Additionally, if the CC is configured to store data to disk, a third thread is created with the sole role of streaming data from shared memory to permanent file on disk. Shared memory and mutual exclusion objects are utilized for thread synchronization.

The three threads are necessary because when encountering maximum throughput (~700 Mbps), the CC consumes most of a processor's resources receiving data, and yet the CC must still be responsive to configuration and other requests from the rest of the system. By splitting the tasks into multiple threads, the operating system can schedule multi-processor systems to optimize efficiency as detailed further in Section III-B.

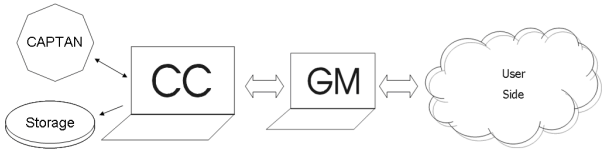

Fig. 2. Above, the CAPTAN side of the system overview is filled in. The CAPTAN Controller (CC) is shown as the go-between for the Global Master (GM) and a CAPTAN. The CAPTAN Controller also has access to disk space which may be local or remote.

C. The Storage Controller

The Storage Controller or SC connects to a CC to allow the CC to save data on a remote machine. An SC can connect to only one CC, but a CC can connect to multiple SCs.

Because the SC is on a different computer than the CC, the user has the flexibility to place the SC in a different physical location – maybe outside the lab environment of the CAPTAN – or on a machine with extensive disk space.

The master GUI user is responsible for configuring a CC such that it forwards data to an SC. The SC is responsible for declaring how much free space it has available when prompted by the CC. When multiple SCs are connected to a single CC, the CC splits the bandwidth among its SCs by evenly distributing data packets.

D. The GUI

The GUI is the user access point to the CAPTAN network. It is a C++ Windows application, but could become a platform-independent web application in the future. Through the GUI, the user has access to all of the stored CAPTAN data files that are sitting in permanent storage. The user has access to real-time data to gauge the status of detectors. The user can formulate commands and choose where to direct them.

There can be multiple GUIs, but in order to avoid conflict within the system, only one of the GUIs can have master permissions. This is enforced by the Global Master. The GUI with master permissions essentially has read and write access, while the other GUIs are restricted to read-only access.

The master GUI has full access rights to configure every CC and SC in the system and to send commands to every CAPTAN in the system. The Master GUI can decide whether or not the other GUIs will receive system data.

Fig. 3. Above, the GUI is placed in the system overview diagram. The GUI is the access point for the users to the control and acquisition features of the system.

III. Design Decisions

In this section some of the unique design decisions for this software project will be discussed along with their motivations.

A. The Modular Command Library

The solution for the user interface must facilitate adaptation to new readout chip technologies, assorted CAPTAN stack configurations, and varying CAPTAN network topologies. The approach that was taken was to utilize Visual Studio's tab class. This approach allows users to develop their own dialog boxes independently of the other tabs in the system. The user can create personalized tabs that, for instance, may contain specialized commands or uniquely interpret data. The user only needs a basic understanding of dialog box creation in Visual Studio, and knowledge of the hooks to the underlying functionality of the GUI software.

Once a new dialog box is created it can be instantiated as a tab in the GUI interface. The tabs can be dynamically added or removed through a configuration file.

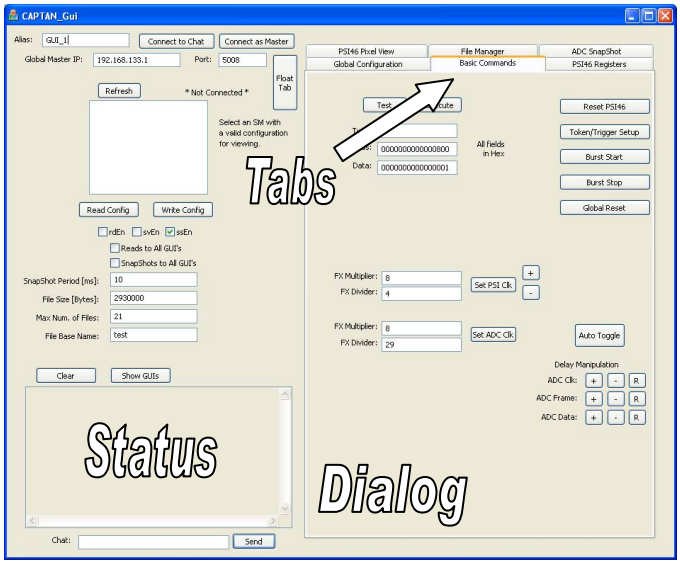

Fig. 4. This is a screenshot of the GUI. Highlighted are the tab system, the dialog area, and the status area. The dialog area is where standardized and user dialogs appear once their corresponding tab is selected. The status area is where chat and system messages are displayed, and where basic system configuration can be managed.

