MULTICHANNEL ANALYSER TO MICRO-COMPUTER SOFTWARE INTERFACE FOR ON-LINE DATA ANALYSIS (*)

A. M. GONÇALVES, M. H. GONÇALVES
Departamento de Física, Faculdade de Ciências, 1294 LISBOA Codex

P. BARAHONA (1)
Departamento de Informática, Universidade Nova de Lisboa, 2825 MONTE DA CAPARICA

(Received 3 June 1982; revised version 20 December 1982)

ABSTRACT — A general purpose Multichannel Analyser (MCA) to a micro-computer (μC) software interface has been developed. The system is based on a small low cost 8-bits μC running under the CP/M disk operating system, and it allows bidirectional data transfer betwenn a MCA and a μC, the storage of the acquired data on the MCA into a disk file, the decoding and condensing of the data, and it enables the user to perform an immediate data analysis. The concept of the design is user independent and it can be configured at any given time by a simple modification of some data files. Specially suited for an easy implementation on common situations in physics laboratories, the system developed offers (in many respects) the same facilities as those available in data acquisition systems based on PDP mini-computers, for a fraction of the cost.

1 — INTRODUCTION

The application of computing power in the physics laboratory is an expanding field, specially after microprocessors became generally available in the last few years. These applications

(*) Research partially supported by Centro de Física Nuclear (INIC) and Centro de Física Molecular (INIC). The present paper is based on results presented at the Third General Conference of the Portuguese Physical Society (Coimbra, June 1982).

(1) Present Address: Departamento de Estatística, Invest. Operac. e Computação, Faculdade de Ciências, 1294 LISBOA CODEX.

concern different requests in laboratory measurement work, which we may classify in the following groups: i) control; ii) data logging; iii) data analysis.

It is possible or even, in some cases, essential for laboratory on-line computing to support an integrated philosophy comprising the above functions. However it is more efficient when approaching a given problem to start by the identification of its specific and less complex aspects.

In many cases, the standard general purpose instrumentation could be the best answer for one or more of those requests. This is the case of the Multichannel Analysers (MCAs), used in many experimental devices as a sophisticated data logger. The use of a MCA in a laboratory strongly suggests, as first priority, the use of computational means for the handling of the acquired data. The use of paper punched tape to carry out the data to a computer is still used in several physics laboratories, but it is increasingly being considered as a slow and less convenient method. The industrial standard answer is the application of sophisticated and expensive systems based on PDP 11 minicomputers to perform data analysis, as they provide on-line fast and easy data transfer with MCAs.

Our aim is to develop for the same task a low cost system taking advantage of the existing industrial standard instrumentation, without using any special bus. We can find a support for our point of view in a recent review article by Brignell and Young [1].

The low price of up-to-date 8-bits μCs, together with a flexible software with increasing capabilities, provides an attractive and efficient alternative to produce in the laboratory an immediate data analysis even with a large amount of numerical treatment. Apparently the main problem which arises is how to transfer the data from the MCA to the μC without any expensive and specific interface. This problem is easy to solve as many modern MCAs have standard interfaces (e.g. RS 232 or 20 mA current loop) for data transmission to peripheral devices such as a teletype. These interfaces also provide the way to connect directly the MCA to a μC equiped with a similar interface. This restricts the problem to writing the appropriate software. The easy solution is to write programs for a given experimental situation based on a particular personal μC hardware configuration running under a resident BASIC interpreter. However, the most attractive and efficient way is to design a general purpose system, configurable to different users and experimental requirements, being hardware independent, and allowing typical procedures in data manipulation and analysis.

In this work, we report on a system program, written in a modular way, easy to fit to different experimental requirements. The system allows the data transfer from a MCA to a μC (and vice-versa), the storage of the data into a disk file, the decoding of the data, and enables the user to perform an immediate data analysis. The program runs under the CP/M disk operating system [2] which is the most frequently used in 8-bits machines based on 8080-like μPs. Consequently, this software offers great portability, and is hardware independent for systems under CP/M.

