CONF-870552--22

CONF-870552--22

DE87 011453

THE DESIGN AND PERFORMANCE OF THE PARALLEL MULTIPROCESSOR NUCLEAR PHYSICS DATA ACQUISITION SYSTEM, DAPHNE

L. C. Welch, T. H. Moog, R. T. Daly and F. Videbaek
Argonne National Laboratory, Argonne, Illinois, 60439

The ever increasing complexity of nuclear physics experiments places severe demands on computerized data acquisition systems. A natural evolution of these systems, taking advantage of the independent nature of "events", is to use identical parallel microcomputers in a front end to simultaneously analyze separate events. Such a system has been developed at Argonne to serve the needs of the experimental program of ATLAS, a new superconducting heavy-ion accelerator and other on-going research. Using microcomputers based on the National Semiconductor 32016 microprocessor housed in a Multibus I cage, CPU power equivalent to several VAXs is obtained at a fraction of the cost of one VAX. The front end interfaces to a VAX 11/750 on which an extensive user friendly command language based on DCL resides. The whole system, known as DAPHNE, also provides the means to replay data using the same command language. Design concepts, data structures, performance, and experience to date are discussed.

Introduction

The construction of a new superconducting heavy-ion accelerator, ATLAS, at Argonne National Laboratory introduced a new class of experiments whose complexity necessitated the creation of a new data-acquisition system. The new class of experiments are characterized in general by many more parameters per event (~100 vs ~10 previously), higher data rates and more complex experimental apparatus to debug. It was also expected that relatively more outside users would participate in the research program. All of these characteristics placed severe demands on the data-acquisition system and to meet them a new system was designed and created. The system is known as DAPHNE (Data Acquisition by Parallel Histogramming and NEtworking).

The combination of high-data rate, large-event sizes, complexity of experiments, and ease of use places orthogonal requirements on the system. An example of orthogonality is a generalized data-acquisition system that can be used for a complex arbitrary experiment and still be easy to learn for the new user. Another example is the fact that the incoming data has to be transformed, in general, to yield easily interpreted quantities, and yet the transformation is a costly process in terms of CPU usage and hence is orthogonal to the requirement for speed.

System Design,...Overview

The novel feature of DAPHNE is the use of parallel single-board computers (SBCs) to analyze independent events. At the time (early 1983) of system design available competitive 16-bit SBCs were based on the Motorola 68000, the Intel 286, and the National Semiconductor 32016. The 68000, at that time, did not have a floating point coprocessor and for that reason was not further considered. Based on preliminary considerations the Intel 286 was the original choice. Several factors eventually led us to the NS32016. These were: 1) difficulty in getting delivery of the 286; 2) the existence of good programming tools for the NS32016 including a symbolic debugger; 3) the similarity of the instruction set of the 32016 to the VAX; 4) a much larger capability for on-board memory (the 286/10 used only 64K and utilized bank switching); 5) the multibus addressing scheme required hardware modifications for the 286 but not for the 32016 and finally; 6) the source code for the 32016 monitor was available. For a system bus, both the VME bus and Multibus I would have sufficed for the needs of the project. Previous experience [1] with Multibus I and better vendor support (at that time) were our reasons t chose Multibus I. The combination of the the 32016 and Multibus I, while not the most popular choice today, has proven to be a very good choice from the point of view of price, availability, reliability, software development tools, and ability to meet the requirements.

The choice of the host computer, a VAX 11/750, was made because of the experience of the people involved in the project, its familiarity to the majority of the nuclear physicists who were to be users, its real-time response, the availability of public software, and the ability to share most of the code for replay purposes with an existing VAX 11/780.

The choice of a VAX makes available the VMS operating system features for use in software design. The command line interpreter [2,3] is used to guarantee identity of syntax between the operating system commands and DAPHNE commands. The HELP facility can provide online documentation not only about commands but also about DAPHNE objects and concepts. The MESSAGE utility is used to construct uniformly formatted and meaningful error messages. Other features, not obvious to the user, include global sections, event flags, the lock manager, priority adjusting, condition handlers, mailboxes, logical names, and subprocesses.

