ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di Milano

INFN/BE-90/08
26 Luglio 1990

W. Bragagnini, P. Guazzoni, M. Pitalieri and L. Zetta:

COMPUTATIONAL LOGIC UNIT FOR A MICROPROGRAMMED DATA ACQUISITION SYSTEM: AN EVALUATION PROTOTYPE

Servizio Documentazione
dei Laboratori Nazionali di Frascati

INFN - Istituto Nazionale di Fisica Nucleare
Sezione di Milano

INFN/BE-90/08
26 Luglio 1990

W. Bragagnini, P. Guazzoni, M. Pitalieri, L. Zetta.

Computational Logic Unit for a Microprogrammed Data Acquisition System: an evaluation prototype

Contribution to
Euromicro90 - Hardware and software in System Engineering. Sixteenth Symposium on Microprocessing and Microprogramming, Amsterdam, The Netherlands, August 27-30, 1990

TO BE PUBLISHED ON MICROPROCESSING AND MICROPROGRAMMING

Computational Logic Unit for a Microprogrammed Data Acquisition System: an evaluation prototype

Walter Bragagnini(*), Paolo Guazzoni, Maurizio Pitalieri(+), Luisa Zetta
Dipartimento di Fisica dell'Università and Istituto Nazionale di Fisica Nucleare - via Celoria 16 - 20133 MILANO - Italy

(*) present address: Aeritalia-Div. Avionica - Ronchi dei Legionari (Go)
(+) present address: Harris - Scientific Calculations - Agrate B. (Mi)

ABSTRACT

This paper describes an evaluation prototype of a computational unit designed to compute a typical algorithm used in nuclear electronics to identify in mass and charge light reaction products, and the test system, computer aided, employed to verify its performance. The computational unit is based on the VLSI AM29325 Floating Point Processor and uses bit-slice components and microprogrammed architecture.

KEYWORDS

Microprogrammed applications, function computation with polynomial expansion, data acquisition with on-line processing, nuclear electronics.

1. INTRODUCTION

In a previous work(1), we have described the first evaluation prototype of an advanced data acquisition system, based on a microprogrammed hierarchical architecture and on bit-slice components. This system was designed in order to solve a typical problem of nuclear electronics: the on-line processing of data acquired in digital form by an electronic chain of detectors used in nuclear experiments. The present paper describes an evaluation prototype of computational logic unit, which has to be employed with the system described in ref. 1, for the on-line computing of special algorithms. In section 2, the physical constraints necessary to build the system,are exposed. A general description of the unit, together with design and building specifications, is given in section 3. Finally the description of the testing system and experimental results obtained are presented in section 4.

2. PHYSICAL CONSTRAINTS

A problem arising in low-energy experimental nuclear physics is identification of charged reaction products and measure of their energy spectra. A method, in order to solve this problem, consists in using a telescope of two silicon surface barrier detectors, the first one (DE) crossed by the reaction products, leaving only a part of their energy on it, the second one (E) which stops the reaction products, leaving their remaining energy on it. Different algorithms have been proposed (2), in order to use these pairs of data (DE and E) to construct a function allowing identification of light reaction products. One of the most suitable is the so called "power-law" (2,3)

$$PIF \div (E + DE)^x - E^x$$

where DE and E are the numerical values, (discretized by the analog-to-digital converters of the data acquisition system), corresponding to the energies lost in the two detectors, and x is a real number, having a precise value depending on the product mass and ranging between 1.40 - 1.80. PIF is the special function for particle identification, characterized by one peak for each particle detected. Different methods have been used for the computation of PIF.

In fact, the computation can be made on-line by means of analog circuits (2,3) or off-line, using computers, analyzing the collected data after the experiment is done(4).
Our approach to the problem was to compute on-line the algorithm of the power-law in digital way. For this purpose we chose a hierarchical microprogrammed architecture for the data acquisition system of ref. 1 and designed to build a Computational Logic Unit (COLU), able to perform on-line the computation of the power-law algorithm. To solve on-line an algorithm like the power-law, it was necessary to find a numerical method to compute powers and to choose a proper processor able to perform arithmetical operations at high speed. The main characteristics of the evaluation prototype and the test system have to be:
1) capability to compute the algorithm chosen
2) precision and reliability in order to ensure a correct working mode
3) possibility of verifying the goodness of numerical analysis methods used to solve the chosen algorithm.