In the following sections, we describe briefly the system hardware and software. In section 3 we give particular emphasis to the description of modifications that may be required to fit different hardware configuration and users' needs. In section 4 we present an example of the system implementation.

2 — SYSTEM DESIGN

2.1 — *Hardware*

There are several microcomputers running under CP/M on the market that answer the necessary requirements. The selection of the microcomputer, based on budgetary limitations, made it attractive to choose a Zenith Z 89 [3]. This system has been equiped with a 48 k memory (8 bit word), one mini floppy disk drive (allowing a mass storage area for programs and data of 90 k), and two serial interfaces RS 232C. Unfortunately the video terminal is not equiped with vector graphic capabilities allowing only alpha numeric display. It should be stressed that this is the minimum possible configuration for any CP/M based system.

In the main tests we have used the low cost MCA Inotech 5200 [4] provided with a 20 mA current loop interface without hand shaking. This interface is designed for data output in ASCII characters on a teletype, with a data transfer rate of 110 baud. This MCA has 1024 channels, with a maximum storage capacity of 10^6-1 counts per channel. In Fig. 1, we show an overview of the system.

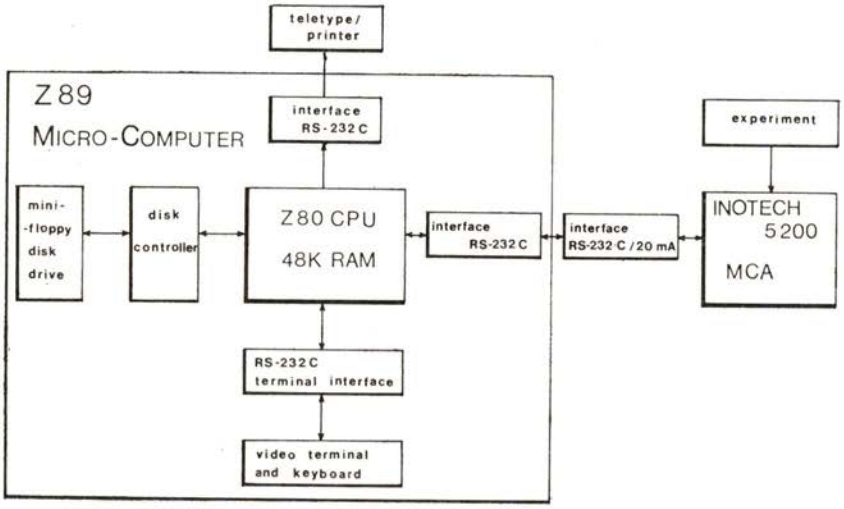

Fig. 1 — The system overview, showing the micro-computer configuration and peripherals used.

The communication between the MCA and the μC is performed through a laboratory made interface, which transforms the 20 mA current loops' signals into RS 232 C standard signals and vice-versa. To increase the transfer rate, the MCA I/O baud rate was changed from 110 baud to 1200 baud. This means that a whole spectrum of 1024 channels is transferred in less than one minute.

2.2 — *Software*

Our approach to software has been to write the program in extended BASIC (from Microsoft) [5]. Although this language does not assist a modular program structure (as it does not allow for local scope of the variables and to pass on parameters to the subroutines) this is not a major problem when the programs are not too long. On the other hand the interactive nature of the BASIC interpreter, with text editing facilities, and run time error messages, significantly reduces the effort required for program development.

The BASIC also allows to read and write external devices and to examine and alter memory locations. This facilities have been extensively used by many groups when "personal" microcomputers with a resident BASIC interpreter are applied as data-acquisition systems [6-10]. However due to the slow speed of the interpreter, we prefer to write assembly code routines (easily called from a BASIC program) to communicate with the MCA and to create a file using directly the system calls of the CP/M Operating System [2].

