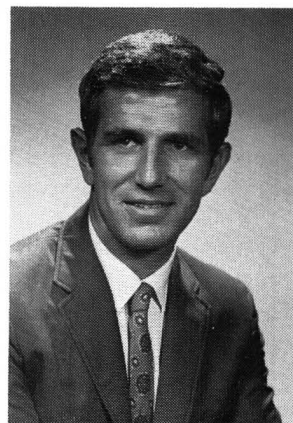


# A COMPUTERIZED UNDERGRADUATE PROCESS DYNAMICS AND CONTROL LABORATORY

R. A. SCHMITZ  
*University of Illinois*  
*Urbana, IL 61801*



A project to computerize the undergraduate process dynamics and control laboratory in the Department of Chemical Engineering at the University of Illinois was begun in 1969. The plan was to use an existing IBM 1800 computer, which was operating on a time-sharing basis in the School of Chemical Sciences, to provide a facility for computer-aided instruction and for computer-aided experiments by undergraduate students. Our motivation for the project was the realization that a facility of this type would permit more extensive and more meaningful laboratory studies of process dynamics, simulation and control than are possible in conventionally-equipped laboratories. We also felt that the hands-on use of an on-line data acquisition and control system would be a valuable experience for the students and that the lecture portion of certain courses would be greatly improved by the demonstrations and visual displays made possible by an on-line computer. Similar projects have been undertaken at a few other chemical engineering departments at universities in the United States and Canada<sup>1-7</sup>. A recent report of the CACHE (Computer Aids in Chemical Engineering Education) Committee<sup>8</sup> presents a brief description of these activities.

Certainly at this point in time, the use of a computerized undergraduate laboratory in ChE education is a relatively new development. Since the few facilities that are in existence were established quite dependently, each incorporates some unique features, and publications of their descriptions should be informative and valuable to other departments which may embark on similar projects. Many such departments will probably find themselves in a situation similar to ours at Illinois at the outset of this project; that is, with an accessible time-sharing computer of the large or medium-sized variety and an existing laboratory equipped with conventional instruments and controllers. While the task of interfacing the laboratory and the computer and of developing a software system in such cases may be demanding in

Roger A. Schmitz received his BS degree from the University of Illinois and his PhD from the University of Minnesota. He joined the ChE faculty at the University of Illinois in 1962. His area of specialization is the stability and control of chemically reacting systems. He was awarded a Guggenheim Fellowship in 1968-69 and was the recipient of the Allan P. Colburn award of the AIChE in 1970.

terms of labor, the investment of dollars in hardware should be minimal. Departments not so fortunately situated with regard to an existing computing facility will probably go the route of investing in a minicomputer.

## THE SYSTEM AND ITS OPERATION

The IBM 1800 computer in the School of Chemical Sciences has 32,000 words of core memory and uses three disks (500,000 words each) for auxiliary storage. It is equipped with the usual peripherals including (1) an analog-to-digital (A/D) converter capable of receiving signals on a -5 to +5 volt range, (2) a digital-to-analog (D/A) converter capable of sending voltages on the range 0 to +5 volts, and (3) a card reader. A description of the computer system, the monitoring program and IBM's Time-Sharing Executive (TSX) System has been published.<sup>9</sup>

Three thirty-conductor cables connect the computer to the process control laboratory. These cables actually branch to nine stations connected in parallel. Four of these are chemical engineering research laboratories, another is a terminal in a room housing an EAI 580 analog computer, and the remaining four are in the process control laboratory. One cable is devoted entirely to an IBM 1053 output printer which can be connected at any of the stations.

