Fermilab

TM-1243
2311.000

EPICS SYSTEM: SYSTEM STRUCTURE AND USER INTERFACE*

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

February 1984

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

EPICS SYSTEM: SYSTEM STRUCTURE AND USER INTERFACE

Robert E. West
J.F.Bartlett, J.S.Bobbitt
T.E.Lahey, B.J.Kramper, B.A.MacKinnon
Fermi National Accelerator Laboratory
Batavia, Illinois

ABSTRACT

This paper presents the user's view of and the general organization of the EPICS control system at Fermilab. Various subsystems of the EPICS control system are discussed. These include the user command language, software protection, the device database, remote computer interfaces, and several application utilities. This paper is related to two other papers on EPICS: an overview paper and a detailed implementation paper.

INTRODUCTION

The Experimental Physics Interactive Control System (EPICS) at Fermilab is used for time-sensitive monitoring and control of devices on the beamlines. This paper presents the user's view of the EPICS system and is partitioned into the following categories:

1. user command language
2. software protection
3. database
4. user console
5. data acquisition
6. system resource displays
7. error reporting and logging

USER COMMAND LANGUAGE

The user command language is CBASIC, which is a superset of standard BASIC. CBASIC is the user's only access point into the EPICS control system. We obtained from Lawrence Livermore Laboratory a version of BASIC called REBEL BASIC and modified and extended it to meet our needs.

There are several differences between the CBASIC language and a standard version of BASIC. First of all, there is explicit declaration of variable types in CBASIC. A variable may be declared as real, integer, double integer, byte, logical, or string. If a variable is not explicitly declared, the name is "typed" using the first letter of the name. Initially, all first letters correspond to type real. However, this may be changed by use of the IMPLICIT statement.

There are two classes of variables in CBASIC: external and internal. An external variable is the name of a beamline device which is defined in the database and hence is shared among all users. The type of an external variable is also defined in the database. An external variable name consists of four to nine alphanumeric characters. An internal variable is unique to each console. Its type and value are specified by the user. An internal variable name consists of one to three alphanumeric characters. If an underscore is part of the name, the length may be a maximum of nine characters.

We have implemented the lowest level of the standard subroutines defined for CAMAC access by the Department of Energy:

1. CDREG(e,b,c,n,a)
2. CFSA(f,e,d,q)

The subroutine CDREG is used to assign a CAMAC location (b = branch, c = crate, n = station, and a = subaddress) to the name of an internal variable (e).

The subroutine CFSA is used to issue a function code (f) to a previously declared CAMAC location (e). A 24-bit data word (d) may be read or written. The Q response (q) is always returned.

We have also provided a subroutine to return to a variable (k) the bit-encoded status of the preceding CAMAC operation:

CTSTAT(k)

Another feature the CBASIC command language provides is the accessing of utility and application tasks via RSX spawning. We have added a number of verbs to CBASIC which are processed by passing the rest of the command line to a spawned task. Some examples of these are the verbs TYPE, DIR, PURGE, and UNSAVE, which are spawned to the PIP utility for processing. The HELP processor is spawned to provide the users with concise information on a specified subject. The text editor EDT is also spawned from CBASIC.

SOFTWARE PROTECTION

Software protection is implemented via a user account scheme. We use the RSX system accounting

file SY:[0,0]RSX11.SYS. A standard record in this file contains the following information:

1. account name
2. password
3. UIC
4. last logon
5. number of logons

We have added the following fields to a record entry:

1. resource class
2. resource privileges
3. device access privileges

A user identifies himself to the system via the LOG utility, which has a user interface similar to that of the RSX HELLO utility:

LOG <account-name>/<password>

The account editor (AED) is used to create, delete, modify, and examine entries in the accounting file.

Software protection consists of a number of elements. The UIC under RSX provides for file protection. It also defines whether a user is privileged in the RSX sense.

The resource class specifies what type of user, e.g., operator, manager, or system. This is used for the general assignment of resource privileges. For example, the account editor will not allow any user lower than manager class to list account passwords except his own. Resource class is also used to determine quotas when assigning resources to a user.

The resource access mask is used for the specific assignment of resource privileges. For example, the CBASIC interpreter uses this to determine if a user is allowed to use the MCR verb or the direct-CAMAC-access subroutines. It is necessary to explicitly restrict usage of the CAMAC subroutines because they bypass the normal beamline device protection mechanisms.

