Data-acquisition and automation system based on a small microcomputer

Facundo Ruiz and Jesús Urías
Instituto de Física "Manuel Sandoval Vallarta",
Universidad Autónoma de San Luis Potosí,
78000 San Luis Potosí, SLP, México.
(Recibido el 29 de noviembre de 1989; aceptado el 1 de agosto de 1990)

Abstract. An inexpensive and general purpose data-acquisition and automation system using a low-cost microcomputer as controller is described. Eight multiplexed analog inputs; two low-power analog outputs; a set of eight actuators and a RS–232 serial port are the system's facilities. A kernel of subroutines in assembly language is used to enlarge the BASIC lexicon of the microcomputer with five new instructions to operate all the system functions from the BASIC interpreter. Results in the automated C-V characterization of diodes and common-emitter output characteristics of transistors are presented. Some other applications are briefly discussed.

PACS: 98.80.Cq; 51.10.+g; 52.60.+h

1. Introduction

Oversimplifying things, one could say that activities in any physics laboratory are either part of a data collection process (the measuring process itself) or of a data analysis process; *i.e.* the process that converts data into significant information. In the last few years computers have proved to be of great help both as controllers to automate the data collection process as well as for data analysis. This double ability of microcomputers of being both controllers and formal processors has introduced a deep unification of the measuring and analysis processes. As a result, computers have become essential equipment in every physics laboratory, where by now most of the activities are organized around them. This situation has faced physicists, and other scientists, with the urgent need of acquiring a computer culture deeper than just writing high-level language programs. Physicists are familiar with electronics and programming but they need to develop skills for the use of computers as controllers in experimental science.

After some experience with commercially available equipment, we became convinced that a group capable of developing the hardware and software necessary to incorporate microcomputers into the measuring process should be associated with every research physics laboratory. Some of the advantages that were recognized are (i) *ad hoc* tailoring, (ii) easy expandability to meet future needs, (iii) training of students in the applications of microprocessors to physics, (iv) quick and

inexpensive servicing, (v) efficient management and exchange between computers of data files and (vi) much lower prices. It was thus decided to develop the present data-acquisition and automation system (DAAS).

The guiding idea we had in mind when conceiving the DAAS was to allow users in our semiconductor laboratories, using a small and inexpensive microcomputer, to develop their own applications in the study of electronic and optical properties of new semiconducting materials. For instance, experiments to measure the optical-absorption coefficient and the minority-carrier diffusion length in semiconductors using photocurrent methods [1,2] are based on one-sided junctions obtained by Schottky barriers; electrorreflectance experiments [3] are made by modulating the bias voltage across a Schottky barrier, and MOS structures are used to profile, e.g., electrolytically diffused impurities into Si [4]. In all these cases characterization of the junctions by numerically processing capacitance versus bias voltage (C-V) measurements [5] is required. Evidently, a station for the automated acquisition of digitized C-V measurements to perform routine characterization of rectifying junctions and MOS structures is particularly useful. Another example of routine characterization in our crystal-growing laboratory are Hall effect measurements [6]. Many other semiconductor experiments that can be automated by the DAAS might be cited [7].

The DAAS hardware is a set of several memory-mapped devices that are addressed individually by a microcomputer through the expansion port; almost any of the low-cost microcomputers that are commercially available provide the user with at least one such port. Due to financial restrictions we had to chose one of the cheapest, a Commodore 64 [8].

The DAAS was implemented to provide the user with eight multiplexed analog inputs, two programmable low-power voltage supplies and a set of eight actuators. A TTL to RS–232 level conditioner provides the DAAS a port for information exchange. Except for the actuators themselves and the RS–232 level conditioner, all the devices were assembled within a small PC board connected to the expansion port of the microcomputer. A Floppy disk unit and a 5" monochromatic monitor complete the DAAS which was mounted in a rack with the rest of the equipment (see Sec. IV).

The DAAS software written for the C64 provides an interactive environment and has the flexibility to meet the needs of a rather wide diversity of experiments and applications, without demanding sophisticated programming skills from the user. Any operating condition of the DAAS is programmable from the BASIC environment of the microcomputer: the approach adopted uses resident machine-code subroutines accessed through the BASIC interpreter. A kernel of subroutines that takes care of all the primary functions of the hardware was written in assembly language and used to enlarge the lexicon of the BASIC interpreter with five new instructions. It should be said that this approach does not give the system the fastest response; however, it provides the user with an amenable programming environment and allows the use of floating point arithmetics to handle and process data.