B. Multiple Processes and Multiple Threads

The specialized blocks are implemented as individual processes which divide the total computation needed for the system into manageable pieces. Beyond that, these processes are further divided into smaller tasks that are handled by threads. This multi-process and multi-thread approach exploits the processing power of multi-core PCs.

When a single-threaded process is run, the operating system is left without an option – the process must be executed sequentially by a single processor. It makes no difference if the system has additional computing resources available, only one processor can be employed.

However, if that same single-threaded process were segmented into threads or processes that could work together in parallel, then the operating system could optimize their execution by utilizing more of the system's resources. This is the principle that the CAPTAN software solution attempts to exploit.

C. Memory Mapped File Buffering

Writing small fragments of data directly to disk is too slow to keep up with the gigabits of data per second that a CAPTAN may generate. To resolve this issue, the CAPTAN software solution uses a memory map technique with double buffering. In such a method, two files on disk are initially mapped to the memory space of the process. The process proceeds to begin writing to the first file's area in the memory map. These writes are relatively quick because they are writes to memory within the process' address space.

When the end of the first memory mapped file is reached, a monitoring thread recognizes that the file is full, flushes the file to disk, and maps a new file to the same location in the application's memory space. Concurrently, the storage thread begins writing to the portion of the address space which is mapped to the second file. By the time the second mapped file is full and ready for closeout, the first area is already mapped to a new file on disk and is prepared for the cycle to repeat itself.

This approach maximizes the fragment sizes of writes to disk, which optimizes throughput.

D. Data Forwarding to Storage Controllers

The feature to forward data from the CAPTAN Controllers to Storage Controllers was implemented to enable users to have central storage locations, or to utilize certain locations with vast memories. With such capability, users are not restricted to store their data to machines that are in the lab environment – physically connected to the CAPTANs. Users have the flexibility to divert their data to any machine on the main Ethernet network.

IV. CAPTAN Network Topologies

The CAPTAN's Ethernet interface makes it ideal for networking. With a few CAPTANs, some off-the-shelf components, and the proper software solution the user has a powerful unified system. The goal for this software solution was to achieve enough flexibility to seamlessly function with one or one hundred CAPTANs and PCs in a single network.

A. The Test Stand

The simplest topology is that of a single CAPTAN and single computer, as shown in Fig. 5. This is anticipated to be a common test stand and development configuration. In this configuration, all of the software components (the GM, a CC, and a GUI) must be running on the PC.

For the test stand case, a PC with a single network interface card is sufficient. A standard Ethernet cable connects the CAPTAN to the PC.

Fig. 5. Above is an example simple test stand network topology. A single CAPTAN (C) and a single personal computer (PC) make up this network and are linked by a single Ethernet connection.

B. Multiple CAPTANs, Single Computer

The complexity rises when more than one CAPTAN are in the system. If a single PC can handle the data rate of multiple CAPTANs, then a topology as shown in Fig. 6 could be implemented. On the other hand, if the expected data throughput would overwhelm the single computer, then multiple computers could be used as discussed in Section IV-C. Either of these two configurations might be employed for a production run of a detector, for example, in order to conduct tests and calibrations in parallel.

With a multiple CAPTAN, single computer topology, the user must instantiate on the computer a CC for each CAPTAN connected to it. Additionally, the GM and a GUI must be executed on the machine.

The computer in this topology only needs one network interface card if there is a gigabit Ethernet switch or router also present. If a switch or router is not present, then to use one computer, the computer would need a network card for each CAPTAN.

