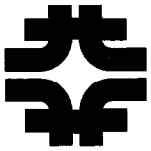

Fermilab

TM-1244
2311.000

THE EPICS SYSTEM: AN OVERVIEW*

J. F. Bartlett, J. S. Bobbitt, B. J. Kramper,
T. E. Lahey, B. A. MacKinnon, and R. E. West

February 1984

*Submitted for publication in the Proceedings of the Digital Equipment Computer User Society, Las Vegas, NV, October 1983.

THE EPICS SYSTEM: AN OVERVIEW

J. Frederick Bartlett
J. S. Bobbitt, B. J. Kramper,
T. E. Lahey, B. A. MacKinnon, R. E. West
Fermi National Accelerator Labo tory
Batavia, Illinois

ABSTRACT

This paper presents an overview of the EPICS control system at FERMILAB. EPICS is a distributed, multi-user, interactive system for the control and monitoring of particle beamlines at a high-energy experimental physics laboratory. The overview discusses the operating environment of the control system, the requirements which determined the design decisions, the hardware and software configurations, and plans for the future growth and enhancement of the present system.

This paper is the first of three related papers on the EPICS system. The other two cover (1) the system structure and user interface and (2) RSX implementation issues.

INTRODUCTION

EPICS (Experimental Physics Interactive Control System) is a computer-based process control system at Fermilab for the control and monitoring of the extracted particle beam from a high-energy proton synchrotron. This paper, which is the first of three related papers on the EPICS system, presents an overview of the control system and provides the necessary background for the other papers which cover (1) the system structure and user interface and (2) RSX implementation issues.

The body of this paper is in several sections. The first describes the nature of the external environment in which the system must operate. This is followed by sections which discuss the system requirements and the hardware and software configurations. Finally, there is a section on future plans and directions.

ENVIRONMENT OF A PROTON SYNCHROTRON LABORATORY

The Synchrotron and Experimental Areas

The characteristics of a high-energy proton synchrotron impose unique conditions upon a process control system. The operation of the Fermilab synchrotron is cyclical with periods which can range from approximately ten seconds to several minutes. Once the proton beam reaches the desired energy, it is extracted from the accelerator and delivered to numerous experiments. The peak communication and computation load on the control system occurs during the extraction period.

The duration of the extracted proton beam can be as short as a fraction of a second or as long as tens of seconds. When the proton beam is extracted it must be transported over distances of several kilometers by a combination of magnetic and electrostatic devices which change its direction, alter its cross-sectional profile, and select a range of particle momenta. The original proton beam is first split into separate beams for delivery to one of three major experimental areas where these beams may in turn be further divided to satisfy the requirements of the experiments which run concurrently. Some of the primary proton beams are directed into targets from which particles of other types are produced and these secondary particle beams are manipulated by similar devices.

The profile, composition, and intensity of the particle beams must be measured and many of the utility support systems which supply cooling water, electrical power, maintain vacuum in the transport pipes, and other similar functions must be monitored for proper function.

Devices

There are between one and two thousand distinct devices in each experimental area which are interfaced to the control computers with branched serial CAMAC communication links having a total cable length greater than fifteen kilometers. It is this highly-branched collection of controlling, measuring, and monitoring devices, organized into multiple beamlines, which are the direct objects of concern to the EPICS system.

Examples of devices which are accessed by the control system are:

(1) current and voltage supplies which energize the bending (dipole) and focussing (quadrupole) magnets
(2) stepping-motor controllers which position targets, collimators, and beam stops

(3) hydraulic and pneumatic controllers which move measuring devices in and out of the beam
(4) terminal interfaces which are the ultimate sources of control and destinations of collected information
(5) multiplexed analog-to-digital converters which transform the sensor outputs
(6) digital interfaces which input status and output control functions

The Beamline Communication Link

Communication to these widely dispersed control elements is provided by a serial CAMAC (Computer Automated Measurement And Control) link. CAMAC is an international interface standard which is widely used in the physics community and has recently become popular in the industrial control field.

A CAMAC system is organized as a collection of powered "crates", each of which provides a bus, the DATAWAY, whose characteristics have been specified by the standard. The crate provides mechanical support, cooling, and power for multiple modules which connect to the DATAWAY. The DATAWAY is managed by a crate controller, which itself is a CAMAC module occupying two or more of the available module positions or "slots" in the crate. Other modules, which also connect to the DATAWAY, occupy the remaining slots in the crate, and they generate and accept the various process control signals from the external environment. Since CAMAC is a widely used standard, some of the modules have been obtained from commercial sources. Most, however, are specifically designed to match the beamline control requirements.

