A major purpose of the Technical Information Center is to provide the broadest dissemination possible of information contained in DOE's Research and Development Reports to business, industry, the academic community, and federal, state and local governments.

Although a small portion of this report is not reproducible, it is being made available to expedite the availability of information on the research discussed herein.

CONF-830501--4

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36.

LA-UR--83-1387

DE83 012671

TITLE: PERFORMANCE RESULTS FOR THE NEW ^{RSX}~~RSX~~-11M Q SYSTEM

NOTICE

PORTIONS OF THIS REPORT ARE ILLEGIBLE.

It has been reproduced from the best available copy to permit the broadest possible availability.

AUTHOR(S): Thomas Kozlowski, James F. Amann, Gail T. Anderson,
Richard A. Floyd, James F. Harrison, Michael A. Oothoudt

SUBMITTED TO: Presented at the "Real Time Computer Conference"
Berkeley, CA May 16-19, 1983

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

MASTER

By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes.

The Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy.

Los Alamos Los Alamos National Laboratory
Los Alamos, New Mexico 87545

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

PERFORMANCE RESULTS FOR THE NEW RSX-11M Q SYSTEM*

Thomas Kozlowski+, James F. Amann+,
Gail T. Anderson+, Richard A. Floyd#,
James F. Harrison+, and Michael A. Othoudt+

Abstract

The new RSX-11M version of the Q data acquisition and data replay system has been in use for approximately one year. The structure and flow of data through the system is described. Performance data is presented on the Q system event acquisition, event distribution, histogramming, data testing and tape writing.

The Q System

Purpose of the Q System

The purpose of the Q project is to provide a high-performance general-purpose experimental data-acquisition system. The system was developed primarily for use in the "medium energy" physics environment at the Los Alamos Meson Physics Facility. Important requirements in this environment include high speed data acquisition, event by event data logging, extensive on-line event analysis and experiment monitoring. Event sizes typically range from a few sixteen bit words to a few thousand words.

Q is designed to provide those facilities routinely required by all experimenters, such as the acquisition and recording of data on magnetic tape, histogramming of data, and display of histograms. Q attempts to provide sufficient computing support so that the experimenter can be free to concentrate on physics without distraction by complex computing issues involved with buffering schemes, multiprocessor communication protocols, memory allocation methods, queuing systems, and the like. In the majority of Q experiments, the only code a user needs to supply is that specific to the physics of the particular experiment, for example, subroutines that process raw data from the detector system to identify particle types or to calculate missing masses. These aspects of the computing problem are usually readily dealt with in FORTRAN.

Because a general-purpose system is unlikely to suit all needs equally well, a major goal of the project has been to provide a modular system that is relatively easy to understand, modify, and maintain.

Structure of the Q system

The functional components are diagrammed in figure 1.

*Work supported by the U. S. Department of Energy.

+Los Alamos National Laboratory, Los Alamos, NM 87545.

#Presently at University of Rochester, Rochester, NY 14627.

Q System Components

Figure 1

The facilities that are provided by the Q system include: event data acquisition, distribution of events to user written processing subroutines, histogramming, testing of event data (arithmetic and logical tests), tape writing and an experiment parameter management system. An overview of the Q system may be found in reference 1.

The acquisition system contains three main components: an MBD (Micro-programmable Branch Driver) that acquires data; a PDP-11 resident subsystem that receives the data from the MBD and synchronizes operation of the system, and a largely user written "analyzer" task.

MBD. Event data acquisition is done by the code that resides in the MBD. In the case of "replay" of a previously written Q data tape, the corresponding process is the reading of a multi-event buffer into PDP-11 memory from magnetic tape. The event data acquisition process includes the recognition of a particular event trigger, the execution of the CAMAC and logical operations associated with the event trigger and the dumping of the contents of the MBD buffer into a PDP-11 buffer for further processing and writing to tape. Various optional ways of dumping the MBD buffer into the PDP-11 for processing are available: when the MBD buffer is filled, at the end of every event, or on command from the MBD event specification code. A special language, QAL (Q Acquisition Language), and compiler has been provided to facilitate the user's writing of the event specific part of the MBD code.