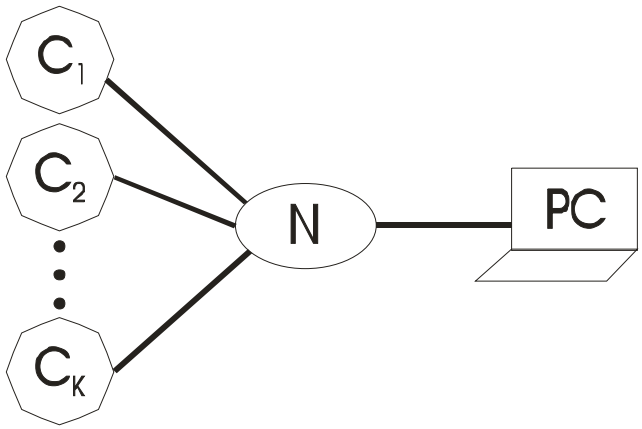

Fig. 6. Above is an example CAPTAN network topology with multiple CAPTANs and one computer. The N in this figure represents a network switch or router. This topology could be applied to relatively low data throughput applications.

C. Multiple CAPTANs, Multiple Computers

With a multiple CAPTAN, multiple computer topology, computing resources are available to handle high data throughput applications. The expandability afforded to the user is highlighted in Fig. 7 and Fig. 8.

In multiple computer topologies, the machines where the CCs are executed need two network interface cards to maintain full throughput capacity for their CAPTANs.

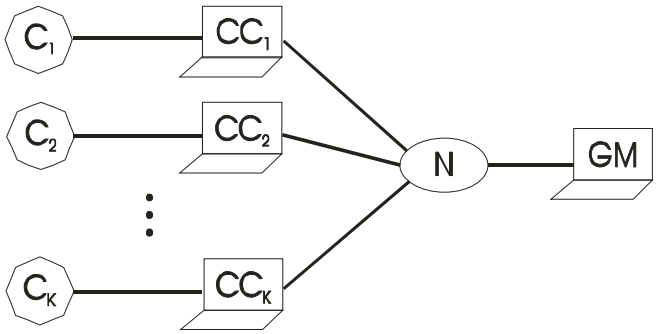

Fig. 7. Above is a topology that could be used for high data rates. This network has one coordinating Global Master (GM) and one CAPTAN Controller (CC) for each CAPTAN Ethernet interface.

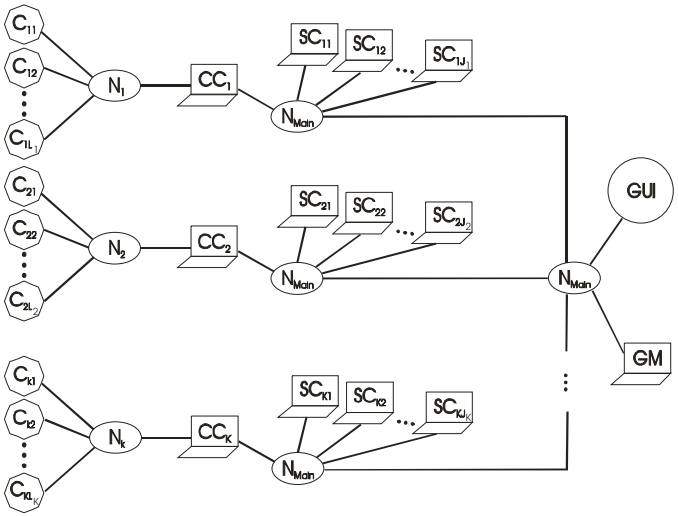

Fig. 8. This figure represents the wide range of CAPTAN network topologies supported by the CAPTAN software solution. Note that on the CC machines, there must be a CC instance for every connected CAPTAN.

V. Conclusion

The software system is novel in that it gives the user the option of maintaining the same controls, acquisition, and analysis tools from a single unit test stand to final system production.

The system scales by taking advantage of multi-process and distributed computing techniques, and by exploiting design decisions which allow for the replication of function simply by executing multiple instances of standard processing blocks. The CAPTAN software solution has capacity for yet unknown devices and permutations of specialized boards by utilizing a modular graphical interface. This software solution for CAPTAN network topologies achieves a balance of flexibility and functionality.

References

[1] M. Turqueti, R. A. Rivera, A. Prosser, J. Andresen, J. Chramowicz, S. Kwan, "CAPTAN: A Hardware Architecture for Integrated Data Acquisition, Control, and Analysis for Detector Development (Periodical style—Accepted for publication)," *2008 IEEE Proc. Nuclear Science Symposium*, to be published.