Although there is now a standard for a serial connection between CAMAC crates, there was none at the time that the beamlines were originally constructed. The serial communication link which connects the beamline crates was the prototype for what became the CAMAC serial protocol standard. As such, it differs from the standard in several significant features. The standard uses a single cable loop for the bit-serial message with each crate controller passing the serial message to the next crate until it is returned to the serial link controller.

The Fermilab link is branched rather than looped with the crate controller which is addressed by the message returning the modified message on a second cable. This has the advantage of being more robust than a loop which is disrupted by a break anywhere in the loop; however, the standard also allows unsolicited "demand" messages to be generated by a crate controller. The branched structure complicates the transmission of unsolicited messages; and, consequently, the modules in branched system must be polled to detect changes in the external environment.

A further feature of the Fermilab system, which is absent from the standard, is a separate block-transfer capability using an additional pair of cables. Whereas the normal, single-message lines, which operate at a one megahertz bit rate, require seventy microseconds to transmit a message with a 24-bit data word; the block transfer facility can transmit a block of 16-bit words at a rate of one word every thirty microseconds.

SYSTEM REQUIREMENTS

EPICS was initially conceived as a replacement for an existing control system. As such, it was required to provide all the functionality of that system while incorporating contemporary computer hardware and good software design practices. Additionally, it was expected that many of the resource limitations, such as the number of devices in the data base, would be significantly extended.

The Command Language

One of the most important requirements placed upon the EPICS system was that the command language, through which the user must formulate all requests, must be a complete programming language. It must be possible to construct programs of considerable complexity with access by name to the "controlled objects" of the external environment. The principle motivation for this was the observation that it is more efficient to provide someone who understands the problem with a language in which to construct the solution than it is to explain the nature of the problem to a programming specialist. This leaves the system programmer free to concentrate on providing the tools with which to construct the solution.

Following a survey of existing accelerator control systems, the BASIC language was selected as the foundation upon which to build a control dialect, CBASIC, for the following reasons. First, the BASIC language was widely used and was familiar to most of the users of the control system. It is fundamentally interactive, a strong feature in itself. Additionally, it contains its own line-oriented editor which performs simple syntax analysis and error reporting when a statement is entered. Finally, there were several existing BASIC language processors which could be expanded to meet the special needs of a beamline control system, thereby avoiding the considerable effort of constructing the major elements of a syntax analyzer and an interpreter.

Resource Management

Since the operation of the Fermilab synchrotron is cyclical with periods which can range from approximately ten seconds to several minutes, the communication and computational load upon the control system follows a similar pattern with peaks of activity during certain phases of the acceleration cycle. Those operations which are not time-critical should be executed when the system is otherwise lightly loaded; and, where the system cannot adjust automatically, operations personnel must be able to redistribute the load to achieve a balance.

Although a main operations center exists, the majority of the user consoles are distributed throughout the experimental areas and are interfaced to the system with CAMAC modules. The CAMAC link does not support unsolicited messages; therefore, consoles must be polled. This represents a constant, non-negligible load on the system, and consoles which become inactive should be polled at a much lower rate, returning to the normal rate when a new character is typed. The operators must be able to add or delete consoles from the polled list without disrupting other system activities.

There are several classes of users of the system. Some, like the operators and managers, require global privileges while others, like the experimenters, must be protected from mutual interference. This requires a dynamic system of privileges and protection for devices, files, and system resources. Protection in general and device protection in particular is a matter of preventing accidental rather than intentional disruption.

Classic Process Control Issues

There are several design decisions which are common to most process control systems. These "classic" issues are often fundamental in determining the system efficiency.

The first and, perhaps, the most important is the question of whether data acquisition from devices should be in response to an explicit demand or whether a data pool of device readings should be maintained with requests for new data being satisfied from the shared pool. The demand method assures that a device access is made only when there is a request for the device at the cost of redundant accesses when there are several requests for the same device. Data pooling, on the other hand, can burden the communication link with unnecessary device accesses. EPICS maintains a system data structure for each device being accessed which contains its characteristics; and, when a new device request matches all of the characteristics of an existing request, the datum is shared. This provides the advantages of both methods at the cost of an additional data structure and the computational overhead required to manage it.