The device access mask is used to provide beamline device protection. Each device has a protection mask associated with it. When a user requests access to a device, the protection mask of the device is logically "anded" with the device access mask of the user's account. The result of this logical operation must be nonzero, i.e., the intersection of the two masks must contain the type of access requested (e.g., read, write, or control), for the device access to successfully complete.

DATABASE

The beamline device database is a file of beamline device names on disk. A subset of the disk database is maintained in memory. Entries in the in-memory database are hash-indexed to provide for quicker access. The in-memory database is updated from the disk database on a demand basis. When a reference is made to a beamline device, the in-memory database is searched for the device entry. If it is not found, the disk database is searched and the entry is put into the in-memory database.

It is possible that a device entry in the in-memory database is not the most recent, i.e., the device entry in the disk database has been updated by a user. In this case, the entry from the disk is moved to the in-memory database on the next new reference to the device. The old entry is deleted when the last reference to it completes. Under this scheme, it is possible for a single device to have several in-memory database entries which are of different "ages."

Eventually, an attemp to insert an entry into the in-memory database from the disk database will fail because the in-memory database is full. When this occurs, the least recently used database entry is deleted and the space is used for the new entry.

There are four types of descriptor records in the beamline device database:

1. MDR - Module Descriptor Record
2. PDR - Physical Descriptor Record
3. SDR - Simple logical Descriptor Record
4. CDR - Compound logical Descriptor Record

The Module Descriptor Record (MDR) describes common characteristics of a module type. Some of the characteristics described are:

1. status bits
2. attributes such as reading, setting, status, and control
3. data formats for each attribute
4. units of measurement

The Physical Descriptor Record (PDR) describes the unique physical characteristics of a module. A PDR is linked to an MDR via the module type. Some of the characteristics described by a PDR are:

1. CAMAC address - crate and slot, which uniquely identify the PDR
2. setting limits for selected attributes

The Simple logical Descriptor Record (SDR) describes the variable characteristics of a module's physical occurrence described by a PDR. An SDR is linked to a PDR via the CAMAC address. SDRs which reference the same PDR are synonyms. An SDR supports different ways of using a device. Some of the characteristics described by an SDR are:

1. device name, which uniquely identifies the SDR
2. transformation parameters, e.g., scale factor
3. protection mask
4. owner UIC

The Compound logical Descriptor Record (CDR) describes an arithmetic or boolean relation of simple logical device attributes. An example is the ratio of the reading attributes of two devices.

Some of the information contained in a CDR is:

1. compound device name
2. component SDR names
3. attribute for each component SDR
4. transformation parameters, e.g., scale factors
5. compound device type

A CDR has only one attribute which is either read-only or write-only.

There are two disk database utilities: (1) the database editor (DBE) and (2) the database list utility (DBL). The database editor is used to create and modify entries in the disk database. It interacts with the user in a question and answer format. The database list utility lists selected entries in the disk database. The user may select which devices to list according to a number of different criteria:

1. all module types

2. all occurrences of a module type
   1. a single type
   2. an alphabetical range of types

3. device name
   1. a single name
   2. an alphabetical range of names
   3. all names

4. all references to a CAMAC location
   1. crate and slot
   2. crate
   3. a range of crates

For each device entry listed, the following information is output:

1. device name
2. module type
3. crate
4. slot
5. channel (subaddress)
6. scale factors

USER CONSOLE

A typical beamline console consists of a CAMAC module, a CRT, and a keyboard. A device name must be defined in the database for the CAMAC module. In addition, a default user account must be associated with this module definition. The default user account is associated with the console whenever a carriage return is typed at the console and no CBASIC task is associated with the console, which occurs after:

1. system reboot
2. the CBASIC interpreter exits, which occurs when:
   1. the user types BYE
   2. the console times-out as a result of no I/O activity

The default user account continues to be in effect until a user specifies another account via the LOG utility.

The CRT may be divided into multiple display frames, which are managed by the terminal ACP. If the console has local microprocessor intelligence, a graphics frame may also be defined on the CRT.

The keyboard consists of three parts, each of which is separately attachable by a utility program:

1. typewriter keys
2. auxiliary keypad - used for special editing functions and device control
3. special function keys - utility specific

As a result of the multiple display frames and the multiple keyboard sections, more than one task may be interacting with the console at the same time.

