IEEE Transactions on Nuclear Science, Vol. NS-30, No. 4, August 1983

DATA ACQUISITION SOFTWARE FOR THE LAMPF CONTROL SYSTEM*

Stuart C. Schaller and Patricia A. Rose
Los Alamos National Laboratory
Box 1663, MS-H810
Los Alamos, NM 87545

Abstract

As part of the upgrade of the Los Alamos Meson Physics Facility (LAMPF) control computer, the software providing the interface between application programs and the accelerator hardware was redesigned. This new Data System design - which handles more than 12,000 widely different devices - allows greater uniformity, flexibility, maintainability, and hardware independence than before. The Data System portion of the LAMPF Control System is described in this paper.

Introduction

The original LAMPF Control System was written for a Systems Engineering Laboratory (SEL) 840 computer. Locally designed Remote Instrumentation and Control Equipment (RICE) hardware provided access to accelerator data and controls. Eventually, additional accelerator data was made available through CAMAC interfaces addressed by remote PDP-11 computers. The remote PDP-11s were linked to the SEL-840 via locally designed CAMAC data link modules.

Several years ago, a decision was made to replace the SEL-840 with a VAX-11/780. At that time we recognized that there were problems with the way many application programs on the SEL-840 dealt with accelerator devices. Often programs contained embedded hardware addresses for accelerator devices. This made the programs difficult to maintain when wiring changes were made. It also made the code difficult to read and update since the physical meaning of the data was obscure. In addition, some application programs contained explicit message formatting software.

A symbolic device-naming scheme using ten-character "operator designators" was designed when the RICE system was first implemented. However, this naming convention was not extended to include remote PDP-11 CAMAC devices. This fact forced the use of hardware addresses in programs dealing with CAMAC, and made it almost impossible for the accelerator operators to reference CAMAC devices directly.

One of the goals of the VAX upgrade was to introduce a Data System interface to make application programs as hardware independent as possible and to allow more flexible device implementation and improvement. The next two sections of this paper outline the VAX based LAMPF Control System context, and describe the new Data System. Information about other aspects of the VAX upgrade may be found in [1] and [2].

LAMPF Control System Context

Figure 1 shows the data acquisition and control hardware which makes up the new LAMPF Control System. The VAX-11/780 - used as the central control computer - is at the hub of a star network. All application programs which communicate with the accelerator operators run on this VAX.

The RIU-11 is a PDP-11/34 computer used to dual port the RICE system to the VAX and the SEL computers during the upgrade process. The RIU-11 accepts messages from the control computer and causes RICE data takes and commands to be issued through the RIU (RICE Interface Unit). The RIU-11 also receives timing signals and gate information from the LAMPF Master Timer (MT). These signals allow the RIU-11 to make timed and untimed RICE data take requests with specific timing gate configurations

The NET-11 is a PDP-11/34 which is used to dual port the remote PDP-11s during the upgrade. It switches messages between the control computer and the remote PDP-11s. The remote PDP-11s contain software which allows the control computer to directly address CAMAC devices in the associated CAMAC crates. In addition, the remotes also contain software for extremely time-dependent data acquisition tasks (such as acquiring beam profiles).

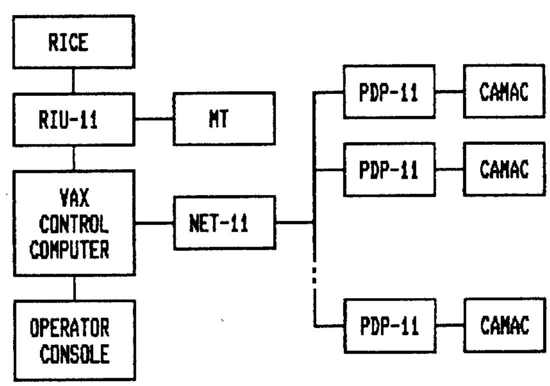

Figure 1 - LAMPF Control System Context

Data System Design

At the time the VAX upgrade was started, the intention was to convert and/or rewrite the application software which existed on the SEL-840 control computer. We did not have the resources to rewrite or heavily modify the software running on the remote PDP-11s, so the goal of making application programs hardware independent was limited to those applications which ran on the control computer. (We are currently studying the problem of making the applications programs in the

*Work performed under the auspices of US Department of Energy

0018-9499/83/0800-2308$01.00© 1983 IEEE

remote PDP-11s hardware independent.)

We attacked the problem of hardware independence by re-emphasizing the symbolic device naming that had been developed for the RICE hardware devices. We insisted that all references to accelerator devices by application programs must be by operator designator. We also attempted to make the operator designators reflect the actual physics of the accelerator itself as opposed to reflecting accidents of hardware address assignments. We hoped this would make the application programs easier to understand. (The only exceptions to the operator designator rule were hardware level diagnostic programs.)

Device Tables

The Device Tables define the Data System devices available to the application programs and provide the mapping from operator designators to hardware information. A Device Table Entry contains two kinds of information about a device. The first kind is device characteristic information including the domain (digital or analog), the dimension (scalar or vector), allowable requests, the equipment type, and the hardware address.

The equipment type can be: RICE - for devices in the RICE system; CAMAC - for directly controllable CAMAC modules; Remote - for devices defined by software in the remote PDP-11s; or Pseudo - for devices defined by software in the VAX control computer. Pseudo devices allow us to define new or experimental devices in software. The defining software may make recursive Data System references to other devices defined in the Device Tables. Pseudo devices give us great flexibility in defining very high-level devices. For instance, the centroid of a beam profile could be defined as a pseudo device whose defining software reads an entire profile and computes the centroid.