Results obtained with the DAAS in the C-V characterization of diodes and collector characteristic curves of transistors are presented in Sec. IV. They show that the DAAS is a reliable alternative to commercial equipment. Altogether with

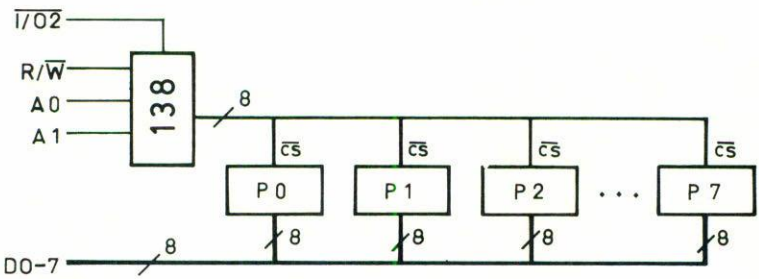

FIGURE 1. Decoding logic used to map four write-only and four read-only devices to four memory addresses through the expansion port of the microcomputer used as controller.

conventional measurement equipment, the DAAS can automate a wide range of, e.g., semiconductor physics experiments without demanding sophisticated programming skills from the user. Almost any experiment in the frequency domain or in a steady state regime (*i.e.*, no transients occur or they are very slow as to be tracked) can be automated by just writing simple BASIC programs for the DAAS. The low-cost of the DAAS, including the microcomputer used as controller, makes it affordable to put one in every rack. When the data analysis requires lengthy computations, raw data acquired with the system is transferred, through a small local network, to a central PC-compatible or any other bigger machine to do the processing.

2. Hardware development

The system hardware consists of several devices capable of independent operation, structured to integrate a single unit (the DAAS), that can be connected to the expansion port of any microcomputer through a very simple decoding logic, shown in Fig. 1. The DAAS was implemented for a Commodore 64 microcomputer (C64), based on the 6510 μP [8], but it can easily be adapted for use with IBM PCs as controllers.

Fig. 1 shows the basic decoding logic that allows the controller to command eight different devices, P0–7, to exchange information on the data bus, DO-7 (a device is active on the bus whenever its $\overline{\text{CS}}$ line is driven low). A 3-bit to 8-line decoder (TTL 74138) is used to assign two devices, a read-only device (ROD) and a write-only device (WOD), to a single memory address. Odd-numbered devices (P1,P3,...) are ROD's and even-numbered devices (P0,P2,...) are WOD's.

As shown in Fig. 1 the addresses are recognized and assigned to a ROD/WOD by the decoder only when the $\overline{\text{I/O2}}$ line is brought down by the controller. The $\overline{\text{I/O2}}$

line plays the role of an *attention line* driven by the controller to call the attention of the DAAS. The simple decoding logic shown in Fig. 1 allows the programmer to read/write data from/into the devices just in the same way he reads/writes memory locations, *i.e.*, the '138 decoder maps eight devices into four memory locations. To the computer and the programmer it does not matter what is inside each one of the P0–P7 blocks in Fig. 1.

The DAAS was originally implemented for a C64 with *a unique* expansion port that drives its $\overline{\text{I/O2}}$ line low whenever the μP addresses any of the memory locations in page \$DF; we chose the \$DF00-\$DF03 locations. The very simple decoding logic in Fig. 1 can be put to work on the IBM PC bus with just a few changes. However we must say that being originally designed for the C64, the decoding logic in Fig. 1 will not be the most efficient way of using the IBM PC bus.

The IBM-PC bus drives its AEN line low whenever the 8088 is able to address any of the I/O locations, where the \$300-\$31F space is nominally reserved for prototype cards. The decoding logic cannot be used as it is, since the IBM-PC bus will surely have several cards attached to it (e.g., drive controllers, serial port, printer port, etc.), and one has to avoid the coincidence of devices at the same addresses. We have verified that the following assignments of the DAAS lines to IBM-PC bus lines works fine: the $\overline{\text{I/O2}}$ line (see Fig. 1) driven by the AEN line; R/$\overline{\text{W}}$ driven by $\overline{\text{IOW}}$; A0 and A1 assigned to A8 and A9, respectively and the $\overline{\text{G2B}}$ pin of the '138 decoder (not shown in Fig. 1) driven by the combination ($\overline{\text{IOR}}$ or $\overline{\text{IOW}}$). With these particular assignments the \$$X$00 IBM-PC bus address ($X = 1, 2$ or 3) is mapped to the P2X ROD and to the P($2X + 1$) WOD; notice that the case $X = 0$ is not singled out by this very simple decoding logic. For those interested in exploring the IBM-PC bus, Ref. [10] is a good introduction and Ref. [11] is a very complete reference about the interfacing techniques for the 8086-8088 family.