DATA ACQUISITION

The primary input to most of the data acquisition processes in the EPICS system is the name of a file on disk. This file is called a page file and specifies a list of device names. The format of the file is:

1. a record containing a one line description of the file
2. a default device record containing the values to be substituted into corresponding device record fields not explicitly specified by the user
3. a maximum of 127 device records

The format of a device record in this file is:

1. the name of the device
2. the time at which the device is to be read
3. the value to which the device is to be set
4. device monitoring parameters
   1. the time at which the device is to be read
   2. lower and upper limits on the value of the device
   3. a 24-bit data mask
   4. the minimum number of consecutive violations before alarming
   5. the name of a CBASIC program to be executed upon alarming

The page editor (PEDIT) is used to create and modify page files. This is a relatively simple line-oriented editor. Its command parameters may be positional, keyword, or positional followed by keyword.

Data acquisition in the EPICS system may be classified in one of five categories:

1. setup
2. repetitive display
3. device monitoring
4. repetitive distribution
5. graphics

Setup: Save and Restore Utility (SAR)
The primary purpose of the Save and Restore Utility is to save the values of a list of devices and then later restore these values to the devices. The list of devices is specified by a page file.

For the save function, SAR reads the value of each device specified via the page file and then writes the value into the setting field of the device record in the file.

For the restore function, SAR reads the values from each device record in the page file and then writes the values to the corresponding devices. Alternatively, the user may specify that a value, usually zero, be written to each device specified by the page file.

Repetitive Display: Page Display Utility (PAGE)
The page display utility displays a list of beamline devices and their values. The list of devices is specified by a page file. PAGE displays a window of consecutive device record entries from the file. This display window may contain as few as one record or as many as fifteen records. PAGE allows the user to scroll this window forward and backward through the file in increments of either a single device or a window of devices.

The page utility gathers data only for the devices displayed on the screen. For each displayed device, it reads the device registers and displays the current values for a maximum of three attributes: reading, setting, and status. Device data is gathered at one of three rates:

1. once each accelerator cycle at the time specified in the device record
2. once every two seconds if the user specified a read time of continuous in the device record
3. five times per second if the user has requested the page utility perform "fast" data acquisition on a device. At most three "fast" data acquisitions may be associated with a single display.

If allowed by protection and device attributes, a user may increment or decrement a displayed device by attaching a knob to the device. A page display may have two knobs (X and Y) associated with it. Four keys on the auxiliary keypad are used to control each knob process: attach, detach, increment, and decrement. Data for a knobbed device is gathered and displayed at the fast rate of five times per second.

When a page display is initiated, the page utility attaches to the keypad and to the special function keys of the console but not to the keyboard. For its output, page uses only part of the CRT display, the size of which is specifiable by the user. Therefore, another utility may use the console at the same time as PAGE if all that utility needs is keyboard input.

Device Monitoring: Watch Utility
The watch utility monitors readings of devices specified via a page file and activates an alarm if specified constraints are violated. There are two types of alarm monitoring which may be initia. upon a device: (1) limit and (2) bit-pattern.

For a limit watch, the user specifies a lower limit and an upper limit. Three possible limit types may be specified:

1. a value
2. a delta value - the limit is calculated by determining an initial value and then adding the delta value to the initial value
3. a delta percentage - the limit is calculated by determining an initial value, taking the specified percentage of this initial value, and then adding the result to the initial value

The initial value is obtained by reading the device at the specified time for a number of accelerator cycles and taking the average of the readings.

For a bit-pattern watch, the user specifies a 24-bit mask with each bit position containing a 0, 1, X, or R. The X indicates ignore that bit. The R indicates determine an initial value for that bit position by reading the device.

Whenever the watch utility has to determine the initial value it is to use for a device, it must go through a setup phase of reading the device multiple times. The user specifies the number of accelerator cycles in this setup phase via the repeat parameter in the device's record entry. This repeat parameter also specifies the number of consecutive violations of the boundary constraints which must occur before WATCH generates an alarm for the device.

Whenever a device goes into alarm, the watch utility activates an audible alarm and/or displays an alarm message at a specified console. A message is also displayed when a device goes out of alarm. An alarm message may identify which device changed state or it may only specify the number of devices currently in alarm.

The maximum number of devices which may be monitored at one time by a particular user depends on the resource class of the user's account.

Repetitive Distribution
Pulse Train Manager (PTM) - A pulse train module serves as a relatively simple data acquisition interface between the EPICS system and an experimenter's computer. A pulse train module is placed in a CAMAC crate accessible by EPICS. A cable is connected from the front of the pulse train module to a scalar module in a CAMAC crate connected to the experimenter's computer. When a value is written by the EPICS system to the pulse train module, the module generates an integral number of pulses. These are transmitted via the connecting cable to the scalar module and are counted by the module. This count is then read by the experimenter's computer.