A user at any station in the process control laboratory has access to A/D and D/A channels, a process interrupt switch, digital inputs (two-position switches) and

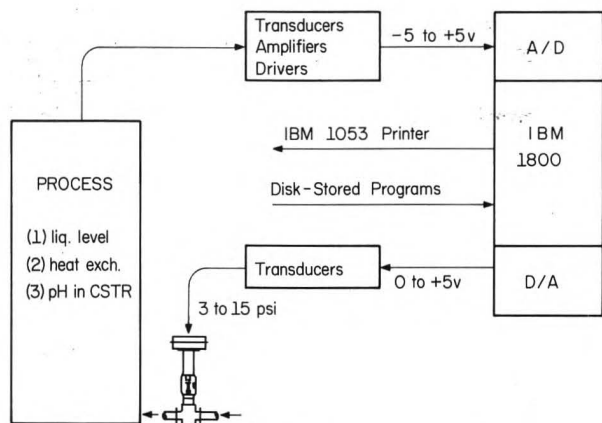


Fig. 1. Schematic Diagram of Closed-Loop Systems

digital outputs (lights). Altogether there are eight A/D and four D/A channels, nine lights, and nine switches. The switches and lights allow for user inputs to the stored programs and for user options.

To gain access to the disk-stored programs for the process control laboratory, the user at any station must activate the process interrupt switch. This causes the transfer into core and the execution of an initializing program which governs the building of the core load for the process control laboratory. The core load consists of a supervisory program along with associated subprograms and the subroutines used—all initially stored in disk memory. When core-load building is complete, execution of the supervisory program begins. It first instructs the computer to flash two lights at all stations in the laboratory repeatedly, waiting for the user to select, by means of switch settings, a desired subprogram. As it is presently written, the supervisory program can accommodate twenty-seven different subprograms each having a different three-digit code number. To call a desired subprogram, the user enters the three integers in the appropriate code number successively in binary form using the two switches at his experiment station. After a three-digit code is entered, the supervisory program branches to the subprogram stored under that code. Upon completion of the subprogram, control is returned to the supervisor, lights are again flashed at all stations waiting for another subprogram to be called.

In the usual operation of the laboratory, the initialization procedures are handled by the instructor or the laboratory assistant. The student then encounters the system and takes over its operation with the supervisory program being executed; that is, with lights flashing in wait of a subprogram code to be entered. The students are responsible for the writing, disk-storing, and debugging of the subprograms. They ordinarily work in three-man groups, and each group is assigned a subprogram code number for the course. The subject of programming will be taken up in more detail in a later section of this paper.

Only one subprogram can be executed at a time as the system presently stands. This means that no two experiments in the control laboratory, both using the computer, can be carried out simultaneously. We can make programming modifications to remove this limitation if it becomes necessary to do so.

As mentioned earlier, the computer operates on a time-sharing basis. For background jobs, that is those not involving real-time applications, each user's core load resides in the core and is executed for a short time before being "swapped" to disk-storage for a period of time. Our system monitor program presently allots a time slice of twenty seconds for core residence time. Much of the computer usage in the process control laboratory falls into this background category. Those programs, however, which make use of the real-time clock; that is, those which call for frequent interrupts for reading and/or sending analog signals at specified intervals of times, have priority status and are not swapped from the core. The computer services these frequent interrupts as a foreground task, returning to background jobs in the interim periods.

## THE LABORATORY

The equipment at each of three of the four stations in the process control laboratory comprises a simple control loop with a single measured variable and one manipulated variable. One of these involves the control of the liquid level in a cylindrical column packed with Plexiglass spheres; another, the control of temperature of the effluent water from a tube-and-shell heat exchanger; and the third, the control of pH in a stirred vessel. Each of the three setups, which are represented schematically in Figure 1, involves standard measuring and signal transducing devices so as to produce dc voltage signals for the measured variables on the range  $-5$  to  $+5$  volts for input to the A/D converter on the computer. The measurement originates from a process pressure (hydrostatic head)-to-current transducer in the liquid level system, from a thermocouple in the heat exchanger and from a submerged pH electrode assembly in the stirred vessel. The D/A signal from the computer is converted to a current and then to an air pressure signal in the