In the following, specific memory addresses will refer to the C64.

For data-acquisition an ADC-0809 [12] analog-to-digital converter is used; it provides eight multiplexed analog inputs that are digitized to 8-bit numbers: the voltage range 0-5 V is converted to a number from 0 to 255. The decoder sees the ADC converter as devices PO and P1, corresponding to the memory location \$DF00 for both read and write operations. The analog input to be digitized by the ADC is selected by the three least significant bits written at \$DF00. The analog inputs are numbered from 0 to 7, so that to write a number N(0-255) at the \$DF00 location has the effect of selecting the analog input N(mod 8) for conversion. The ADC converter is wired in a free running configuration and driven by a clock that runs at a frequency of 700 Khz, giving the ADC a conversion rate of about 100 μsec. In this configuration the ADC digital output can be picked up by the μP by just reading the memory location \$DF00, at any time and without a previous protocol. However, it should be noticed that the ADC output will be significant only after the first conversion cycle has been completed, *i.e.*, at a time delay of at least 100 μsec after the input-channel selection has been done.

WOD's P2 and P4 in Fig. 1, at addresses \$DF01 and \$DF02, respectively, are two 8-bit latches (TTL 74245) used as inputs of two DAC 0809 [13] digital-to-analog converters. The DAC's reference voltage is 5 V so the output voltage is 5/255 times

the 8-bit number written by the μP at the corresponding latch. The maximum current that can be drained from the output amplifiers is of about 200 mA, enough to drive, e.g., any programmable power supply in situations that demand the use of high currents.

Finally, another 8-bit latch is WOD P6 in Fig. 1 Actuators in a set of eight are driven through optocouplers by these latch output bits. In the first version of the DAAS we chose the actuators to be four 2p–2t relays and four triacs.

3. The system software

The software to operate the system was required to meet the following characteristics:

i) Support a wide range of experiments and applications;

ii) Accessible to users that do not have advanced training in computer programming;

iii) Provide an environment for interactive operation;

iv) Incorporate the speed of machine code subroutines.

It was found that the above-mentioned conditions could be met by structuring the software around a set of five *interface* programs written in assembly language and letting them be resident in the BASIC environment of the microcomputer during normal operation of the DAAS. The interface programs are supported by a kernel of subroutines, written in assembly language, that take care of all the primary operations of the DAAS. A *wedge* [14] is introduced into the BASIC interpreter to allow the interface programs recognize a set of five new instructions at the BASIC environment and to interpret them as operations of the DAAS. When the BASIC interpreter reads one of the new instructions, the execution flow is vectored by the wedge to the corresponding interface program. Once the program completes the DAAS operation, execution is vectored back to the BASIC environment.

After a few experiences this approach was found to be the best way of combining the speed of machine-code programs with a friendly and comprehensive interactive environment. The set of five new instructions incorporated to the BASIC lexicon are described very succinctly in Table I. They can be used freely in any BASIC program without special considerations; except, of course, that the interface programs, the kernel and the wedge must be resident within the BASIC environment. Notice that all the new BASIC words begin with an exclamation.

Automation programs for the DAAS can be complemented by the use of, e.g., software for high resolution graphics to report on the screen the present status of the experiment, send some particular data to a printer, set warning alarms, dump RAM buffers into floppies, etc. At this point, any further elaboration is left to the user's fantasy!

New BASIC instruction	Description
`!ACQ(A)`	The floating point variable A takes an integer value in the range 0–255 proportional to the voltage at the previously selected input channel (See the `!CHNL(N)` instruction).
`!CHNL(N)`	N = 0–7. Puts the N-th analog input ready to read data.
`!SET(N)`	N = 0–7. Turns the N–th actuator on.
`!UNSET(N)`	N = 0–7. Turns the N–th actuator off.
`!VOLT(N,A)`	N = 1–2; A = 0–255. Sets the voltage at the N–th output to $(5/255)A$ volts.

TABLE I. Set of new instructions incorporated to the BASIC lexicon to operate the data-acquisition and automation system.

4. Applications and results

For applications the DAAS has to be combined with some conventional measuring equipment. Automation of (i) C-V measurements of rectifying junctions and (ii) collector characteristics of transistors are the two simple experiments that are considered.

