ELSEVIER

Nuclear Instruments and Methods in Physics Research A 483 (2002) 830–832

NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A

www.elsevier.com/locate/nima

A camac data acquisition system based on PC-Linux

R.V. Ribas

Laboratório Aberto de Física Nuclear, Departamento de Física Nuclear, Universidade de São Paulo, CP 66318, 05315-970 Sao Paolo, SP, Brazil

Received 11 July 2001; accepted 11 August 2001

Abstract

A multi-parametric data acquisition system for Nuclear Physics experiments using camac instrumentation on a personal computer with the Linux operating system in described. The system is very reliable, inexpensive and is capable of handling event rates up to 4–6 k events/s. In the present version, the maximum number of parameters to be acquired is limited only by the number of camac modules that can be fitted in one camac crate.

PACS: 07.05.Hd; 29.85.+c

Keywords: Data acquisition; Computer data analysis

1. Introduction

Camac is a very old standard to interface instrumentation to computer systems, but is still widely used, mainly due to the large number of commercial modules available and easy interface to the widely used personal computers (PC). In many newly developed systems, specially when used with complex detection instruments, other standards, such as VME, VMX, that allow faster readout of data and distributed computer processing are often used [1], but these systems are very expensive and difficult to maintain. Due to the complexity of these systems, they are normally dedicated to a specific detector system and are necessary only when high event rates and extensive on-line data processing are needed. In many situations, it is important to have a system that is flexible for use with different experimental setups, easy to maintain by an average skilled computer programmer, and uses almost only commercial modules. One condition to be fulfilled by this system is to have the event definition and the action of the system on the occurrence of an event programmed by the experimenter in an easy and safe way. The recent development of Linux as a reliable operating system for pc, implementing multi-task and multi-user capabilities, has also given a good environment to implement such systems. In this paper, we describe a simple and versatile data acquisition system based on these premises.

2. The front-end interface

The front-end camac instrumentation is interfaced to the ISA bus of a pc by a CAEN c111a

E-mail address: ribas@if.usp.br (R.V. Ribas).

0168-9002/02/$ - see front matter
PII: S0168-9002(01)01950-7

camac controller and the A151-A turbo board. The main characteristic of this controller and its interface to the pc, is that they allow very fast data transfer to the computer using only two 16-bit io-port operations. The device driver we developed can be easily ported to other controllers and interfaces. We have also developed one for the DSP-6002 controller, that is very efficient, even if his controller has an 8-bit interface, needing about 5 io-port operations to perform one 16-bit NAF read. This driver also includes a set of instructions programmed by the user, to be executed in a loop at each event reported by the camac modules. In the present version, only one type of event can trigger the system, using camac LAM signals. A camac module that receives at least one parameter that is mandatory in all events is programmed to send LAMs to interrupt the computer. It is very desirable that the multi-input modules used implement a pattern word that can be used by the driver program to read sparse data in the module, skipping non-triggered channels. This pattern is present in many ADCs and TDCs used in Nuclear Physics experiments, like the Silena 4418 series and the Phillips 7164 and 7186. This method of recognizing events is probably the main limitation of this system. A very simple event trigger module could be developed to handle more complicated trigger situations. As our previous data acquisition system was based on the Oak Ridge Event-Handler module [2,3], we have implemented in the driver for our c111a crate controller, an emulator for the set of machine instructions used in that processor. Only a few modifications to the compiler called ADAC [4], that is used to produce the executable code to the original EH module, have to be made in order to implement it to the new situation. As the actual use of this language in the pc processor permits much more flexibility, new instructions could be easily implemented in the compiler. This solution is very handy, since it allows the flexibility necessary for a general-purpose data acquisition system and maintains the same functionality of the programs developed for the EH by the experimenters at our laboratory. With this police, a more specific driver (or compiler) could also be developed to attend to specific applications. At the occurrence of an interrupt, the event processor program is executed and the event parameters are read and formated and then placed in a fifo buffer. The user program can asynchronally call the driver and check for full data buffers in the fifo and read it.