PDP-11 Resident Q System. The underlying PDP-11 subsystem acquires the data either from the CAMAC system (data acquisition) or a Q format input tape (replay), supplies the data for each experimental event to the analyzer for processing, and allows keyboard control of the experiment. In the data acquisition mode, the raw data are recorded on an output magnetic tape if the user so chooses. User-generated data may be recorded to an output tape in either mode.

The system is built around an RSX-11M device driver (QX:) and an ACP (Ancillary Control Processor) which is a task which serves as an extension of the driver. A device driver provides a well defined interface between a user task and the RSX operating system (usually for the purpose of controlling hardware devices). The ACP contains the MBD interrupt service routine and the code for writing of data buffers to magnetic tape.

Analyzer Task. The analyzer task is an important part of the Q system. Since the analyzer contains the event data processing subroutines, the user must provide much of the code. Events are distributed by the analyzer task from the PDP-11 multi-event buffers for processing by user written subroutines. The analyzer task does this by calling a particular named subroutine (user written) when a particular event is encountered in a buffer. Facilities for retrieving data, histogramming, data testing and dot-plotting are provided to support the user written processing. Selected event data can easily be written to the output tape by the user processing code.

The analyzer task contains the Q multi-event data buffers and system data base.

Q System Data Processing

MBD Data Acquisition

Actual data acquisition occurs asynchronously to the rest of the Q system, since the MBD is a micro-processor and executes most of the actual data acquisition operations. Figure 2 shows the three main asynchronous data flow streams of the Q system.

Reading Event Data into the MBD Buffer. A hardware trigger received by the special Q system Trigger Module (CAMAC module) is associated with a particular "event number" (0-32) by the hardware. The MBD data acquisition code is started by a LAM from the Trigger Module. A separate procedure exists in the MBD for each such event number. These procedures consist of event specifications (generally CAMAC operations) supplied by the user and compiled by QAL.

Data acquired in response to an event trigger are transferred by the MBD into its internal multi-event buffer. The data in this first layer of buffering are stored temporarily in the MBD for transfer to the PDP-11 later, either on request or when the MBD buffer fills. The size of the MBD buffer and the size and number of the PDP-11 buffers may be specified by the user.

Transferring Event Data to the PDP-11. The second layer of buffering takes place in the raw data buffers in PDP-11 memory. These buffers may be larger, the same size, or smaller than the MBD buffer. The MBD can fill these buffers in three modes: the single-event-per-buffer mode, the

Q System Data Processing
Figure 2

multiple-event-per-buffer mode, or by a QAL "flush event". In the single-event-per-buffer mode, the MBD selects a new PDP-11 buffer at the end of every event and alerts the PDP-11 via an interrupt that this operation has been accomplished. In the multiple-event-per-buffer mode, the PDP-11 is not interrupted until an entire raw data buffer in the PDP-11 is filled. The multiple-event mode is normally used during "production" data-acquisition runs because it takes maximum advantage of the multiprocessor aspects of the system and, therefore, gives the highest performance. The single-event mode is usually used for prompt response for debugging purposes. Single- or multiple-event-per-buffer mode is selected by the keyboard command used to start a new run. "Flush" events (specified by a QAL statement) cause the MBD buffer to be dumped, up to and including the present event, to the PDP-11 buffers, and the last PDP-11 buffer (containing the flush event) is declared to be full and ready for processing. The flush event is useful for guaranteeing fast response for processing special user-control events. In any case, the PDP-11 can perform other processing, such as writing previously filled buffers to magnetic tape or analyzing previous events, while the MBD is filling the current PDP-11 buffer.

Event Distribution

Event distribution is primarily a function of the analyzer task; this is because it contains the data buffers and the user's event processing subroutines.

The raw data buffers in the PDP-11 contain events to be processed and logged to tape. Events are variable in length and the length can vary even for the same event number; however, the size of an event is limited by the size of the MBD buffer or PDP-11 buffer, whichever is smaller. An event larger than the maximum size will be truncated. When a raw data buffer has been filled by the MBD (or from an input tape in the case of replay), it is ready for event distribution and logging to tape. In the case of data generated by the analyzer through user data buffers, a user data buffer is ready for taping when it has been filled.