3. COMPUTATIONAL LOGIC UNIT

As previously said, in order to compute the power-law, we chose to design and build a special arithmetical logic unit, based on bit-slice components and on a Floating Point Processor (FPP), using algorithms derived from the numerical analysis. Our previous experience on analog devices allowed us to solve the problem in a simple way.
In fact for the identifier of ref. 3, the problem of computing the power-law on line was solved by building an analog computer, based on logarithmic and antilogarithmic operational amplifiers (fig. 1). This method obviously has all the advantages and all the disadvantages of an analog circuit. The first ones consist in a rather good speed (~ 40 μs for the power-law computation), the simplicity of the circuit, easy to handle and transport, and the capability of working on-line. With regard to the second ones, there are impossibility of changing the algorithm, and electronic noise to be added to the signal at each stage of the computation, thus worsening final resolution of the computed function.
Another approach is the off-line analysis of the collected data, with big computers. In this case there are two main disadvantages: impossibility of following the experiment during the data collection, and necessity of practically doubling the time needed for the experiment, taking into account that necessary for the off-line data reduction. The principal advantages are: possibility of changing the algorithm and better resolution, due to absence of electronic noise in the function computation.
The method chosen by us, adds up the advantages of the two methods leaving out the disadvantages. In fact a computational logic unit is a quite simple circuit, fast and allowing the changing of algorithm, without modifying the hardware: in fact it is enough to change microprograms. Moreover, no noise is added to the signals, due to the computation performed in digital way.

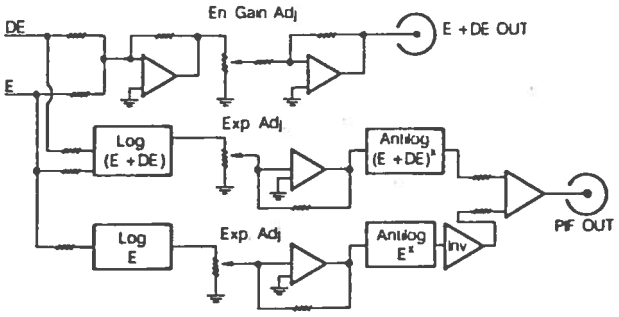

Fig. 1 - Analog computer schematic

3.1 Numerical Analysis

We have now given sufficient consideration to introducing the general problem of the computational unit, that we may turn our attention to the algorithm to be chosen for the digital computation of the power-law.
It is well known that for a numerical computation of a function, its expansion in series can be employed.
In general the series are infinite, but only a finite number of terms can be computed in practice, giving only an approximate value of the true one.
With regard to the power-law computation, our decision was:
1) expansion in orthogonals set of functions
2) a truncation of the infinite series so that the remainder will be equal to the lowest number that can be represented in DEC single-precision floating point format (32-bit) (5).

To satisfy the previous conditions, we chose the Chebyshev polynomials, a set of orthogonal polynomials more rapidly converging than the series in power, defined by the following relations

$$T_n(x) = \cos(n \cdot \arccos(x))$$

$$T_n(x) = \sum_{i=o}^{INT(n/2)} (-1)^i \binom{n}{2i} x^{n-2i}(1-x^2)^i$$

They can be computed by means of the recursion formula

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

Moreover this polynomial only has real simple zeros and all these zeros lie in the interval (- 1, + 1).
Now in the following, we shall show how the problem of computing the power-law, using the Chebyshev polynomials, was solved.
We can represent a power as

$$A^C = 2^{C \cdot \log_2 A}$$

Besides

$$\log_2 A = \ln A / \ln 2$$

in which ln2 is a constant and lnA can be represented by means of an expansion in Chebyshev polynomials

$$\ln A = \sum_{n=0}^{K} a_n T_n(x)$$

The expansion was truncated at 12^{th} term, following the criterion previously exposed, assuring a final accordance better than 1/100000, between the computed value and the "true" one (in this case the "true" value was the result of the computation directly obtained by using the function LOG on a VAX computer).
Finally we can express

$$A^C = 2^{INT[(C/\ln 2)\cdot \ln A]} \cdot 2^{\{(C/\ln 2)\cdot \ln A - INT[(C/\ln 2)\cdot \ln A]\}}$$

i.e.

$$A^C = N \cdot 2^q$$

The first term of the product (N) can be easily computed, because it is a power of 2, with integer exponent. The second term can be approximated, using the Chebyshev polynomials

$$2^q = \sum_{n=0}^{K} b_n T_n(x)$$

In this case, fewer terms were necessary to obtain the required precision.
To summarize, the use of Chebyshev polynomials and the formalism introduced allow to compute a power as finite number of algebraic operations, in particular a small set of simple operations.