When data is pooled, there is the possibility that the current value in the pool is no longer valid. This "stale" data can cause severe problems in feed-back compensation loops which are closed through the computer. The EPICS system uses a time-stamp which contains a beam cycle number and a time relative to that cycle. Any application or system task may select receive the time-stamp with an input operation or check the time-stamp at the completion of an output operation.

Another issue is that of setup versus execution overhead. Systems in which most data acquisition processes are repetitive rather than single-execution gain computational efficiency by the use of special data structures which drive the data acquisition process. The cost is the space and the computational overhead of maintaining these structures. In EPICS, we chose to emphasize execution efficiency since most data acquisition processes are repetitive.

Resource recovery is essential for dynamic systems. This can be automatic, with the system maintaining internal links which guarantee recovery when processes terminate and internal timers which release resources when a process has failed to complete or to be terminated within a required time limit. In addition to these, EPICS checks certain critical resource limits before permitting a new data acquisition process to be initiated. This avoids a type of deadlock in which, once a resource falls below some limit, recovery becomes impossible because that same resource is required for the release. The RSX system pool is a familiar example of this class of resource deadlock.

Contemporary process control systems use databases, either central or distributed, to describe the collection of devices upon which they operate. When concurrent access and modification of the database is permitted, considerable care must be exercised to avoid conflicts. The EPICS system allows such concurrency, which is further complicated by the existence of a master database on a storage device and an in-memory copy of a database record of any device which was recently referenced by a data acquisition process. When a database record is modified, any existing data acquisition process continues to use the old definition while any new process uses the new definition. Since this can result in multiple copies of a record in the in-memory database, the old definition is removed when the last reference to it is deleted.

Finally, there is the question of the user's view of devices. At one extreme is the logical view where the context in which the named device is used determines the property of the device which is accessed. This view is typical of an experimenter who is primarily concerned with setting some device, like a beam stop, to a particular position or reading the current position. At the other extreme is the physical view where the user must be intimately aware of the details of the device and how it is interfaced to the control system. This represents the view of the maintenance technician who must be aware of the nature of the CAMAC module which interfaces a device to the system when a malfunction is diagnosed. In addition to both of these views, EPICS supports an intermediate view in which a named device may have a set of properties, which correspond to operations that may be performed upon the device, and may have multiple channels, which correspond to multiple instances of a function within a single, physical module. Only the physical view bypasses the database and its access protection mechanism; and, for this reason, the physical view is restricted to a selected class of users.

HARDWARE CONFIGURATION

The overall configuration for the EPICS system is shown in Figure 1. There are three identical systems which service the three major experimental areas.

Each system consists of four distinct processor levels. There is provision later for a fifth processor, the Level-1 computer, which is accessible from each of the Level-2 computers and whose resources are shared. The selection of a hierarchy of computing elements rather than a single computer of higher performance was dictated by several factors. First, the cost of equivalent performance, even when it can be supplied by a single CPU, is often more expensive than a hierarchy of smaller computers operating in parallel. Also, the individual units can be specialized, with hardware and software, to perform some compatible or closely linked set of tasks more efficiently than a single larger computer.

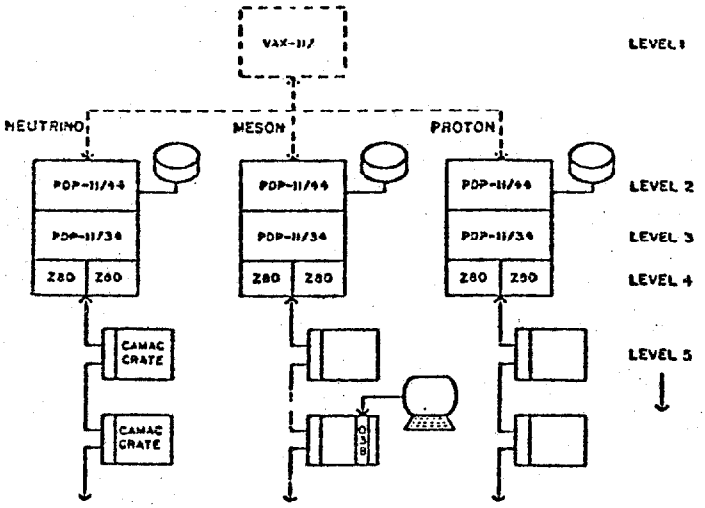

Fig. 1 EPICS System Configuration

At the lowest level (Level-5), dedicated-function microprocessors in individual CAMAC modules are distributed throughout the CAMAC crates. These elements act as data-acquisition concentrators, graphics display controllers, timing-event generators, and other similar functions. In brief, they provide a local, dedicated processor to perform those activities which require millisecond and sub-millisecond responses or which, were they to be executed by one of the higher-level computers, would degrade the performance of the communication link with large quantities of transferred data.