System Design,...Hardware

As part of the fundamental design (Fig. 1) of DAPHNE an attempt was made to use distributed and parallel processing to better provide the appropriate computing environment for solving each of several computational problems. In addition, to help solve the problem of a long set-up time, DAPHNE has the capability of supporting two experiments simultaneously. To read the events out of the CAMAC crates (CAMAC is a necessity because of the ubiquity of CAMAC among nuclear physics experimental programs) a device is needed whose primary attribute is the ability to get the data out of the crate quickly and into an environment where transformations, histogramming, linearizations and condition testing can be done more suitably. Any CAMAC time used in calculations at the level of forming the event is time lost for the acceptance of the next event, and hence causes more dead-time for the experiment. The device chosen to read the events is the "Event Handler" [4] (EH) created at Oak Ridge National Laboratory by D. C. Hensley. The EH is a programmable processor, part of which behaves as an auxiliary crate controller and is significantly faster than a competitive device, the MBD.[5] In DAPHNE it is supported by a Kinetics Systems (KS) Model 3989 [6] RS-232 crate controller connected to the host computer, a VAX 11/750. In order to avoid competition for the CAMAC dataway the data is transferred from the EH, via a front-panel cable to a KS Model 3841 FIFO buffer which provides derandomizing. Out of the FIFO the data is transferred from the experimentalist's electronic rack to the computer room over a differentially driven 25 twisted pair cable. The cable has a driver (optically coupled to the FIFO) at the rack end and a receiver coupled to an Event Distributor (ED). The receivers (one for each experimental station) and the ED are housed in a CAMAC crate to provide physical housing and power but no electronic CAMAC protocol is used by either. The ED controller is switchable to select which of the experimental racks will serve the primary experimental data source and which is to serve the set-up experiment. From the ED the data flows to the individual Event Processors (EP) housed in the Multibus I cage, via FIFO buffers on each processor. There is one Multibus cage for the main experiment and one for the set-up. Each processor in the Multibus cage is a National Semiconductor 32016. From the EP the results of the event sorting, in the form of VAX increment instructions, are sent to the VAX 11/750 via an intelligent DMA Unibus Interface called the Event Processor Interface (EPI). Again, there is a separate interface for the main experiment and for the set-up. Further details of the hardware are described elsewhere.[7]

System Design,...Software

There are three programmable components of the hardware -- the Event Handler, the Event Processors, and the VAX. To the extent possible the design goal was to divorce the user from having to deal with the Event Handler and the Event Processors and to deal with the VAX only through the familiar VAX/VMS Digital Command Language (DCL) interface provided by the Command Language Interpreter (CLI). To a large extent this goal has been successfully met. If a user requires no special transformation of the incoming event to form "pseudo-parameters" then a code generator is used to produce the code for the event processors automatically from the user-defined DAPHNE objects (histograms, windows, linearizations, etc.) and downloaded transparently. If a "user transformation" is required then the user can write a subroutine in FORTRAN77 and the resulting object module is downloaded and its address made known to the supervisor code in the NS32016.

The code for the Event Handler is created via a cross assembler and is created for each experiment by the experimentalist. Several example programs exist and users have proven adept at learning the techniques.

On the VAX all the data structures are contained in global sections which can be accessed by programs initiated by DCL-like commands. The programs typically are activated, modify the global sections, and then exit. There are no software limits to the number of structures that can exist although defaults are chosen which the user can override. The relationships among the more significant data structures are depicted in Fig. 2. The user defines the structure of the event types, the quantities to be histogrammed, conditions, windows, linearizations, etc. From these definitions the code to accomplish these tasks is generated for the front-end processors, in the case of real time, and for the VAX in the case of replay. The software concepts are discussed further in Ref. 8. A noteworthy feature of the system is the creation by the EPs of buffers of VAX machine code instructions (INCs) which increment the relevant histograms. These buffers are transferred to the VAX and then merely executed thus minimizing the participation of the VAX on an event-by-event basis.