Using the BASIC interpreter, we need a system with almost 64 k bytes of main memory to run the whole program. In this program version, the machine code routines must be loaded into memory, before the BASIC interpreter and the BASIC source program itself. To overcome this evident drawback, the BASIC program has been compiled and linked with the relocatable modules produced by the Macro Assembler M80 [11] from our Assembly routines. In this way, we get an absolute code, transparent to the user, running significantly faster under the CP/M operating system, and demanding only a 48 k system.

3 — DESCRIPTION OF THE PROGRAMS

In Fig. 2 we show the block diagram of the whole program. In the first module, MAIN-INIT, the μC performs a dialogue with the user, and initiates the experience variables. In the module TRANSFER the data is transferred from the MCA to the μC accordingly with the variable "I". The module DECODE generates a direct access file with the number of counts of each channel, in a binary form, for future treatment. The module OPTIONS allows for the implementation of user defined data processing programs, for the immediate data analysis, using a

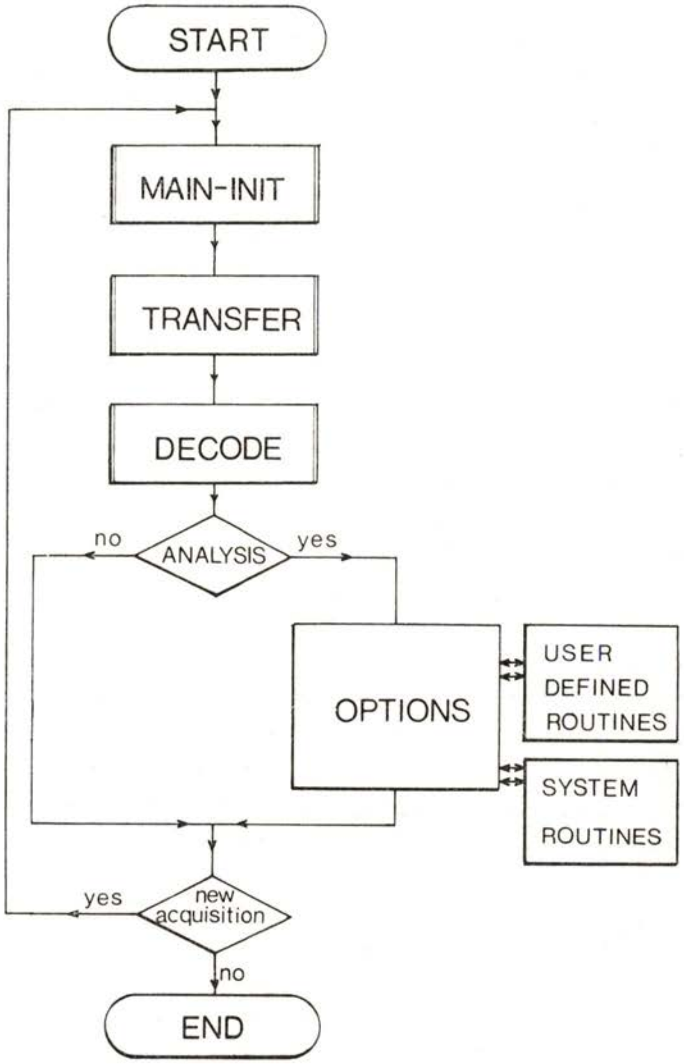

Fig. 2 — Flowchart of the acquire program. At least two parameters are generated in the MAIN-INIT module: the «filename» for the present run (FN, a string of 8 alphanumeric characters) and the number of channels to be transferred (NC). These parameters are passed through the entire program.

chaining structure. After the treatment, and since with our μC we can not view the spectrum in the video terminal, the module ENCODE generates a file in the MCA format from a direct access file of the above type. The module TRANSFER allows this data to be transferred from the μC to the MCA according with the variable "O".

In the next sections we give a detailed description of each module.