There are three possible processing options available for an event in the PDP-11: "must process," "may process," and "no process." If a particular event number has been specified by the experimenter as "must process," all events of that class are distributed to his analyzer processing subroutines, even if data acquisition must be slowed down to do so. "Must process" is usually specified if the user wants to perform his own taping of events or for those events that are used for experiment control. If "no process" is specified for an event, no events of that type are distributed for processing. If "may process" is specified for an event, the user's processing subroutine receives a fraction of that type of event. The fraction depends on the rate at which events are being acquired and the processing speed of the analyzer. In "may process," every effort is made to allow data acquisition to proceed at the highest possible rate and the limiting factor should be recording speed or CAMAC acquisition speed, not event-processing speed. The algorithm used is as follows: If the MBD has run out of PDP-11 raw data buffers and internal MBD buffer space, only "must process" events are processed; the system returns to distribution of all "may" and "must" events whenever a raw data buffer has been completely processed and tapped (if logging of raw data is enabled).

Events are distributed by the analyzer in the following manner: The first event in the buffer is

examined. If the event is "no-process" or if the event is "may process" and the system is NOT in the normal distribution state, the event is skipped; the analyzer processing subroutine is not called, and the next event in the buffer is examined immediately. If the event is "must process" or if the event is "may process" and the system is in the "normal" distribution state, the analyzer processing subroutine is called. When an event processing subroutine finishes, control is returned to the event distributor so that another event of the same or a different type can be processed.

User Event Processing

Event processing is determined entirely by the user. If no event processing subroutine (PROCn) is provided by the user a subroutine is provided from the Q system library which only does a RETURN.

Event Retrieval. The processing subroutine can retrieve any subset of the data for the event into a local array by calling an event data retrieval subroutine. An event consists of two (sixteen bit) header words followed by the event data (if any). The event tag or header consists of two words. The first word contains the number of words of event data (including the header). The second word contains the event number as determined either from the trigger causing the event or by the user (in the case of events generated directly by user written code).

Data Testing. A Q subsystem is provided for applying sets ("blocks") of commonly used arithmetic tests to data and commonly used logical tests to the results of previous tests. Some of the available tests are:

Arithmetic Tests

BIT n,nb	Is bit 'nb' set in word "LARRAY(n)"?
EQUAL n,v	Is data word "LARRAY(n)" = "v"?
GATE n,l>,hi	Is word "LARRAY(n)" >= "lc" and < "hi"?

Logical Tests

AND n1,...,nn	Is AND of tests n1,...,nn true?
IOR n1,...,nn	Is inclusive OR of tests n1,...,nn true?
EOR n1,...,nn	Is exclusive OR of tests n1,...,nn true?
USER	Is user defined result true?

The definitions of these tests may be changed on-line without re-compiling and re-linking the code, by editing a test file when reading it in via a keyboard command. Tests are executed in "blocks". A test block is a set of tests defined in the descriptor file that are all executed by means of a single subroutine call in the user's event processing code. As many blocks of tests as desired may be set up. The data testing system maintains an array of logical variables indicating the result of every test executed. An array of double word counters (one for each test) that count the number of times each test is successful is maintained. Subroutines are available to examine these arrays.

Histogramming. A histogramming subsystem is provided that supports a general purpose histogramming system. Histograms are set up on-line (via keyboard command or command files) in "blocks". A block of histograms is incremented by a single subroutine call. The subroutine arguments include a block number and a data array address. The system attempts to increment each histogram that has been defined in the block; the data to be histogrammed resides in the specified data array. Each histogram may be "cut" (conditionally incremented) according to the result of a particular

data test (see previous section).

Putting Data onto the Output Tape. The analyzer task may contain a set of user data buffers with a structure like that of the raw data buffers. Data is put into these buffers by the user-written processing routines and, when a buffer is filled, it is automatically written to the output tape if taping is enabled. Subroutines are available to put any data array or the current event into a user data buffer. The user data buffers provide a means for writing user data to tape and for supporting an event "filtering" mode in which only event data selected by the processing routines are written to tape. This latter mode requires that the taping of raw data buffers be disabled (if not disabled they are automatically taped).

