BNL 17827

Conf-730540--1

UNIQUE ASPECTS OF UTS REAL-TIME DATA ACQUISITION AT BNL*

P. C. Rogers and J. J. Kolata

Brookhaven National Laboratory
Upton, New York 11973

Abstract

The UTS System has been used at BNL in a real-time environment for collection, display, and sorting of nuclear physics data. This is in addition to the usual batch streams and on-line terminal operation. The data acquisition routine consists of a small resident portion which buffers incoming data onto disc and tape and a ghost job which sorts and displays the results every 1.7 seconds. Extensive use is made of the various entries to NEWQ but few other special features of UTS are required. Data rates in excess of 40,000 events per second have been input with no ill effects outside of response degradation for normal users.

*Work supported by the U. S. Atomic Energy Commission.

Introduction

The UTS system has been utilized at BNL in a real-time environment for collection, display, and sorting of nuclear physics data. The data derived from experiments being performed at the Tandem Van de Graaff Facility through the use of its two MP Tandem Van de Graaff accelerators. These accelerators can operate at 10 MV terminal potential either simultaneously and independently, or conjunctively in three-stage mode.

Experiments are conducted by an in-house staff of about 25 physicists and an outside user group of about equal size. With such a large experimental staff and with accelerators which can be quickly and easily changed from one beam and energy to another, the facility computing complex is required to provide not only for comprehensive, easily variable data acquisition but also for concurrent data-base manipulation of the experimental spectra, complex final data analyses, program development, and normal batch and on-line facilities.

To meet these requirements, the UTS system was chosen, with some trepidation, since its extension to real-time was a task of unknown complexity and since the resultant ability to handle high rates of experimental data was in question. It was assumed, correctly, that UTS would be capable of adequately handling the problems of data-base manipulation, analyses, program development, etc. in conventional fashion.

Our extension of UTS to real-time took the form of a single sizable program, named CASPER the friendly ghost, which is split into a non-resident portion which operates as a ghost job, while collection is in progress and a core resident portion which remains present at all times. The resident portion is interrupt driven, providing a short response time to the data collection requirements. The non-resident portion is awakened at regular intervals to sort the existing data and update the display. Its response time is dictated by the need to provide the human with a display update rate which is reasonable to him--currently every 1.7 seconds. An overall block diagram of CASPER is shown in Fig. 1.

I. Resident Real-Time Data Collection

The resident portion of CASPER consists of a pool of eleven 256-word data buffers, a table of constants, pointers, etc., and a series of semirelated tasks which are driven by four external interrupts.

The buffers are arranged in a queuing structure consisting of EMPTY, FULL, and EMR queues. The EMPTY queue is a pool of buffers available for attachment to data-collection channels or for dedication to one of several other internal uses. The FULL queue consists of buffers which presently contain data to be sorted. The EMR queues (maximum of 2 entries) contains buffers whose contents are to be written to magnetic tape in event-by-event fashion (event-mode recording).

The function of the resident tasks is basically to manipulate the buffers in the pool--moving them between the EMPTY, FULL, and EMR queues with an eye toward keeping each active data-collecting I/O channel attached to a buffer at all times. Eight I/O channels (unknown to UTS) are so dedicated but all eight are not necessarily in use at one time.

The tasks have been written with great attention paid to their speed of operation. In particular, these resident interrupt tasks are only involved with full buffers of 500 events, not event by event manipulations. Also, buffers are themselves buffered on a 7212 RAD until CASPER (non-resident) is awake to sort them into the final spectra--at which time they are read back and sorted.

This resident portion of CASPER occupies 4K of core between 76 K and 80 K, above the known top of UTS core. The tasks are associated with four levels of external interrupt priority as described below.

I.1 Disc Location Counter (1F1 Interrupt)

This counter is incremented every 400 μs by the sector gap of the 7212 RAD. It serves the very important function of providing at all times the relative rotational position of the disc.