The second kind of Device Table Entry information describes the default parameters to be used for read or command requests. If a program requests a read on a device and specifies no additional parameters, the read request will be defined by the defaults contained in the associated Device Table Entry. Default request information includes: a delay gate; timing type - timed/non-RF/anytime; time relative to a specific edge of a specific Master Timer gate; and required beam pulse gate configurations. For example, the default parameters for a beam current monitor might specify timed data taken 500 micro-seconds before the end of the H+ beam gate on a beam pulse with H+ beam gate and an RF gate present.

More information on the design of the Data System Device Tables may be found in [2].

Application Program Interface

Application programs use the Data System in much the same way that they interface with the operating system's file structure. Operator designators must be "opened" by the application in order to establish a connection with the appropriate Device Table Entry. Eventually, the application program - or the Data System itself on application program termination - must "close" the devices that have been opened.

After the device has been opened, the application program may access or override default request information in the Device Table Entry before requesting a read or command on the device. In addition, the application program may specify information for averaging data, error recovery, synchronous or asynchronous operation, engineering unit conversion, and increment, setpoint, or limit commands.

Data System Request Handling

Figure 2 shows the Data System request handling software. In the outline below, each word in brackets corresponds to a box on the diagram. A typical request (read or command) on a device progresses as follows:

1. [INITIAL] The request made by the application program is validated, and a Data Request Packet (DRP) is allocated from a system-wide pool. The DRP contains complete information on the current state of the request. The initial state is determined by the Device Table Entry [DEVTAB] associated with the device and by a parameter array specified by the application program.
2. [NEXT] Based on the setting of program-specific, device-specific, or system-wide flags, the request may be logged. The setting of these flags also determines if the request is sent to the Imaginary Meson Physics Facility [IMPF] simulator. If it is not simulated, a message transfer is started between the VAX and the RIU-11 or NET-11 computers, or the request is sent to the associated pseudo device handler [PSEUDO]. Data take timing values are resolved by interrogating the current state of the Master Timer [MT].
3. [COMPLETE] When the action is completed, error returns are examined, and - based on the current DRP state - the action may be repeated. If the action has been successful, further actions may be performed to acquire average data or to complete command requests. If the request is complete, then data, status, and completion notification are sent to the application program.

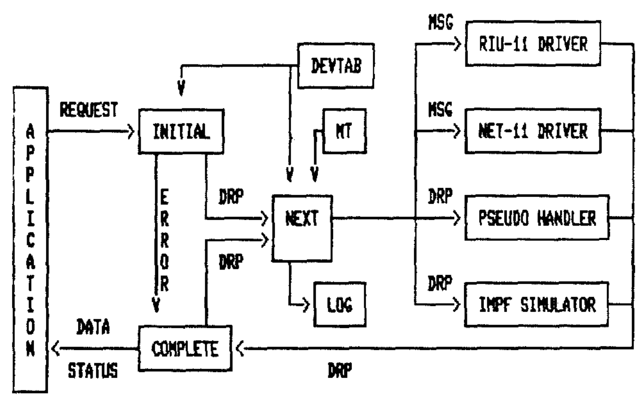

Figure 2 - Data System Request Handling

Implementation Concerns

The parallel with operating system file structures is even more pronounced at the Data System implementation level. Design decisions included the desires: to have global access to all Data System data structures for diagnostic purposes; to prohibit applications from using device hardware addresses; and to separate the Device Table data base manager from the Data System request handling software.

Global access required: protection against accidental user writes; synchronization of competing programs using common data structures; and cleanup of program-specific data structures at program termination.

The prohibition of hardware address use was achieved by requiring programs to have a special privilege before being allowed to talk to the Data System device drivers. This privilege is granted by privileged Data System software to requests which have been made via operator designators.

The separation between Device Table maintenance and Data System request handling was accomplished by an "object level" Device Table structure. The data base manager maintains a symbolic "source level" data base containing the device descriptions. This data base is translated into an object level Device Table description which is used by the Data System request handling software.

Conclusions

We have been able to take data and control RICE devices through the new Data System. Several application programs have been written including an accelerator summary status display, an analog slew program, and a program which produces a single line display of the state of any device in the Device Tables. We passed a milestone in late January when the summary status program was used to turn the LAMPF H+ beam on and off.

We feel we have achieved our design goals of flexibility and hardware independence. We have much left to implement in the way of new devices and expanded simulation and diagnostic capabilities, but our experience indicates that these extensions will fit in well with our design.

Since application programs are independent of the hardware, and indeed of the actual algorithms used in the Data System software, optimization can occur at either the hardware or software levels with no detrimental impact on existing applications. We expect that we will make optimizations when areas needing them become apparent as more application programs are written for the new control computer.

Given basic system correctness, we would rather work on making a flexible system speedy than making a speedy system flexible.

Acknowledgments

We would like to thank Eric Bjorklund, Jamii Corley, Phyllis Wallis, and Stan Brown for their contributions to the Data System design and implementation.

References

[1] Schultz, David E. and Brown, Stanley K., "On-Line Replacement of a Particle Accelerator Control Computer," Proc. IEEE Real-Time Symposium, Dec. 1981, pp.78-82.

[2] Brown, Stanley K., "The Use of a Commercial Data Base Management System in the LAMPF Control System," these proceedings.