Data logging

The Q system automatically logs both the raw data buffers and any user data buffers to an output tape if this feature is enabled. The taping proceeds asynchronously to the analyzer task event analysis and the filling of the data buffers by the MBD. As soon as the MBD has filled a buffer it marks it ready for taping, interrupts the PDP-11 and request the Q system ACP to run (the ACP does the actual taping). Only one I/O request at a time is issued for the output tape. As soon as a tape I/O request has completed, the ACP issues a new request if there is a buffer ready for taping.

Timing Measurements

A set of timing measurements has been made on the various components of the Q system. These measurements were done using the following specific hardware: a PDP-11/44 CPU with cache enabled, a PDP-11/44 CPU with cache disabled, a tape drive at 1600 bpi and 75 ips. The timing studies were done using a pulser, imitating the effectively DC structure of the LAMPF beam (6 percent duty factor, 120Hz repetition rate).

Several Q system parameters that are optional to the user influence system timing; these include the MBD buffer size, sizes and numbers of the PDP-11 buffers, and the size of an event. For most of the timing tests an MBD buffer size of 500 words, and two 1500 word PDP-11 raw data buffers and two 1500 word PDP-11 user data buffers were used. Two buffers of each type are used so that one may be filled while the other is being processed or taped. The event sizes were varied from 0 to 498 data words. All possible Q system processing options were used: "no", "may" and "must" processing, taping, no taping, and taping data via the user event processing code ("filtering").

From these measurements specific timing data were extracted for all parts of the path of events through the system. It should be noted that the three processes, MBD data acquisition, analyzer processing, and data logging, proceed (for the most part) asynchronously to each other. The dominating process determines the throughput of the system. The one that dominates depends on the event sizes, event frequency, buffer sizes, tape speed and processing algorithms.

MBD Data Acquisition

The accuracy of the MBD results are limited by variations in MBD clock frequency (10 percent) and variations in crate controller CAMAC cycle times (50 percent or more).

Reading Event Data into the MBD Buffer. The actual MBD acquisition time depends a great deal on the algorithms used in acquiring the CAMAC data. These timing test were made using a QAL event specification that consisted only of 16 bit read statements, for example,

```
SPECIFY EVENT 5
RD16 S1,S2,S3,S4,S5,S6,S7,S8
PROCESS EVENT 5
END EVENT 5
```

The time obtained for reading a single event from CAMAC into the internal multi-event MBD buffer is:

$$22 + 4.9 * (\text{number data words}) \quad (\text{microseconds})$$

where "number of data words" does not include the two word event header.

Transferring Events to the PDP-11. The time to transfer the internal MBD buffer to the PDP-11 raw data buffer is approximately given by:

$$62 + 4.7 * (\text{number words in MBD buffer}) \quad (\text{microseconds})$$

where "number of words in MBD buffer" includes all the two word event headers. This result includes the variable but difficult to measure overhead due to UNIBUS activity during the DMA from the MBD to the PDP-11 memory. It also assumes sufficient space is available in a PDP-11 buffer to contain the entire MBD buffer.

Event Distribution

The times associated with event distribution include the MBD buffer full interrupt service, the time for the Q system to be notified of the buffer and to access it, time to scan the buffer for events and time to call the user supplied processing subroutine.

The measured complete buffer processing time (in microseconds) is given by the following data:

8000 us. per buffer (cache off)
7000 us. per buffer (cache on)

126 us. per event to find the event (cache off)
72 us. per event to find the event (cache on)

116 us. per event to call processor (cache off)
62 us. per event to call processor (cache on)

User Event Processing

The user event processing may often be quite extensive. The timing data presented here include only the basic operations commonly used and supported by the Q system.

Event Retrieval. In most cases the event processor retrieves the data for an entire event into a local FORTRAN array. In other more special cases it

may be necessary to retrieve only a portion of event and then possibly make a decision whether or not to retrieve more data and to proceed with the event analysis. The measured times for retrieving event data are:

$$120 + 5.5 * (\text{number words}) \text{ (microseconds) (cache off)}$$

$$58 + 4.0 * (\text{number words}) \text{ (microseconds) (cache on)}$$

where "number words" is the number of words retrieved from the event data.