3.1 — *Main-Init*

The dialogue with the user is performed following the data read from the file PROTOCOL.DAT. This file must be defined by the user according with his own experience and can be changed at any time, to answer new requests. The file is organised in order to give the number of questions and for each of them the text defining the parameter to be inputed (e.g. material to be analized, temperature, pressure, radiation wave length etc.), the type of entry (integer, real or string) and the data variation limits for data entry validation. For integers we assume that they are the lower and the higher limits and the step eventually given as a power of 2 or 10. For reals we assume that they are the radix, and for the mantissa and the exponent the same as for integers. For strings the number of character or a special format — as DD-MMM-YY for date.

The PROTOCOL. DAT file must contain the key definition of the file name for the present run.

After the data entry, MAIN-INIT allows the optional data printing in the hard-copy device and creates a sequential file with the same data. This file has the name defined for the present run with the extention DAT.

The entry and validation process uses extensively the video attributes and cursor addressing in a user oriented mode. As other video terminals may answer differently from the H 19 we use, the video attributes and cursor addressing are defined in the file VIDEO. DAT. In Fig. 3 we show a block diagram of the whole module.

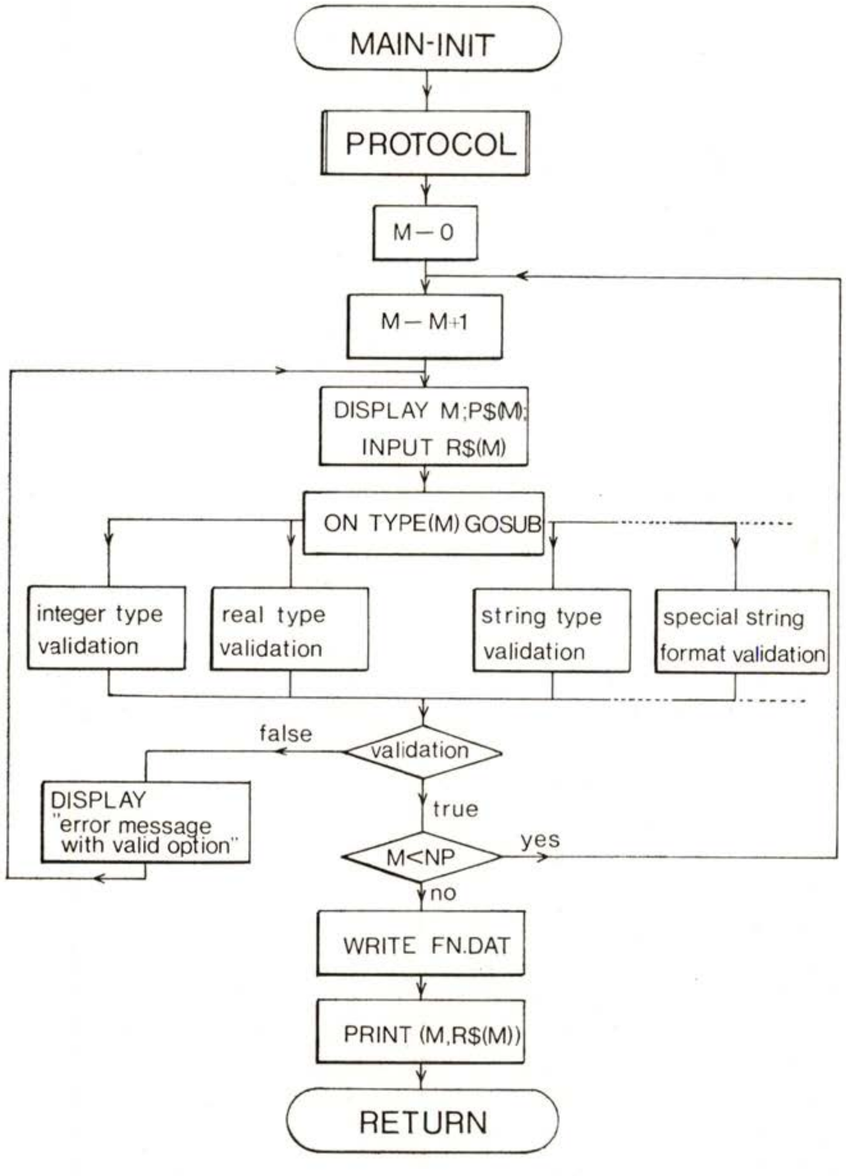

Fig. 3 — Flowchart of the MAIN-INIT module. In this module data are inputed and validated accordingly with data definitions made in the PROTOCOL.DAT file. NP is the number of parameters to be inputed, P$ represents the corresponding prompt, and R$ is the operator answer. The file FN.DAT is generated.

3.2 — *Transfer*

The data transfer from the MCA to a diskette file or from a diskette file to the MCA is handled by the TRANSFER module. Then, the entry parameters to this module are the name of the file, the number of channels to be transferred, and the direction of the transfer.