I.2.1 Disc Timer (F3 Interrupt)

This interrupt is triggered 30 times per second by the track origin signal of the 7212 RAD. It is used to reset the Disc Location Counter and to provide general timing for all CASPER tasks-- in particular, time to display (1.7 seconds), time to save all spectra on disc (30 seconds) and others.

I.2.2 I/O Channel Control (F1 Interrupt)

This interrupt is triggered by incoming data filling a buffer (I/O interrupt). It puts the buffer in the FULL queue, acquires one from the EMPTY queue, and restarts the I/O channel if at least one empty buffer is available--these channels are not known to UTS.

I.3 Service Dispenser (F2 Interrupt)

This service interrupt is triggered by a buffer being entered into any queue. It is used to drive the tasks which dispose of full buffers of data. These tasks ultimately return the buffers to the EMPTY queue for reuse. In addition it is responsible for restarting a data I/O channel which was not started by F1 (see above) and for causing the non-resident CASPER to be awakened when required. To increase efficiency, non-resident CASPER is only awakened if new data have arrived for a spectrum on display. Otherwise it waits for the time to save all data (30 sec) or for the disc buffer storage to become at least 3/4 full.

Those tasks which utilize buffers and ultimately return them to the EMPTY queue are described below.

I.3.1 EMR TAPE output

If desired by the experimenter, CASPER can write the raw descriptor buffers on a tape for future analysis. Such tapes are called event-mode-recorded (EMR) tapes. After a buffer is so written it may be returned to either the FULL queue for direct sorting into a spectrum or to the EMPTY queue.

I.3.2 Litepen Tag I/O

In our facility, the display and associated litepen tags are kept on a separate disc (7212 like--only faster). In order to tag a displayed point, the disc address is entered through the F1 interrupt, a display disc sector (250 words) must be read, modified, and rewritten. This process uses a buffer from the EMPTY queue and returns it when no further tags are to be processed.

I.3.3 Disc Buffering

In order to reduce our overall dependency on fast response times, buffers from the FULL queue may be temporarily written on RAD (7212) from whence they can be retrieved when non-resident CASPER is awake and prepared to sort them into their final spectra. This buffering is the most important aspect of our ability to handle high data rates. It is accomplished very rapidly by writing (reading) the buffers to (from) the first available (full) disc sector starting at the current location plus two. By knowing the disc location at all times (1F1 and F3) the average access time for a write is 1-2 ms instead of 17 ms as would be expected without such knowledge. Since our disc buffer area is rarely more than 10% full, this short write access time is seldom degraded. After a FULL buffer is written to disc, it is returned to the EMPTY queue. Buffers read back from disc are placed in the FULL queue for sorting--up to a maximum of three entries. Beyond three, even if CASPER is awake, newly filled buffers will be written to disc.

I.4 Sort Driver

This interrupt is actually a part of non-resident--it is only enabled when CASPER is awake and is used to service as quickly as possible buffers which are filled directly via data I/O channels or are read from temporary disc buffering. Sorted buffers are returned to the EMPTY queue.

One additional use of a buffer from the EMPTY queue is made by a routine to create a long COMMAND/DATA chain for writing 32K to disc or 24 K to display disc. This routine is an extension of the non-resident functions and is only used by them.

In addition to the above buffer manipulating tasks, CASPER may be controlled by normal programs through a series of CAL4 commands.

These commands are entered into a CONTROL queue by one of the resident tasks and non-resident CASPER is immediately awakened.

As should be clear from the above descriptions, our use of fast real-time response is minimal. The use of resident tasks is limited to manipulating full buffers of 500 events each and moving them between the EMPTY, FULL, and EMR queues. These tasks were written to operate efficiently using interrupts and normal UTS I/O end action.

Only three entries into UTS are needed by CASPER. These are NEWQ, NEWQNWM, and T:GJOBSTRT (awaken non-resident CASPER).