To implement a characterization station, the DAAS was mounted in a rack together with two Keithley 177 DMM, that provide an analog output in the range 0-2 V proportional to the measured quantity, and a capacitance meter 410 of PAR Instruments. This rack of instruments is a very complete station for the characterization of semiconductor devices [5] if provided with the adequate BASIC programs to operate the DAAS (see Table I).

For the automation of static characterization of semiconductor devices the system must (i) have the control of all the bias conditions (ii) acquire the relevant set of data at every bias condition and (iii) process data to get and report the device parameters. Only the first two steps require software to operate the DAAS. They are described in some detail for the transistor characteristics.

To program base-emitter biasing, one of the two DAAS voltage supplies drives the base current through a limiting resistor. As usual, the resistance value is selected by trial and error to get an adequate current range. The collector-emitter voltage, V_{CE}, of low-power transistors may be programmed directly by the DAAS using its second voltage supply. To bias power transistors, draining high collector currents, I_C, the analog output should first drive a programmable power supply. The base current and collector-emitter voltage are settled by the instruction `!VOLT(N,A)` (see Table I).

For collector characteristics the relevant data is a set of (I_C, V_{CE}) pair values at different constant values of base current. Although V_{CE} was provided by the system itself, the actual values were read by one of the system's analog inputs. The collector current I_C was read as the analog output of a Keithly ammeter. The base current was read as the voltage applied to the limiting resistor and the base current was calculated using a constant voltage drop across the emitter-base junction.

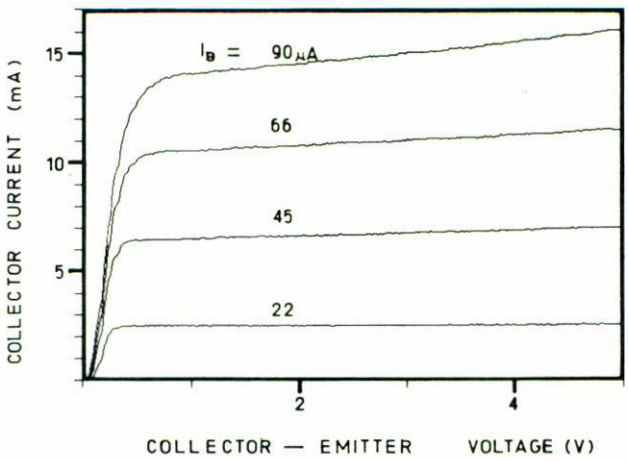

FIGURE 2. Common-emitter output characteristics of a 2N2222 transistor after data acquired with the DAAS.

One of the nice features of the DAAS is that the user can perform data averaging using floating point arithmetics, provided by the BASIC environment. It is thus easy to get data with very good statistics, a very high signal to noise ratio and a resolution much better that the single reading eight-bit resolution of the ADC [15]. The BASIC program that was written to characterize transistors with the DAAS, creates files of averaged (I_C, V_{CE}) pair values for several settings of the base current. To get nice graphical displays, all data files were transferred to a PC-compatible microcomputer and commercial software for graphics was used.

Fig. 2 shows the resulting collector characteristics of a 2N2222 transistor obtained this way with the DAAS. It should be stressed that the use of floating point arithmetics to calculate average values produces very accurate data values, greatly improving the 8-bit single reading resolution of the ADC and improving also the signal to noise ratio [15]. The 2N2222 collector characteristics in Fig. 2 represents the average of twenty readings at every bias condition.

Following a similar procedure averaged C-V data was acquired, filed and transferred by the DAAS for the 1N4001 diode. The results are shown in Fig. 3 as a $1/C^2$ versus bias voltage plot [5]. The small squares represent average values of twenty readings at the same bias condition. The straight line is a linear fit to the negative-bias points [4]. Notice that the bias voltage runs from negative to positive values in Fig. 3 while the analog inputs of the DAAS can only read positive voltages. To cover the full bias range, one relay from the set of actuators was used to switch the diode polarity by using the `!SET(N)` and `!UNSET(N)` instructions.

In conclusion, the DAAS, altogether with some conventional measurement equipment, is, following the criteria proposed in Ref. [16], adaptable (*i.e.*, it can be used to automate a wide range of physics experiments) and comprehensive (users may develop their own applications without requiring sophisticated programming tools).