The next level (Level-4) consists of microprocessors which are directly connected to the lowest level general-purpose computer. They act as specialized device controllers which directly interface the higher levels of the control hierarchy to the external environment. In general, they perform operations which are relatively fixed, highly repetitive, and require a minimum-time response. There are presently two such controllers which provide (1) the ability to process priority-ordered, linked lists of CAMAC single or block-transfer CAMAC commands and (2) accelerator-based timing events.

Levels 2 and 3 are general-purpose, minicomputers. The Level-3 computer, a PDP-11/34A, operates in a polled environment and serves primarily as a data-acquisition processor for the next higher level. The Level-2 computer, a PDP-11/44, interprets and executes the command language, manages the device database, and provides most of the user utility functions.

A future, Level 1 computer would be shared by the three experimental areas and would provide beam diagnostic functions, high-level graphics, extended data-logging, and general, time-sharing services to the experimental areas.

Microprocessors

With a single exception, all of the microprocessor-based computing elements use the Z-80 computer. The specialized device controllers use a commercially developed, single-board Z-80 computer, the UMC-Z80 from Associated Computer Consultants, which is directly interfaced to the UNIBUS of a PDP-11 computer. Each of these is coupled with an additional board which contains the circuits which are specific to the processor's function. In the case of the serial CAMAC controller, the special-function board synchronizes the transmissions over serial CAMAC, performs error detection, and buffering of multiple commands. The adjunct board to the timer controller contains circuits to decode the accelerator clock, queue input timing and abnormal events, and queue completed events to the the PDP-11 supervisor at the next level.

The distributed CAMAC modules containing a microprocessor computing element are each of special design and may in some cases operate with some degree of local autonomy while in other cases they execute fixed functions and employ microprocessors only to reduce circuit complexity.

General-Purpose Computers

The general-purpose computers in the beamline control system are standardized on a single computer series, the DEC PDP-11, whose members share a common architecture. Figure 2 provides a more detailed view of the two general-purpose computers.

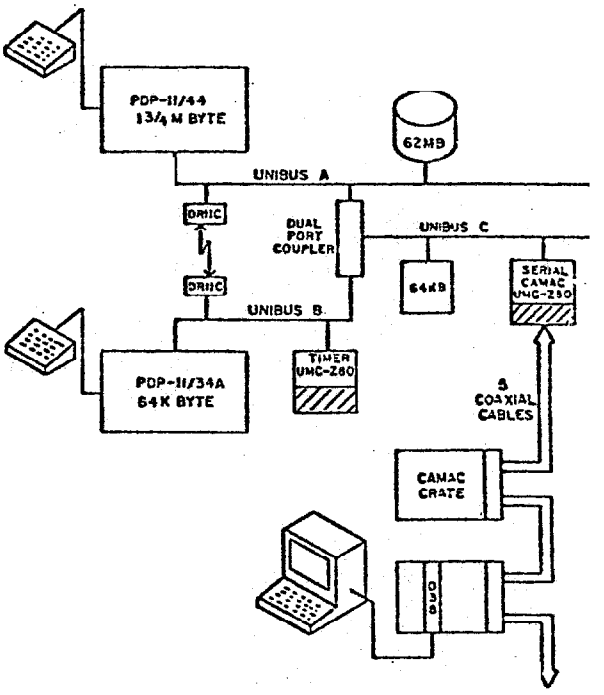

Fig. 2 EPICS Bus Configuration

The PDP-11/44, in addition to the usual UNIBUS peripherals, is provided with a communication path to the PDP-11/34A through a bank of shared memory. Messages or shared data structures are loaded into a partition of the shared memory and a pointer to the structure is passed through a full-duplex, interrupt-driven channel composed of two connected DR11C modules.