II. Non-Resident Data Collection and Display

The non-resident portion of CASPER consists of about 2 K of code plus space as required for the 1 to 8 spectra (up to 32 K of core) being collected. Unlike the resident portion which is entirely interrupt and end-action driven, the non-resident CASPER has a polling structure subservient to data sorting done by the interrupt F4 sorting driver in the resident code. All buffers in the FULL queue or on disc are sorted before CASPER returns to sleep. This is possible because non-resident CASPER runs in master mode, thereby locking itself in core. This feature enforces absolute priority for data-collection, which is another distinguishing mark of our system.

Non-resident CASPER is also responsible for displaying the data when required (usually every 1.7 seconds), saving all updated spectra on permanent disc storage every 30 seconds (in case of system crashes), and responding to control requests entered into the CONTROL queue either by the CONTROL pushbutton switches (F1) or programmatically via the CAL4 instruction.

II.2.1 Control Request Handling

CASPER may be directed to perform the following functions:

STOP - Clean up all operations in progress and go to sleep for an extended period.

DIE - Clean up operations and terminate job.

RESTART - Restart from a previously stopped condition.

START/STOP DISPLAY - Turn ON/OFF the automatic displaying of updated spectra.

ERASE TAGS - Erase all litepen tags.

REWIND EMR - Rewind and restart EMR tape.

RESET SPECTRA - Reset all or selected spectra to zero.

The response time required of non-resident CASPER is dictated entirely by the human need to see updated spectra being displayed. Currently an update rate of 1.7 seconds is quite adequate.

III. Timing Considerations

The CASPER program, structured as described above, is regularly used (about 20 shifts a week) to provide for active data collection along with normal UTS operations. Our anticipated timing problems have not actually occurred, partly due to the conceptual design which reduces the need for a great deal of fast response--while replacing it with temporary disc storage.

In our case, the external interrupts do not appear to be inhibited by UTS for undue lengths of time. The time of our inner sort loop is between 8 and 20 μs per event depending upon the spectra being collected. The effect of high data rates is felt at about 25,000 events per second at which time approximately half of the Sigma 7 is available

At 40,000 events per second UTS is effectively locked out (10% or less) and beyond that 70,000 events/second have been collected. As the rate drops, UTS perks up and resumes normal activity with no difficulty.

Perhaps the most striking feature of the overall timing picture is the importance of knowing where the disc is at all times. This was driven home to us when position-measurement was inadvertently disabled (hardware) at one time with the result that 15,000 events per second required about 90% of the Sigma capacity, to the considerable agravation of all other users. This effect should be expected because of the change of average access time for each read or write from 1-2 ms to 17 ms.

IV. Problems Encountered

Our main problem encountered has been in the allocation and use of core memory by UTS. In particular, even though the upper 4 K of core is not known to UTS its allocation is handled as though it were, reducing the effective total core available to any given program. In other words, programs attached to the upper 4 K can acquire 4 K less dynamic or common core than normal.

Our second most serious problem results from the failure of UTS to swap jobs out while any I/O is in progress even though that I/O is fully monitor buffered. In our case, a plotting program uses all of core for up to 20 seconds even though its output is completely monitor buffered. This problem is especially severe since almost all of our data-base manipulation and analysis programs are large (up to 32 K) thus requiring a great deal of swapping.

Another problem which we face is that keyed UTS files do not release granules back to the system as records are deleted. In our instance of many large records (4½ K) this failure uses up disc space at an unprecedented rate.

Our final major problem is the very poor tape handling by UTS. This comes about partly because we have no operators. The Sigma 7 is operated solely by the users who frequently fail to obey precisely tape mounting instructions. UTS is particularly unforgiving of such failures.

The changes which we have had to make to UTS have been unexpectedly minimal--about 7 patch cards for required UTS real-time application and about 17 more for our own convenience (e.g. reducing wasted pages of printer output).

In summary, our experiences over the last few months with UTS-C01 have been gratifying. It has not only met but exceeded our original expectations and has required much less internal modification than anticipated. Its acceptance by our experimental staff has also been exceptional. Only twice since its beginning, January 1, 1973, have we gone back to an earlier system.