Data Testing. Because of the complexity of possible test types and combinations it is not possible to specify the performance in terms of only 1 or 2 numbers. In general there is an overhead for each call to execute a block of tests plus an execution time for each test performed during the call. This latter number depends on the specific type of test, whether it was passed or failed, and if it failed where in the course of the test execution the failure was encountered. In order to use the test the logical flags must be cleared before the tests are executed. The time to clear the test result flags for all tests is given by:

$$222 + 2.76 * (\text{number tests}) \text{ (microseconds) (cache off)}$$

$$119 + 1.66 * (\text{number tests}) \text{ (microseconds) (cache on)}$$

where "number of tests" is the number of tests defined in the block.

The time required for a given call to execute a block of tests is a combination of the overhead time,

$$143 \text{ microseconds per block (cache off)}$$

$$75 \text{ microseconds per block (cache on)}$$

plus the time required to execute the tests in that block. The time needed for a given test depends both on the type of test and on how much of the specified test information must be examined before a "true"/"false" decision can be made.

In general it takes from 20 to 60 microseconds (cache on) or 40 to 110 microseconds (cache off) to execute a test. An additional 10 microseconds (cache enabled) or 20 microseconds (cache disabled) is required if a test is passed, to set the test result flag and increment the test counter.

Histogramming. Because of the complexity of the options available both in the call to increment a block of histograms and in the definition of histograms (no test cut, test cut, one parameter or two parameter, etc.), it is not possible to characterize performance in terms of one or two numbers. In the simple case of one parameter histograms defined without any test cuts and for which the data falls in the limits of the histogram the times for incrementing a block of histograms are as follows:

$$1530 + 84 * (\text{number}) \text{ (microseconds) (cache on)}$$

$$1930 + 146 * (\text{number}) \text{ (microseconds) (cache off)}$$

where "number" is the number of histograms defined in the block.

Putting Data onto the Output Tape. The analyzer event processors are able to put data into user data buffers for logging to the output tape. The transfer of data by the user event processing code from a local array into a user data buffer, excluding the actual time overhead for writing the buffer to tape (see section on data logging), is given by the following expression:

$$190 + 5.5 * (\text{number words}) \text{ (microseconds) (cache off)}$$

$$94 + 4.0 * (\text{number words}) \text{ (microseconds) (cache on)}$$

where "number of words" is the number of words put into the user data buffer.

Data logging

An important quantity with respect to speed of logging data to tape is the time it takes the tape drive to write a record for tape. In general this may be found from an expression of the form:

$$\frac{(\text{header size}) + (\text{number data words})}{(\text{tape density in words}) * (\text{tape speed})} + (\text{gap time})$$

"Gap time" is the time required to move past the interrecord gap. For the hardware used in these measurements the expression reduces to:

$$\frac{9 + (\text{number data words})}{800 * 75} + .020 \text{ (seconds)}$$

In the case of our 1500 data word buffers this reduces to 45.2 milliseconds.

The time overhead to set up a buffer for taping (measured indirectly) appears to be 16.5 milliseconds. If the data in a buffer can be acquired from CAMAC or processed in the PDP-11 (whichever takes longer) faster than this tape writing time, it is possible to log to tape at tape writing speed. In most cases at least two raw data buffers are required to accomplish this. In the case of event processing that only consists of retrieving data and possibly moving it to a user output buffer, the Q system can write data close to the maximum tape speed when event sizes are on the order of or larger than about 10 words and with the buffer sizes and tape drive hardware used in these measurements. Even in the case of more extensive event processing, if it is possible to run in a "may process" mode, taping will proceed at close to maximum speed.

References

1. J. F. Harrison, T. Kozlowski, R. A. Floyd, J. F. Amann, G. T. Anderson, M. A. Oothoudt, and D. G. Perry, "Design of the RSX11M Q System," IEEE Transactions on Nuclear Science, Vol. NS-28, No. 2, p. 3724-3730 Oct. 1981.