3.2 Mathematical Processor

On the basis of the results reached in the previous section, we shall now design the hardware of the computational unit. Since we chose AM2900 bit-slice components for the first prototype of ref. 1, we have now chosen a 32-bit floating point processor, from the AM29300 family. AM29325 (6) is a single VLSI integrated circuit which performs single cycle addition, subtraction, multiplication and conversion, using either the single precision IEEE and DEC format. In this way, using AM2900 and AM29325 devices, we have the possibility of tailoring them to our specific applications.

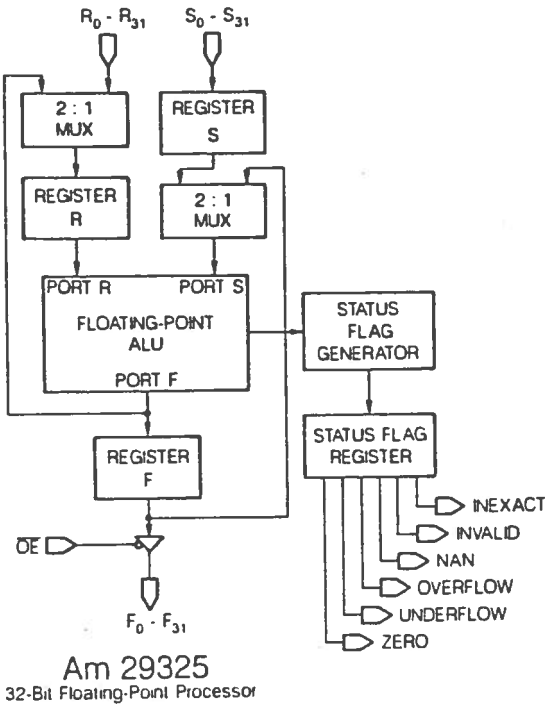

Fig. 2 - Schematic of FPP

The AM29325 (fig. 2) has a 3-bus, 32-bit architecture with two input buses (R,S) and one output bus (F), and a high speed floating point ALU. This configuration provides high I/O bandwidth, allows access to all buses and affords a high degree of flexibility when connecting this device in a system.
With this device, the division has to

be implemented, using a modest amount of external hardware and a special 15-cycle software. This microprogram compute a division C=A/B as C=A·(1/B), in which the reciprocal can be evaluated by using a technique based on Newton-Raphson method for obtaining the roots of an equation.
Before using FPP, we made a simulation code of its behaviour, starting from the specifications. The simulation code consists of about 3200 statements of Pascal language.

3.3 Design and Building

To verify the goodness of the algorithm developed for the power-law computation and of the FPP adopted, we designed, built and tested COLU, the Computational Logic Unit, able to perform the computation of the power--law using FPP.
Its architecture is very simple (fig. 3), also because it is supported by a Control Unit (CONTRO) for the proper timing generation and by a host computer, through a special RS232 interface (INTERF), ad hoc built, as will be explained in section 4.1.
The timing signals come from CONTRO and the data to be processed from the INTERF to which the result is sent after each computation.
Due to the fact that the DEC format of FPP differs from the Standard VAX format (in the two cases the 16 MSBs are swapped with respect to the 16 LSBs for the floating point numbers), the bus S was properly cabled, in order to obtain the swapping. For the R and F buses, it was necessary to use a special circuit to realize it. In fact through these two buses flow not only FP numbers, but also integers for which the two standards are the same.
The input data, coming from INTERF are bufferized in two latches. In this way the interface can receive new data, while COLU is working.
The data of the S-bus arrive to the processor without any changing, while the one of the R-bus can be "swapped", if FP number, or used to obtain the first value K to computing the reciprocal with the recursion formula.
In this way as input for FPP we can have: on the R-bus the number R or K; on the S-bus the number S or the number R after the swapping.
The result and the status word in output from FPP, are memorized in two registers (F-bus), again to speed-up the process, allowing COLU working during the interface transmission.