Another key issue in the software design is the nature of the event structure. Multiple-event types must be possible as well as variable-length events. Modern experiments produce variable-length events through two mechanisms, the first and most common is through sparse data scans where a pattern register conveys the detector identification information. The second is the Q scan where the detector identification information is contained within the data words. The DAPHNE event data structure is general enough to cope with both but has not yet been used for Q scans. Further details are contained in Ref. 9.

Of immense help is a software "event simulator" running on the VAX which produces an event stream identical in format to that coming from the front-end processors. This enables the development team to debug most of the software without a data-acquisition computer.

An extensive online HELP library is available fo the user and a complete set of user documentation is available in draft form.

Performance

DAPHNE has been used by experimentalists for ove 1-1/2 years and has acquired data for over 30 experiments and 3000 hours of beam time. In addition to the real-time data acquisition it has been used extensively to replay data, not only at Argonne but also at three user locations to which DAPHNE has been exported. As a replay system, in comparison with the previous replay system, LISA, improvements in CPU tim usage are typically a factor of ten although factors of 20 and 3.5 have also been seen depending upon the nature of analyses.

As an acquisition system there are three potential bottlenecks in the system, and depending on the nature of the experiment one of the three may manifest itself. The first potential bottleneck is th Event Handler which may appear for very small events at high rates (e.g. monitor counters). The measured deadtime in the EH is 2.1 μsec/parameter and 1.2 μsec/event. The deadtime associated with the event is the same regardless of the number of parameters read in the event. The deadtime associate with the parameter is the time needed to perform the CAMAC read (1.2 μsec) and then to transfer the data t the FIFO. Thus the primary cause of the deadtime associated with each parameter is inherent in the CAMAC dataway cycle time. Also needed in the EH program is a loop to wait for any ADCs or TDCs to complete conversion. Hence for an ADC with a 5 μsec conversion time the EH would take 8.3 μsec to process one event, dictated mainly by the CAMAC cycle time an the conversion time of the electronics. Hence, at an allowed deadtime of 10%, 50% or 90%, the maximum observed random-event (or parameter) rate through the EH would be 12.2 KHz, 61.0 KHz, or 109 KHz repectively. For an event with 20 parameters the maximum observed parameter rate for deadtimes of 10%, 50% and 90% would be 43.1 KHz, 216 KHz, and 388 KHz respectively. Since each parameter is 2 bytes, the byte rate would be twice as high.

Published values [10-12] for dead times for the MBD show some scatter but are typically 8 μsec/event and 5.5 μsec/parameter, significantly higher than for the event handler. The performance difference manifests itself as an improvement in the data rate capability of the EH as a function of dead time as illustrated in Fig. 3. A potential disadvantage of the EH is its inability to control more than 2 crates while the MBD's limit is 7. So far, because of the density of modern CAMAC modules, this has not been a problem.

A second bottleneck could occur in the Event Processors. Each processor has approximately the CPU power of the host VAX 11/750, and to date no experiment has been run for which 3 EPs were a limiting factor. It is expected, because of other aspects of the system, that the EPs will only be a bottleneck if the user transformation is exceptionally long and/or complicated. If this situation were to occur the addition of more EPs would solve the problem. The Multibus cage can hold 7 EPs and the system design allows the daisy chaining of multiple Multibuses together although this has not been done to date. We are confident that the lack of sufficient CPU power in EPs will never be a problem regardless of the expected complexity of the user's transformation.