3. Acquisition and on-line monitoring modules

The basic software for the system is composed of several independent processes that communicate to the master or manager process, and also to the other modules, receiving and sending messages and data. The standard Linux interprocess communication, synchronization and data exchange resources, like messages, semaphores and shared memory are employed. Since many parts of the modules used in this system originates from programs written for the VAX-VMS operating system, almost all code was written in fortran, and the Portland Group pgf77 fortran compiler for Linux [5], that includes most of the DEC extensions to the fortran-77, was employed. The routines dealing with the specifics of the operating system were written using the standard c-compiler available on Linux. The master process receives commands from the user and sends it to the process to which it is directed. The process that communicates with the front-end camac and to the driver, can load the EH emulator instructions list to the driver, start and stop the acquisition runs, accumulate the events into an event buffer and send the event buffers to the process that will sort the events into 1D or 2D spectra and also to mass storage media. The event lists can be recorded both in hard disks or magnetic tapes. Events are processed by the sorting program in an as-much-as-I-can basis. For this procedure to be efficient, the event buffers are stacked into a software-fifo in the sorting program. If the even-buffer fifo is full, the next incoming buffer is discarded. The front-end process also records statistics about the data acquisition run, like the total number of events acquired, time of acquisition, percentage of tape used, etc. For the same user command, the sorting program will also report the number of sorted events and the number of bad events not

recognized by that program. Pre-set time can also be enabled for the runs and a bipper can warn the experimenters if event buffers do not come within a minimum pre-defined time interval. In the present version, the sorting program can produce up to 16 Mbytes of 1D and 2D histograms in ram memory, with 16 or 32 bits for each channel. This module is what is called a CHIL-processor, since it interprets CHIL-language instructions. Comprehensive HIstograming Language (CHIL) is a compiler developed by WT Milner [4], that is used in many on-line and off-line histograming programs. The core memory histograms are shared with the display, analysis and manipulation program named DAMM, also written by Milner [4]. DAMM processes are not linked to the rest of the system, and any number of DAMM processes can have access to the on-line histograms. On-line histograms can also be stored on disk at any time for posterior analysis.

4. Benchmarks

All tests were done using a P-133 cpu running the Conectiva Linux Distribution [6]. In a typical event occurring in our system about 4–10 parameters are read out of a much larger number of defined ones. Typical converting times of camac ADCs and TDCs range from 5 to 100 μs. The driver takes about 15 μs from the time the LAM was posted, in order to start transferring data from the crate. If the LAM can be generated before the end of conversion of the ADCs, the converting time can be superposed to this overhead time. With the CAEN turbo interface, about 4 μs is spent for each 16-bit read operation. With a moderate event rate (2–3 k events/s), and typical sorting requests, the fraction of computer time spent by the sorting program or by the display program almost does not affect the throughput of events to the computer.

5. Conclusions and planned improvements

The system is starting to be used on a regular basis as a secondary data acquisition system in our laboratory. It is very stable, even in high event rates and complex histograming conditions. The driver developed can handle only one crate controller, but there should be no major difficulties to port it to be used with more controllers, since the CAEN A151A can interface up to 16 crate controllers. With typical event rates (2–3 k/s) all acquired events were histogramed by the CHIL processor. Similar systems using the DSP-6002 crate controller are also being used, with quite good performance, despite their 8-bit interface. We are also planning to build an event trigger module that will be able to send LAMs to the computer previous to the conversion completion by the modules and that will also be able to generate a pattern bit word classifying different types of events. This module will also generate a veto NIM signal for the duration of time from the event trigger to the end of the event transfer by the computer, which can be used to measure the dead time of the system. This system could also be adapted to be used with other histograming and event processing languages like NEO [7]. The use with a set of modules that incorporates the FERA bus to the camac can also be foreseen. As a complement to this system, we have also ported to the Linux operating system, many programs from Upak [4], for off-line analysis of the data produced. Source and executable codes for all these programs can be obtained by contacting the author, ribas@if.usp.br.

References

[1] C. Rossi-Alvarez, Nuovo Cimento 111A (1998) 601.
[2] D.C. Hensley, IEEE Trans. Nucl. Sci. NS-26 (1979) 4454.
[3] R.V. Ribas, Annual Report, Nuclear Physics Department, University of São Paulo, 1996, p. 64.
[4] W.T. Milner, Upak documentation, Oak Ridge National Laboratory.
[5] Portland Group, www.pgroup.com.
[6] Conectiva Linux, www.conectiva.com.br.
[7] B. D'Avanzo, M. De Poli, G. Maron, A. Negro, G. Parlati, D. Pascoli, R.V. Ribas, S. Scanferlato, G. Vedovato, Proceedings of the Real Time'91 Conference, Jülich, Germany, 1991.