The COLU result, if FP number, is properly swapped on the F-bus. In this way COLU can operate, on the input data, all operation allowed by FPP, including the division.
With the COLU architecture, we can verify the working mode of FPP and the correctness of all its characteristics in simple way. As previously said, the simple architecture used for COLU, without own memory registers for the partial results and EPROM to store the microprograms for the whole power-law computation, brings, as a consequence, the necessity to use a host computer to memorize partial results and some programs for the computation.

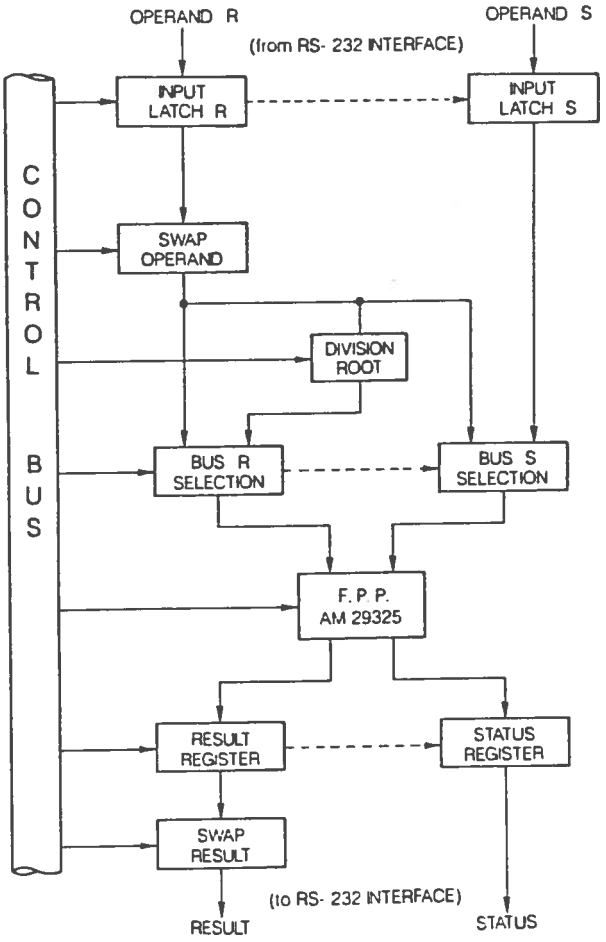

Fig. 3 - Schematic of COLU

4. PERFORMANCE

4.1 Testing System

Hardware

As briefly explained in the previous pages, to test COLU in order to verify the possibility to use FPP and the formalism (Chebyshev polynomials) in the final system, we have employed a special apparatus (fig. 4), ad hoc built and computer aided.
It consists, besides COLU, of the special RS232 interface (7) and of the unit for controlling communications between COLU and INTERF, and for storing microprograms.
To check the design of the apparatus, we have employed a simulation code able to verify the computational correctness of COLU, of the algorithms and of the microprograms. This code consists of about 5500 Pascal statements.

Following the block diagram of fig. 4, a general description of the test system and of its working mode can be given.
The general architecture is typical of a microprogrammed system (8) and is composed, as stated before, of the three different units, INTERF, COLU, CONTRO, connected by means of a proper bus. INTERF is bidirectional, because it has to receive from the host computer command codes and data, and to send to it the results of the computation and the status code of the whole system.
The computer code, used to control INTERF, has to satisfy not only the proper computation on the data, but also the proper two-way handshaking of the system. It must also accept from the computer a special character to give a general reset to initialize the different units.
We have also to notice again that the