The MCA we use has no protocol to suspend the transmission of the data. To guarantee the data transfer in real time with this kind of serial interface the module TRANSFER calls two subroutines in sequence:

i) IODATA, that supervises the data transfer between the multichannel and the memory, using the "punch" and "reader" CP/M standard logical devices;

ii) RWFILE, that supervises the data transfer between the memory and the diskette.

These subroutines have been written in Assembly because with a BASIC interpreter we were not able to transfer the data, in real time, from the multichannel at high baud rates, and with Microsoft BASIC it is not possible to access the registers of the CPU (8080-like). This last facility is essential to call the CP/M system functions.

IODATA handles the data transfer, character by character, from the reader device to a buffer in the memory or from this buffer to the punch device. At the same time, the characters are echoed to the console. This routine uses the CP/M functions READER INPUT, PUNCH OUTPUT and CONSOLE OUTPUT.

RWFILE opens or creates a file with the name received by the calling module. Afterwards the number of 128 bytes sectors, which is determined by the number of channels to transfer, is loaded from the file (opened) to the memory or is stored from the memory to the file (created). Finally the file is closed. The CP/M functions called by this subroutine are: OPEN FILE or MAKE FILE, CLOSE FILE, SET DMA, READ SEQUENTIAL or WRITE SEQUENTIAL. In Fig. 4 we show a block diagram of the whole module.

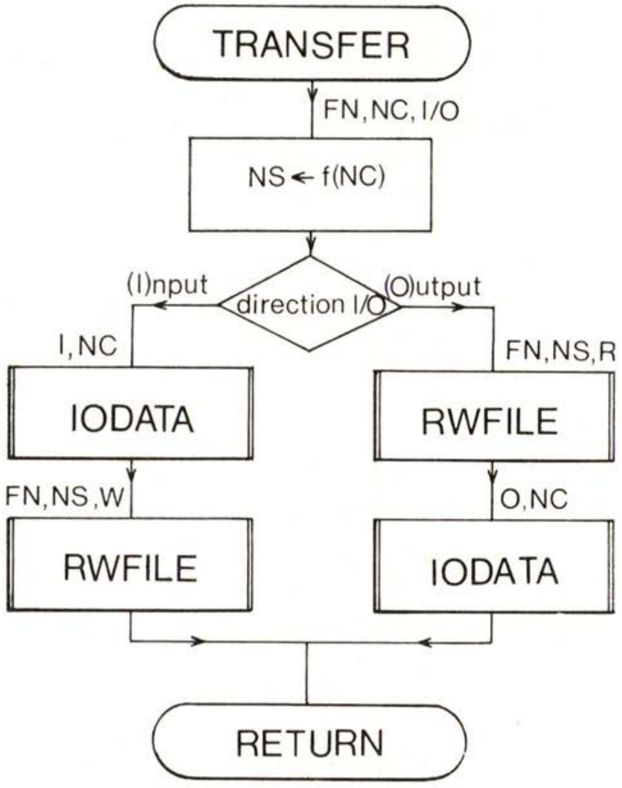

Fig. 4 — Block diagram of the TRANSFER module. FN is the «filename», NC the number of channels to be transferred, NS the number of sectors on the file FN.RDM as generated (or read) by the RWFILE subroutine, and I/O a variable which defines the direction of the transfer.