The third potential bottleneck is the interface to the VAX. This bottleneck, in fact, tends to be the one first encountered. The transfer rate from the Multibus to the Unibus has a measured maximum of 420 Kbytes/sec. One reason for this transfer rate limitation is that the data is moved from the Multibus to the VAX a byte at a time by a Z80 direct memory access controller. However, this transfer rate is well matched to the host in the following sense: When data is being transferred to the VAX at the maximum rate and the data stream consists of only buffers of INCs for the VAX to execute then the process which executes those INCs consumes 36% of the available VAX CPU power. The 36% represents 30% used in executing the increment instructions and 6% buffer overhead (including interrupt service time). Each buffer contains approximately 2000 increment instructions. Since DAPHNE is a two user system the transfer rate is well matched to the CPU power of the 750.

Inasmuch as the transfer rate limitation is the most restrictive bottleneck, several options have been made available to the user to more efficiently use the available bandwidth. The recording of the event data may be turned on or off, but since a raw parameter is 2 bytes while the INC is 6 bytes, a more productive option is to histogram a small percentage of the data while taping all of it. As an example assume an event with 20 parameters from which an INC is formed from each parameter and event-mode recording is turned on. The maximum event rate through the interface into the VAX is 2.6 KHz. If the user turns off event-mode recording the rate only goes to 3.5 KHz whereas if the user chooses to histogram only 10% of the data (while still recording all of it on tape) then the event rate goes to 8.1 KHz, "independent" of the complexity of the user transformation. The user also has the ability to choose not to record every event on an event-by-event basis based upon user-defined conditions. The user may also stop all further processing of an event based upon a KILL condition, hence curtailing the formation of any INCs.

Future Work

As fast as facilities are added users demand others. Currently being implemented is the ability to produce an output tape while replaying. The output tape, in DAPHNE format, would consist of a subset of the original events and/or the user created pseudo-events. Another facility to be implemented is a "multi-window" illustrated in Fig. 4. A pseudo-parameter is created whose value is equal to the domain of the window into which the event is histogrammed.

A port to a μVAX II will be done within the year. The software port should be trivial but our current hardware interface has a UNIBUS device. The μVAX uses the Q Bus and modifications to the driver, dictated by the use of a UNIBUS to Q-BUS converter, may have to be done.

A PAUSE/RESUME capability needs to be created so that tempo ry stops during a run do not create separate f les on the output tape.

Some consideration has been given to using the parallel processors during replay. The idea is attractive and some improvement in performance could be expected. However, in the present architecture the overhead in moving buffers from the VAX, where the event tape is read, to the EPs and then receiving the INC buffers does not promise a large benefit for the typical data analysis. One can certainly imagine situations, such as very complicated transformation, where the use of the parallel processors would help, and we will probably implement the feature but it has low priority presently.

Summary

In summary DAPHNE has met the needs of the experimental program at ATLAS and provides a sophisticated, generalized, fast, and easy to learn data-acquisition/replay program. The limits in terms of data-acquisition rates are determined solely by the hardware.

Work supported by the U. S. Department of Energy, Nuclear Physics Division, under Contract W-31-109-ENG-38.

References

[1] R. T. Daly, J. R. Haumann, M. R. Kraimer, F. R. Lenkszus, W. P. Lidinsky, C. B. Morgan, L. L. Rutledge, P. E. Rynes and J. W. Tippie, "Data Acquisition and Control System for the IPNS Time-of-Flight Neutron Scattering Instruments," IEEE Trans. Nucl. Sci., vol. NS-26 (4), 4554 (1979).

[2] Digital Equipment Corporation, "VAX/VMS," vol. 1-10, (1986).

[3] R. V. Poore, D. M. Barrus, G. Cort, J. A. Goldstone, L. Miller and R. O. Nelson, "A Data Acquisition Command Interface Using VAX/VMS DCL," IEEE Trans. Nucl. Sci., vol. NS-32 (4), 1290 (1985).

[4] D. C. Hensley, "The Event Handler II, a Fast, Programmable, CAMAC-Coupled Data Acquisition Interface," IEEE Trans. Nucl. Sci., vol. NS-26 (4), 4454 (1979).

[5] Bi Ra Systems, Inc., 2520-D Pan American Northeast, Albuquerque, NM 87107.