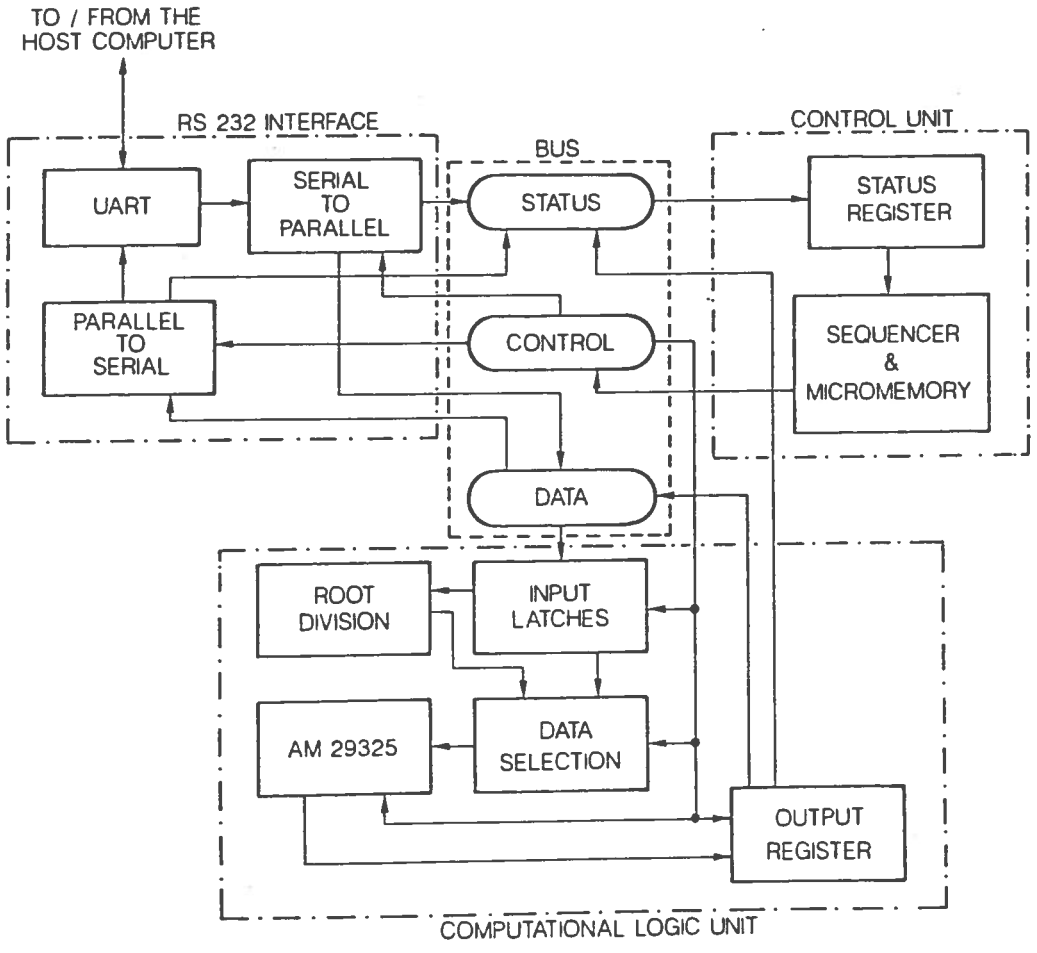

Fig. 4 - Schematic of the Test System

system works on the data sent by the computer, following the command codes, without storing the final results or the partial ones.
Coming now to the unit description, INTERF is designed to generate the proper handshaking signals for the RS232 line and for the system. In fact the data (32-bit strings, written as eleven octal digits, ASCII code) are preceeded by a proper command, specifying the kind of operation to be performed on the transmitted strings, and are followed by a special character to be interpreted as End Of Data. In any case the transmitted data are directly sent to COLU. In fig. 5 typical communication strings are shown.
To finish the hardware description, the testing system consists of four double Europa cards, cabled with the wire-wrap technique.

Software
To run the testing system, in connection with the host computer, it was necessary to write special codes, in Fortran language, able to control the handshaking of the system and to ensure the correct transmission and reception of data and results. In particular, the handshake between INTERF and the host computer (VAX8600), was assured by using the QIO libraries(9). The Fortran code, of about 1000 statements, has a particular structure, due to the simplified architecture of COLU. In fact the program, running on the host computer, was able to send, step by step, the proper sequence of operation codes and data (i.e. send