3.3 — *Decode-encode*

Since the MCA output/input presents a particular format in ASCII characters, e.g.

```
, 0000  000006  000011  000009  000014  000019
, 0005  000012  000016  000023  000020  000036
, 0010  000027  000038  000047  000031  000048
, 0015  000059  000051  000048  000061  000068
, 0020  000060  000064  000067  000079  000081
, 0025  000064  000101  000069  000084  000092
..........................................................
```

the module DECODE generates a direct access file with the extension RDM with the counts of each channel in binary format.

For pratical uses, we assume that the maximum value of each channel does not exceed 2^{15}. The maximum value allowed by the MCA is 10^6-$1 < 2^{20}$. If an "overflow" condition arises, a warning message is edited, and only the remaining value is stored. This procedure is practical in our case, as we do not expect such high counting rates, and so we could store a whole spectrum in only 2 kbytes. If necessary, a simple modification of the file specification can be done allowing greater values. Alternatively, a spline interpolation method could restore the original spectrum before treatment.

The module ENCODE performs the inverse procedure. These modules are designed to deal with the common MCA output format, described before. For different MCA's I/O formats one could implement equivalent modules, without significant effort.

3.4 — *Options*

In the module OPTIONS, user defined data processing programs can be specified allowing for immediate data analysis, using a chainning structure. Essentially, this module is a menu driven routine, and it uses the file MENU. DAT specified by the user, containing the filenames of his own data analysis programs.

After the treatment, and since with our microcomputer we can not view the spectrum on the video terminal, the ENCODE-TRANSFER modules – system routine VIEW – allow the transfer of the spectrum for display in the MCA.

4 — AN IMPLEMENTATION EXAMPLE

This system was implemented and field proven in the environment of a photoelectron spectrometer. For each run, some relevant parameters are stored, following the procedure described in section 3.1. In Fig. 5 we show the corresponding hard-copy form. The file name for this run, as determined by the key specification, was AR314118. This name is used through all the system programs for the present run, allowing the identification of all data files produced during the data manipulation process.

```
PHOTOELECTRON SPECTRUM                                      Pas 1


RUN AR314118                                            11-AUG-81
============


        UV source : selected power      :   1
                    He pressure         :  1E-1 torr
                    temperature         :   38 'C
        Sample    : sas                 :  ARGON
                    inicial pressure :  1E-5 torr
                    final pressure      :  8E-5 torr
        Sweep control : scale           :   3
                        off-set         :   14
                     time/channel       :   2 s
                     number of sweeps:      1
                     inicial channel :      400

        Number of channels to be transferred :      150
```

Fig. 5 — Relevant parameters recorded during a photoelectron spectrometer run, which are saved in the file AR314118.DAT. The «filename» FN for this run includes abridged information about the sample (the first two letters), the scale and the off-set of the sweep control (3 and 14), and the date (11th August) [14].

As specified (see fig. 5) only the first 150 channels have been transferred and stored in the file AR314118.MCA. This file, in ASCII characters, is an image of the MCA output, according to the format presented in 3.3, and could be transferred to the MCA at any given time through the system routine VIEW.

Once the transfer is finished, the DECODE module produced a new file, AR314118.RDM, in binary format. The original file could now be deleted, if storage space in the diskette is not enough, as the module ENCODE could reconstruct the original file from this binary file. In fact, for a 1024 channel spectrum, the original file occupies 9k bytes. In the condensed binary format, the same spectrum requires only 2k bytes of storage space.