The pulse train manager reads data at a specified time from a specified source device and writes that data to a specified destination, which is a channel in a pulse train module. The primary input to PTM is the name of a pulse train file, which is an ASCII text file created using the text editor EDT. A record in this file has the following

format:

<destination-name> <source-name> <read-time>

Also input to the pulse train manager is the name of a page file which contains all source devices referenced in the pulse train file. The device specifications in the page file must exactly match the source specifications in the pulse train file. The primary purpose of this page file is to enable the user to see a display of the names and data values of the devices which are being distributed to pulse train modules.

Remote Computer Data Acquisition (RCDA) - The Remote Computer Data Acquisition utility links an experimenter's computer to the EPICS system for the purpose of acquiring relatively large amounts of data. Communication between the RCDA utility and the experimenter's computer is done through parallel 032 CAMAC modules. An 032 module has an input FIFO and an output FIFO, each consisting of 256 16-bit words. One 032 module resides in a crate on the EPICS system. This module is connected via cables to a second module which resides in a crate on the experimenter's system. The experimenter's computer writes commands into its 032 module. These commands are directly transmitted to the 032 module on the EPICS system where they are read by the RCDA utility. The RCDA utility writes responses and data into its 032 module. This information is transmitted to the parallel module and is read by the experimenter's computer.

The list of devices to read is specified via a page file. In this case, however, the experimenter does not specify a complete file name but only a one-digit identifying number. This number is then inserted into a filename of the format EXPDAQ#.PAG. A user account name must be associated with the device name of the 032 CAMAC module residing on the EPICS system. The associated UIC is accessed from this account and combined with the constructed filename. The requested page file is then opened for input.

The communications sequence which occurs between the experimenter's computer and the RCDA utility is as follows:

1. R # - the experimenter's computer issues a read request for a data acquisition process to be setup. The specified number identifies a particular page file which contains the list of devices from which to gather data.

2. GOOD - when the RCDA utility has successfully initiated the data acquisition process, it returns a response of GOOD to the experimenter's computer.

3. data - at the completion of data acquisition in an accelerator cycle, the RCDA utility outputs the data. A header block for the transmission specifies the identifying number of the data acquisition process and the amount of data being transferred.

4. A # - after reading all the data, the experimenter's computer must issue an acknowledge in order to receive more data. The identifying number must be specified because there may be multiple data acquisition processes underway for a single experimenter interface.

5. E # - the passing of data and acknowledges continues until the experimenter's computer issues an end request to terminate the specified data acquisition process.

6. GOOD - the RCDA utility terminates the associated data acquisition process and returns a response of GOOD to the experimenter's computer.

Graphics: PLT Utility

Each user executing the graphics utility PLT may gather and plot data for a maximum of four devices. A ratio of the values from two devices may also be plotted. Because graphics puts a relatively heavy load on the system, the total number of devices being plotted at any one time needs to be limited.

There are two types of plots: (1) time and (2) scatter. For a time plot, multiple points are gathered between specified times each accelerator cycle for each device. The time range is divided into 120 equal intervals and a data value is read at the start of each interval. The Y coordinate of a point on the graph is the value read from the device and the X coordinate is the time at which the device was read. Multiple device values may be plotted on the Y-axis for each time value on the X-axis.

For a scatter plot, a single data point is gathered at a specified time each accelerator cycle for each device. The Y coordinate of a point on this graph is the value of a device read at a specified time. The X coordinate is the value of another device read at possibly a different time. As with the time plot, multiple devices may be plotted on the Y axis.

SYSTEM RESOURCE DISPLAYS

The SHOW utility enables the user to display various system status information. The command options are the following:

1. USERS - displays the consoles currently active on the system. The following information is displayed for each active console:

   1. logical console number
   2. physical console number and CAMAC location
   3. UIC
   4. user account name
   5. names and states of attached tasks

2. SHAREDMEMORY - displays the allocation and usage of the memory region shared between the level 2 and level 3 computers
3. UIC - displays the terminal's default UIC
4. POOL - displays the size of the pool dynamic storage region
5. TASKS - displays a list of the names of all active tasks

ERROR REPORTING AND LOGGING

Tasks in the EPICS system which detect an error condition format a data packet of related information and issue a send-data request to transmit the information packet to the error reporting system. The error processing utility uses a key in the packet to locate a corresponding text string on disk. The RSX system subroutine SEDMSG is then used to format the passed information into the text string and the result is written to the console log and, optionally, to a designated terminal.