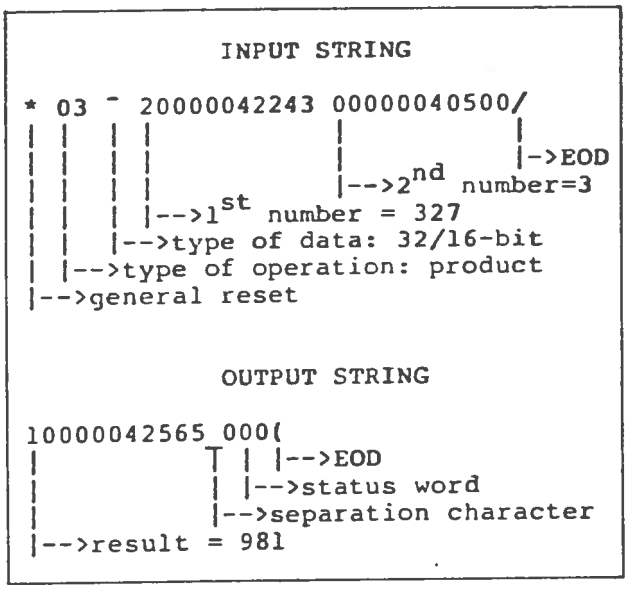

Fig. 5 - Typical communication strings

partial results, received in previous cycles, as new data for the next step).

4.2 Experimental Results

Using the test system and the host computer, we were able to test COLU and FPP in several different conditions.
Firstly, we verified the capability, specially of FPP, to compute simple operations and conversions for 32-bit floating numbers.
The satisfactory results of these preliminary tests, allowed us to try to verify the system in more difficult conditions, i.e. in the computation of different algorithms.
As beginning we computed the logarithm of a real number, using the series in power

$$\ln y = \sum_{i=1}^{n} i^{-1} \left[\frac{y-1}{y}\right]^{i} \quad (y \geq 1/2)$$

truncated at 6^{th} order.
This truncated expansion was computed by means of COLU and VAX, with an agreement better than 1/100000.
As final test, we decided to run with the power-law algorithm. The input data, simulating the collection of light reaction products (p,d,t,3He, 4He) coming from the reactions p+27Al at 36 MeV incident energy and detected at 30° by a SSD telescope with a thin detector 100 μm thick, were computed with the simulation code SCAMPI (10) and stored in the VAX memory: 40 pairs (DE,E) for each of the 5 particles (a number of data we have shown in past completely enough to test the behaviour of a nuclear electronics apparatus). In fig. 6a the values of PIF for the five particles, computed by COLU, are shown in comparison with those obtained with the DAG pulser(11) and the analog identifier(3) for the same simulated data (fig. 6b) and the on-line results obtained with the proton beam of the Milano AVF Cyclotron and the analog identifier (3) (fig. 6c).
From an experimental point of view, compairing the two simulations (fig.s 6a,6b), we can see that the results obtained with COLU show a better resolution than that obtained with an analog device.
We have to remark that all the operations to compute the values of fig.6a, were made step by step by COLU, with the exception of the maximum and minimum value computation, during the discretization. These were computed by means of the Fortran functions MIN and