The channel contents could be printed, optionally, in the form presented in Fig. 6.

```
PHOTOELECTRON SPECTRUM                                                 Pag 2


RUN AR314118                                                       11-AUG-81
============


          CHANNEL = N*10+D

+-----+------+------+------+------+------+------+------+------+------+------+
:\    :                                                                     :
: \   :                                                                     :
:  \D :   0     1       2      3      4      5      6      7      8      9  :
: N \ :                                                                     :
:    \:                                                                     :
+-----+------+------+------+------+------+------+------+------+------+------+
:     :                                                                     :
: 40  :    6     11      9     14     19     12     16     23     20     36 :
: 41  :   27     38     47     31     48     59     51     48     61     68 :
: 42  :   60     64     67     79     81     64    101     69     84     92 :
: 43  :   97     91    109    101     93    105     89    124    115    101 :
: 44  :  122    138    128    150    151    182    159    216    227    220 :
: 45  :  262    295    316    420    469    525    752   1164   1949   2895 :
: 46  : 3982   3900   2667   1400    788    508    316    284    190    176 :
: 47  :  154    121    129    102     95    103     79     82     78     80 :
: 48  :   76     83     78     72     84     77     92     84     85    119 :
: 49  :   99    115     99    112    114    122    146    149    160    196 :
: 50  :  224    284    320    393    604    906   1326   2005   2063   1625 :
: 51  :  950    482    316    208    176    127    107     79     92     77 :
: 52  :   46     56     42     50     45     43     37     45     38     24 :
: 53  :   32     35     27     22     35     32     33     23     28     27 :
: 54  :   33     34     24     30     42     29     30     30     29     24 :
+-----+------+------+------+------+------+------+------+------+------+------+
```

Fig. 6 — Decoded spectrum contents as they have been saved in the random access file AR314118.RDM.

In the present example, the user has implemented some routines for data analysis [12]. These programs use the files AR314118.DAT (containing the experimental parameters) and AR314118.RDM (containing the spectrum). The result of the analysis performed is shown in Fig. 7, the processing time being less than 2 minutes. This on-line analysis presents evident advantages over any other performed on a later stage of the

```
PHOTOELECTRON SPECTRUM                                                     Pag 3


RUN AR314118                                                           11-AUG-81
============


Resonance Line : He I (21.2175 eV)

Working temperature  311 'K


+---------+---------+---------+---------+---------+---------+---------+
:         :         :         :         :         :         :         :
:  Peak   : Position:  Ioniz. : Kinetic :  f.h.w. :Intensity:Assimetry:
:         :(channel :Potential:  Energy :         :         :         :
:         : number) :   (eV)  :   (eV)  :  (meV)  :         :   (%)   :
+---------+---------+---------+---------+---------+---------+---------+
:         :         :         :         :         :         :         :
:2P 3/2   : 461.311 : 15.759  :  5.459  :  17.87  : 13826.8 : +10.34  :
:         :         :         :         :         :         :         :
:2P 1/2   : 508.711 : 15.937  :  5.281  :  17.41  :  7060.2 :  +2.84  :
:         :         :         :         :         :         :         :
+---------+---------+---------+---------+---------+---------+---------+


+---------+---------+---------+---------+---------+---------+
:         :         :         :         :         :         :
:  Peak   :  f.h.w. : Natural :Act.Inst.:Teor.Inst:  Other  :
:         :         :  Width  :  Width  :  Width  : Widths  :
:         :  (meV)  :  (meV)  :  (meV)  :  (meV)  :  (meV)  :
+---------+---------+---------+---------+---------+---------+
:         :         :         :         :         :         :
:2P 3/2   :  17.87  :   4.72  :  17.23  :  16.15  :   6.01  :
:         :         :         :         :         :         :
:2P 1/2   :  17.41  :   4.64  :  16.79  :  15.63  :   6.13  :
:         :         :         :         :         :         :
+---------+---------+---------+---------+---------+---------+
```

Fig. 7 — Condensed output produced by an immediate data analysis program [12], for which files presented in formated «hard-copy» form (Figs. 6 and 7) have been used.

experimental procedure. For example, any drift in the applied voltage changing the calibration condition of the spectrometer is immediately detected after spectrum recording. This allows the immediate correction of the anomaly. The same would happen with the mechanical adjustment of the slits, which produces sharp differences in the achieved instrumental resolution. These facilities have reduced drastically the time consumed in the "mise au point" of the spectrometer and in spectra analysis.