[6] Kinetic Systems Corporation, 11 Maryknoll Drive, Lockport, IL 60441.

[7] R. T. Daly, T. H. Moog, and L. C. Welch, "DAPHNE...The Design of a Parallel Multiprocessor Data Acquisition System," IEEE Trans. Nucl. Sci., vol. NS-32 (4), 1379 (1985).

[8] L. C. Welch, R. T. Daly, D. Leucks, T. H. Moog, and J. Stewart, "DAPHNE...A Parallel Multiprocessor Data Acquisition System-Software," IEEE Trans. Nucl. Sci., vol. NS-32 (4), 1405 (1985).

[9] L. C. Welch, "DAPHNE...A Parallel Multiprocessor Data Acquisition System for Nuclear Physics," IEEE Trans. Nucl. Sci., vol. NS-32 (1), 238 (1985).

[10] J. F. Aman, "Frontend Event Selection with an MBD Using Q," IEEE Trans. Nucl. Sci., vol. NS-28 (5), 3843 (1981).

[11] N. R. Roberson and S. E. Edwards, "Interface for the TUNL VAX Data Acquisition Facility," IEEE Trans. Nucl. Sci., vol. NS-28 (5), 3834 (1981).

[12] N. R.Roberson and C. R. Gould, "A Dual-MBD VAX 11/780 Data Acquisition Facility," IEEE Trans. Nucl. Sci., vol. NS-32 (4), 1447 (1983).

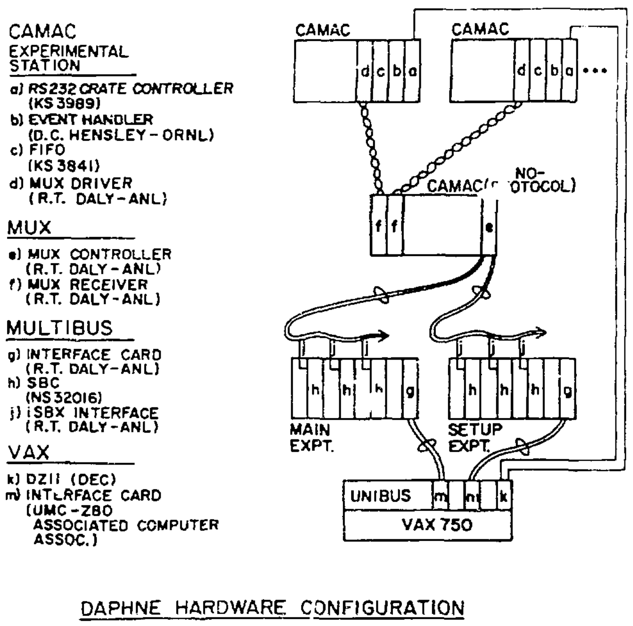
CAMAC
EXPERIMENTAL STATION
a) RS232 CRATE CONTROLLER (KS 3989)
b) EVENT HANDLER (D.C. HENSLEY - ORNL)
c) FIFO (KS 3841)
d) MUX DRIVER (R.T. DALY - ANL)
MUX
e) MUX CONTROLLER (R.T. DALY - ANL)
f) MUX RECEIVER (R.T. DALY - ANL)
MULTIBUS
g) INTERFACE CARD (R.T. DALY - ANL)
h) SBC (NS 32016)
j) iSBX INTERFACE (R.T. DALY - ANL)
VAX
k) DZ11 (DEC)
m) INTERFACE CARD (UMC-Z80 ASSOCIATED COMPUTER ASSOC.)
CAMAC
CAMAC
CAMAC (NO-PROTOCOL)
MAIN EXPT.
SETUP EXPT.
UNIBUS
VAX 750
DAPHNE HARDWARE CONFIGURATION

Figure 1 The DAPHNE hardware configuration.