The DAAS, as it was originally implemented, is not portable but its low cost

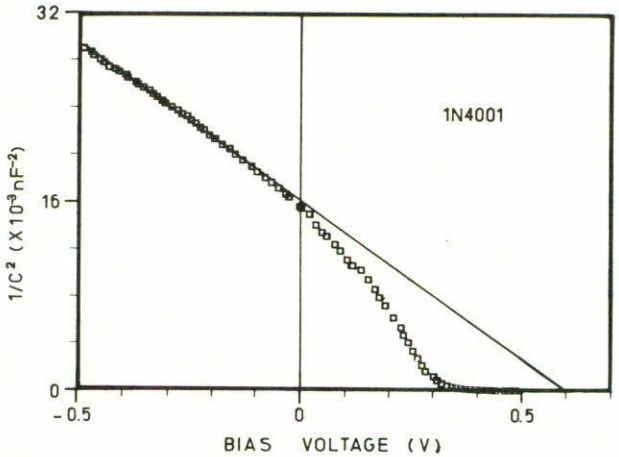

FIGURE 3. $1/C^2$ characteristics of diode 1N4001 as obtained with the DAAS. The straight line is a linear fit to the negative-bias points.

including the microcomputer used as controller, makes it affordable to put one in every rack and to interconnect them all through a small local network. Portability is thus substituted by the capability of exchanging information. For instance, raw data acquired with the DAAS can be transferred to any other computer in the network for data-processing. Alternatively, one might implement the DAAS for IBM-PCs [17] and use a compiled language, as Turbo Pascal or Turbo-C, that have a complete set of built-in I/O instructions to write portable software.

Finally, the results obtained for diodes and transistors (see Figs. 2 and 3) have convinced us that the data-acquisition and automation system reported in this article is a reliable alternative to commercial equipment.

Acknowledgements

One of us (JU) would like to acknowledge a fruitful conversation with W. von Ruden on a first version of the DAAS. This work was partially supported by Dirección General de Investigación Científica y Superación Académica, SEP (México) under contract C89-03-0177 and by FAI-UASLP.

Note added

Source code listings and detailed schematics of the C64 implementation of the DAAS are available directly from the authors.

References

1. A Lastras Martínez, G. Ramírez Flores y J.M. Montejano Carrizales, *Rev. Mex. Fis.* **36** (1988) 393.
2. T. Sukegawa, T. Watanabe, T. Mizuki and A. Tanaka, *IEEE Trans.* **ED-27**, 1251 (1980).
3. B.O. Seraphin, Electroreflectance; appeared in *Semiconductors and semimetals*, Vol. 9 edited by R.K. Willardson and A.C. Beer, Academic Press (1972). D.E. Aspnes, Modulation spectroscopy: Electric field effects on the dielectric function of semiconductors; appeared in *Handbook on semiconductors*, edited by T.S. Moss, Vol. 2, edited by M. Balkanski, North Holland (1980).
4. S.M. Sze, *Physics of semiconductor devices*, J. Wiley (1969). Chap. 8.
5. C. Ruiz, J.L. Martínez and J. Urías, *Appl. Phys. Letters* **44** (1984) 1073.
6. L.J. van der Pauw, *Phillips Res. Repts.* **13** (1958) 1.
7. For instance see J.I. Pankove, *Optical processes in semiconductors*, Dover, NY (1975).
8. *Commodore 64 programer's reference guide.* Published by CBM Inc. and H.W. & Co. (1983).
9. Technical data sheets, INTEL 1980.
10. J.R. Drummond, Three bus interface designs for the PC. Appeared in *Inside the IBM-PCs*, edited by BYTE (1987) pp. 225–245.
11. John Uffenbeck; *The 8086-8088 family: design, programming and interfacing*, Prentice-Hall (1987).
12. *Linear data book*, National Semiconductors p. 8.60, (1982).
13. *Ibidem.* p. 8.160
14. *Intern*, edited by Data Becker Buch, Dusseldorf (1984).
15. F. Ruiz; Adquisitor de datos y sistema de control programable desde BASIC. Tesis de Maestría; UASLP (1988).
16. C.D. Spance, P. Seligmann and D.A. Briotta, *Computers in Phys.* **1** (1987) 59.
17. The implementation of a series of instruments for the IBM-PC bus is in progress.

Resumen. Se presenta un sistema de automatización y adquisición de datos de propósito general que usa una microcomputadora de bajo costo como controlador. El sistema provee ocho entradas analógicas multiplexadas, dos salidas analógicas de baja potencia, un banco de ocho actuadores y un puerto serial RS-232. Se utiliza un kernel de subrutinas escritas en ensamblador para ampliar el léxico de BASIC de la microcomputadora con cinco nuevos comandos con los cuales operar el sistema desde el intérprete de BASIC. Se presentan resultados obtenidos en la caracterización C-V de diodos y curvas características de salida de transistores. Se discuten brevemente algunas otras aplicaciones.