5 — CONCLUSIONS

Based on a small 8-bits microcomputer running under the disk operating system CP/M, we have developed a general purpose system allowing data transfer between a MCA and a μC. The system gives the necessary facilities for an immediate data analysis.

In spite of the small dimensions of the peripheral memory used in the tests, the system offers possibilities comparable to those available on systems based on PDP 11 mini-computers [13] for a fraction of the price and with simplified installation procedures. Without any sofware modification, the performance of the system could be increased by an order of magnitude using a more sophisticated hardware, with greater mass memory, DMA controller and arithmetic processor, once it is CP/M compatible. Vector graphics capabilities on video-terminal are also desirable for a convenient spectrum display.

Higher baud rates than used are provided by the μC serial interfaces — till 9600 baud — and supported by the TRANSFER module. The use of those transfer rates is only restricted by the MCA interface. The use of 64 k bytes of main memory will allow the manipulation of a larger spectrum of 8192 channels.

As a final remark we shall refer that this system is suited to small budgets. It offers the means to provide computer power in the laboratory through a low cost μC. If more powerfull computing power is required, the same μC operating as a virtual terminal can be connected to a main-frame system, allowing data submission for (later on) complex data analysis. In this sense, the μC acts as an "intelligent interface" between a MCA and a main-frame in a dial-up type "on-line" connection, allowing for the decision to transfer only valuable experimental data and relevant parameters.

The authors are grateful to Prof. F. D. Santos for his critical reading of the manuscript. We are also grateful to the Centro de Física de Fenómenos de Ionização Interna (U.L.) and Departamento de Energia e Controle (U.N.L.) for the kind provision of equipment and software tools.

REFERENCES

[1] J. E. BRIGNELL and R. YOUNG, *J. Phys. E: Sci. Instrum.*, **12** (1979), 455.

[2] CP/M Operating System ver 2. 1979 (Digital Research, POB 579 Pacific Grove, CA 93950, USA).

[3] Z89 Operation and Hardware Manual, 1979 (Zenith Data Systems-Heath Company, Benton Harbour, MI 49022, USA).

[4] IT 5200 Instruction Manual (INO-TECH Inc., Nordland Drive, Fort Atkinson, WI 53538, USA).

[5] BASIC-80 Reference Manual, 1979 (Microsoft, 10800 N. E. Eighth, Suite 819, Bellevue, WA 96004, USA).

[6] J. M. JEDJU, *Rev. Sci. Instrum.*, **50** (1979), 1077.

[7] A. R. D. RODRIGUES and D. P. SIDDONS, *J. Phys. E: Sci. Instrum.*, **12** (1979), 403.

[8] J. W. FARLEY, A. H. JOHNSON and W. H. WING, *J. Phys. E: Sci. Instrum.*, **13** (1980), 84.

[9] RON C. ESTLER, *Rev. Sci. Instrum.*, **51** (1980), 1428.

[10] R. SHAW, J. A. HARDCASTLE, A. R. B. JUSON and M. J. BONGKIK, *J. Phys. E: Sci. Instrum.*, **14** (1981), 301.

[11] Microsoft Utility Software Manual, 1979 (Microsoft, ibid).

[12] M. H. GONÇALVES and A. M. GONÇALVES, to be published.

[13] N. NUDING, G. WILL and E. HINZE, *Nucl. Instr. and Meth.*, **173** (1980), 329.

[14] M. H. GONÇALVES, *Internal Report*, Centro de Física Molecular, 1982.