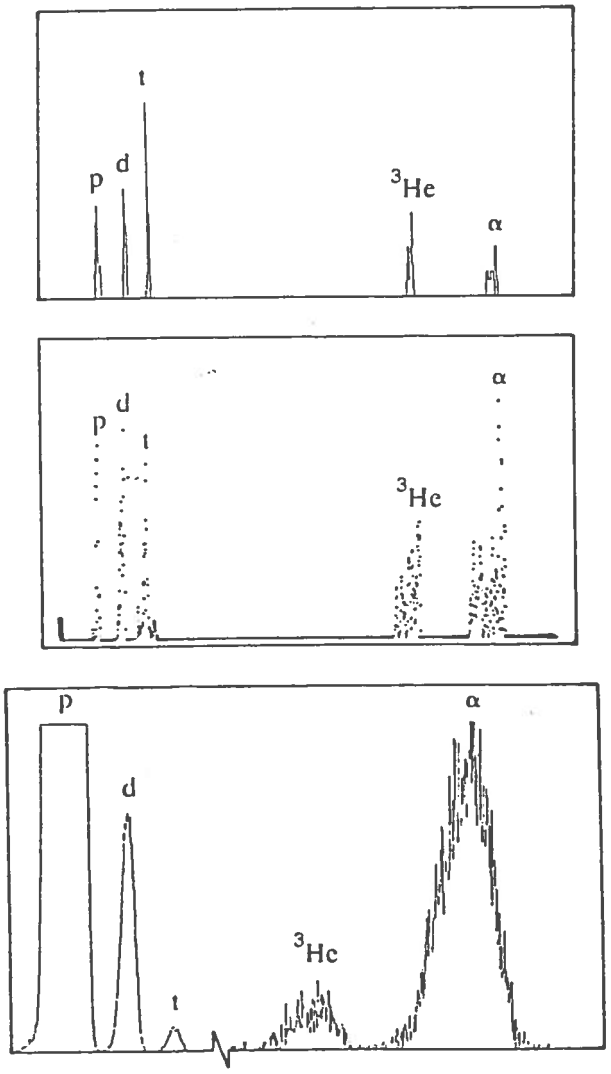

Fig. 6 - Comparison of collected data (see text): a) with COLU; b) in analog way; c) in beam

MAX in the main code.
The first control made on the results was for the power-law computation, i.e. the comparison on-line with the results obtained with VAX, directly computing the powers with the system functions: also in this case the agreement was better than 1/100000.
We verify also in a different way the goodness of the results obtained with the power-law. The PIF function introduced in Sec. 2, can be demonstrated (12,13) to be proportional to the following value

$$PIF \div M^{x-1}Z^2$$

where Z is the charge and M is the mass of the detected particle and x is the exponent of the power-law.
Using for PIF the values corresponding to the five different peaks, computed by a gaussian fit code, and a linear regression method, we obtained an integral nonlinearity for the system equal to 0.0013% and a differential one better than 0.1%.
Finally to better understand the significance of the tests made, it is to note that no care was made to verify the speed of the circuit, due to the RS232 interfacing. Anyway, following the results of the simulation code, 56 cycles are necessary for computing an only power, 117 for the whole power-law computation, while the discretization needs 7 cycles in addition.
The whole system was tested at a frequency of 2.5MHz, without any problem.
We can conclude that COLU represents an attempt to solve in different way the on-line manipulation of data collected in nuclear physics experiments, because of his architecture and the use of sophisticated algorithms in the computation. Moreover the test system here described, in connection with the simulation codes and with proper data sets, allows a careful design and an easy verifiability of the performance of this kind of apparatuses, ensuring, at the end, a correct working mode of the product, without spending additional time for the setting up.

ACKNOWLEDGMENTS

We wish to thank Mr. P. Dellera for the careful drawing of the figures and Mr. G. Frediani for the care given in the camera-ready drafting.
This work was partially supported by a grant from the Italian Ministry of University and Scientific Research.

REFERENCES

(1) Annunziata,M., Guazzoni,P., Lisca,L., Sechi,G.R. and Zetta,L., Microprocessing and Microprogramming, 18 (1986) 515
(2) Goulding,F.S. and Harvey,B.G., Ann. Rev.Nucl. Sci., 25 (1975) 167
(3) Guazzoni,P., Michelato,P., Taiocchi,G.F. and Zetta,L., Nucl. Instr. and Meth. 200 (1982) 323
(4) Shimoda,T., Ishihara,M., Nagatami, K., and Nomura,T., Nucl. Instr. Meth., 165 (1979) 261
(5) DEC, Fortran Language, Reference Manual

(6) AM 29300 Family Handbook, Advanced Micro Devices, Sunnyvale
(7) Bragagnini,W., Guazzoni,P., Pitalieri,M., Taiocchi,G.F. and Zetta,L., A Special Purpose Serial Interface, Report INFN/TC, to be published.
(8) Mick,J. and Brick,J., Bit Slice Microprocessor Design (Mc Graw Hill, New York 1980)
(9) DEC, VAX/VMS I/O Users Guide
(10) Guazzoni,P., Sechi,G.R. and Zetta,L., Nucl. Instr. and Meth. 227 (1984) 526
(11) Guazzoni,P., Michelato,P., Taiocchi,G.F. and Zetta,L., Nucl. Instr. and Meth. 185 (1981) 219
(12) Bowman,J.D., Poskanzer,A.M., Korteling,R.G. and Butler,G.W., Phys. Rev. C9 (1974) 836
(13) Walton,J.T., Goulding,F.S., Haller,E.E. and Pehl,R.H., Nucl. Instr. and Meth. 196 (1982) 107