Sixty-four kilobytes of memory are located on a shared section of UNIBUS which is controlled by a

dual-port coupler attached to the primary UNIBUS of both computers. The microprocessor which manages the serial CAMAC link also resides on the shared bus where DMA transmissions into and out of the shared memory will not degrade the performance of the two primary buses.

The timer microprocessor, which communicates only with the data acquisition computer, is located on the primary UNIBUS of the PDP-11/34A.

SOFTWARE CONFIGURATION

The EPICS system was developed on and runs under the RSX family of operating systems. We attempted to minimize the number of languages used in the implementation of the software and, wherever possible, to make use of existing, fully-supported utilities and products for the RSX system.

The Level-2 computer uses an essentially unmodified version of RSX-11M while the Level-3 computer uses a stripped version of RSX-11M which is essentially RSX-11S. The companion paper on the RSX implementation issues discusses the modifications and extensions to RSX in detail.

Languages

With a single exception, all of the application-level utilities which were specifically developed for the EPICS system have been written in PASCAL. The exception was an existing plotting utility, written in FORTRAN IV, which was taken from the previous control system and extended to meet the new requirements. Where it was necessary to interface the PASCAL procedures to facilities provided by the RSX directives, either SYSLIB routines or small MACRO-coded subroutines were used.

Early versions of the PASCAL compiler were lacking in extensions which would facilitate the writing of system-level processes; consequently, almost all tasks and processes below the level of the application tasks on the Level-2 computer have been written in the MACRO assembly language. The majority of the tasks and all of the drivers and system extensions on the Level-3 computer have been written in MACRO as well.

In two instances, the ACP (Ancilliary Control Processor) for the beamline terminals on the Level-2 computer and the corresponding terminal handler task on the Level-3 computer, tasks were partially written in the PRAXIS language. PRAXIS is a PASCAL-like systems implementation language which was acquired on an experimental basis. Early in its development there was some hope it could be used for a significant fraction of what must usually be written in MACRO. Although the language itself was quite satisfactory (it is comparable to ADA without suffering from ADA's complexities), the support for PDP-11 computers was inadequate; and so, we suspended its use and are now converting the few existing instances to PASCAL.

With the advent of a new version of the PASCAL compiler, which produces highly optimized code and which has several extensions that allow a controlled relaxation of PASCAL's strict type checking, a significant fraction of the MACRO-coded tasks and system processes could now be written in PASCAL. The advantages in terms of readability and maintainability would be significant. Programs which require major improvements are, whenever possible, being rewritten in PASCAL with little or no degradation in performance.

DEC Software

The DEC software products used in the implementation of the EPICS system, in addition to the RSX executive, were RMS-11, DATATRIEVE, and several RSX utilities.

The master device database, which is organized as a multi-keyed, indexed-sequential file, is accessed through RMS-11 file management services. The database editor and the task which maintains the in-memory copy of those database records which describe currently-referenced devices both use RMS-11 for all accesses. Complex database editing operations which modify whole classes of records in the database -- these are principally done by the system maintainers -- use DATATRIEVE when its data-typing limitations are not too restrictive.

The CBASIC language has been used as a filter to control user access to several of the RSX utility tasks in order to exercise resource control. PIP, EDT, and HELP are all associated with verbs in the CBASIC language and are spawned by the CBASIC interpreter.

Non-DEC Software

The PASCAL compiler used in the project is a product of Oregon Software. The current version, PASCAL-2, produces highly optimized code and has enjoyed a remarkably bug-free history. The recent language extensions have made it possible to write system-level processes such as ACP's with the addition of relatively few MACRO-coded procedures.

The CBASIC interpreter is a modified form of REBEL BASIC obtained from the SHIVA laser fusion project at Lawrence Livermore Laboratory. It was altered to support a new class of process control variables that permit access to beamline devices. Numerous new verbs were added to the original REBEL BASIC language most of which spawn other RSX tasks. These provide controlled access to several RSX utilities and other system-specific tasks such as the database editor.

The PRAXIS language compiler, which was primarily used during the early stages of the project, was also acquired from Lawrence Livermore Laboratory where its development was being funded by the Department of Energy.

FUTURE DIRECTIONS

Now that the EPICS system is installed and has satisfied the original design goals, it has been possible to consider enhancements.

One of the serious resource limitations has been the bank of shared memory which couples the Level-2 and Level-3 computers. The addressing space of the shared memory must be subtracted from the total available physical addressing space for the computer. While this poses no problems for the PDP-11/44 with its 22-bit architecture and the