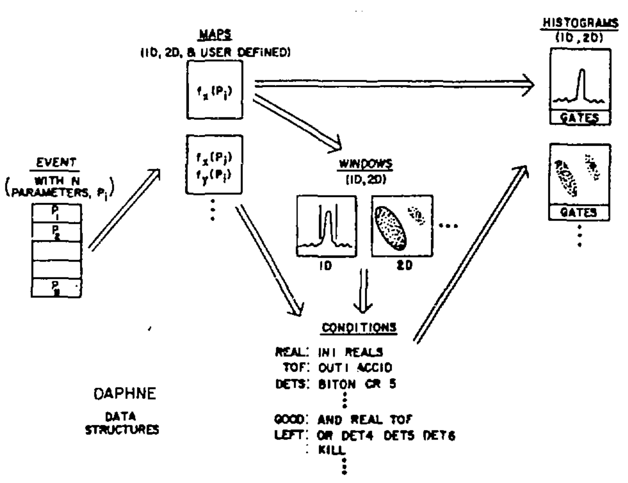

Figure 2 The logical relationships among the major DAPHNE data structures.

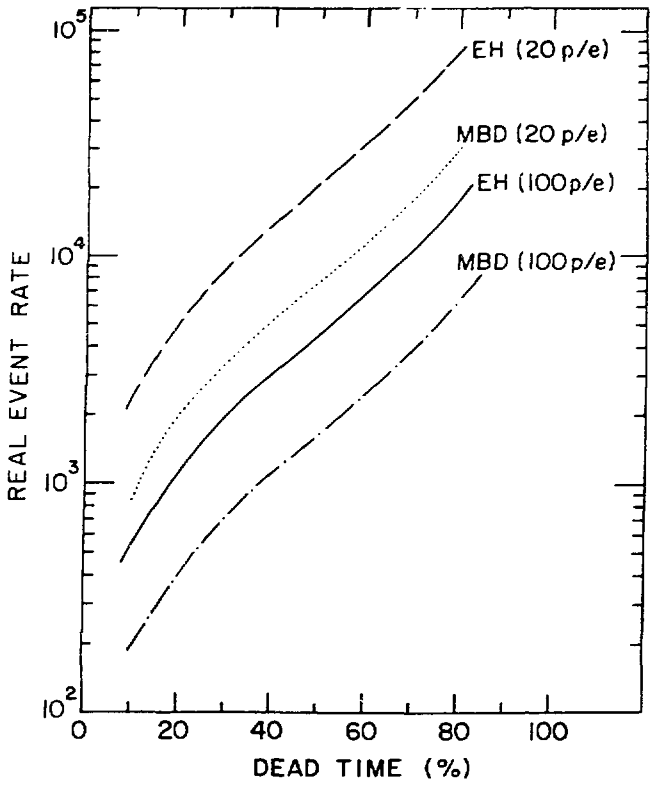

Figure 3 Performance comparison between the MBD and the EH for a 20-parameter event and a 100-parameter event.

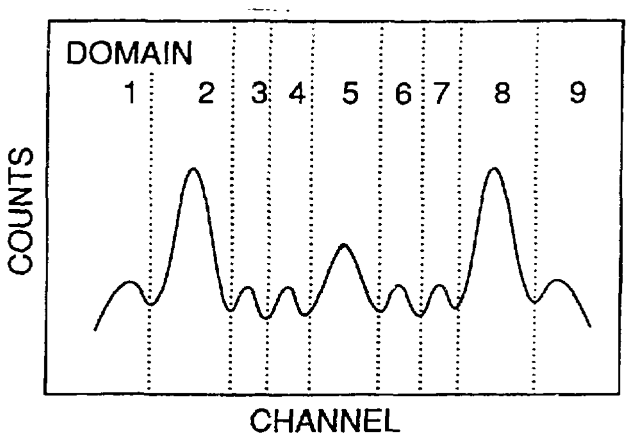

Figure 4 A "multiwindow" establishes a pseudo-parameter which is equal to the domain within a histogram.