BNL REAL-TIME DATA ACQUISITION UNDER UTS

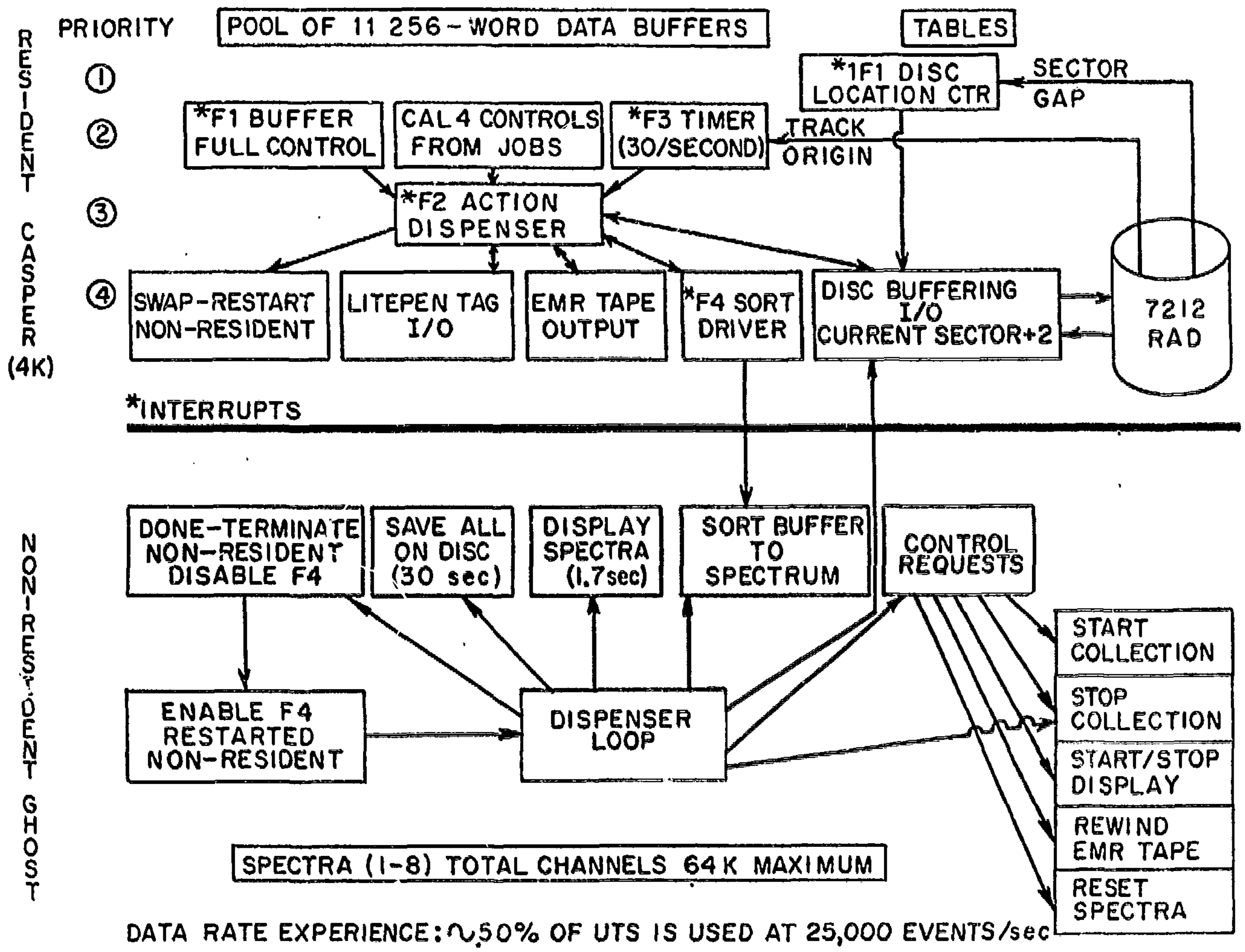

FIG. 1