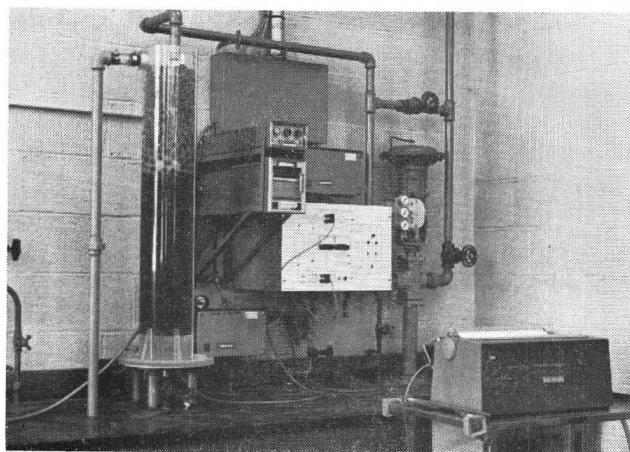


Fig. 2. Photograph of Liquid Level Control System.

range 3 to 15 psi. The pneumatic signal is sent to a diaphragm control valve, the final control element in all three cases. The manipulated variables are (1) the flow rate of water to the packed column of the liquid level system, (2) the steam pressure on the tube side of the heat exchanger and (3) the flow rate of a hydrochloric acid solution to the stirred vessel.

The photograph in Figure 2 shows the physical layout of the liquid level system; the others are similar. As shown in the photograph, each station is equipped with a control panel on which the process signal flow is printed. A close-up view of the control panel for the liquid level process is shown in Figure 3. The panel conceals most of the

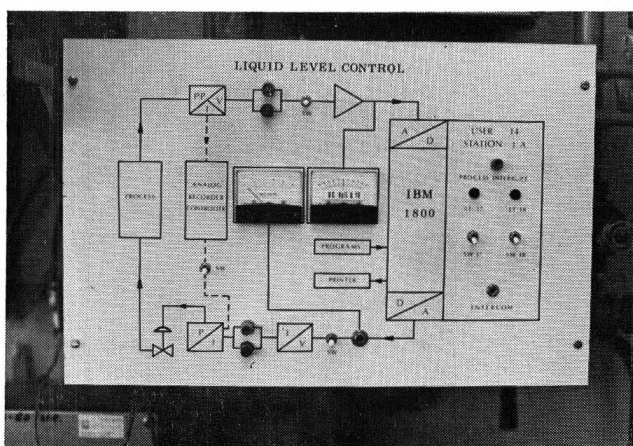


Fig. 3. Operator's Control Panel for Liquid Level Experiment.

wiring connections and interfacing equipment. As can be seen in the photograph of Figure 3, the panel contains a process interrupt switch, two lights (digital outputs) and two switches (digital inputs). As is also shown in the photographs, each of the three systems is equipped with a conventional three-mode analog controller which can be switched into or out of the loop as desired. The equipment and the wiring connections for these systems are intended to be permanent. In working with these processes, therefore, the student in the laboratory does not assemble apparatus or make any interface connections.

Typical experiments with the three systems may involve the formulation and testing of mathematical models via step response, frequency response and analog simulation methods, and investigations of the closed-loop behavior both with direct digital control and with conventional control. In all experiments the subprograms called by the experimenter may contain instructions to read and store data, send feedback signals, dis-

play experimental results on an oscilloscope, carry out calculations and print results or data on the output printer. Further description of the programming effort required will be given in the next section.

A fourth apparatus, which makes use of the computer facility, consists of two different types of pneumatic control valves and associated signal converting and conditioning hardware. Experiments with this setup, depicted schematically in Figure 4, can be carried out using the control panel at the pH control apparatus. As shown on the left portion of Figure 4, voltage signals from the computer may be sent to the valves, separately or simultaneously, and voltage signals, originating from displacement transducers attached to the valve stems, may be sent to the A/D converter on the computer. In the simplest experiments with this system, the objective would be to study and compare the frequency response of the two valves. In such instances the subprogram called by the student would instruct the computer to send a sinusoidal signal of a specified amplitude and frequency to the valves and to read a voltage signal which indicates the instantaneous valve positions. If he so desires, the student may easily have his subprogram analyze the data and print information to be used in constructing a Bode diagram. He may also have results displayed on an oscilloscope.

With an analog computer inserted in the system, as shown in Figure 4, real-time simulations involving the simulation of some process on the analog computer and incorporating real valve dynamics are feasible. In experiments of this type, the student may study the effects of non-ideal valve dynamics in various control loops by experimenting with simulated feedback systems

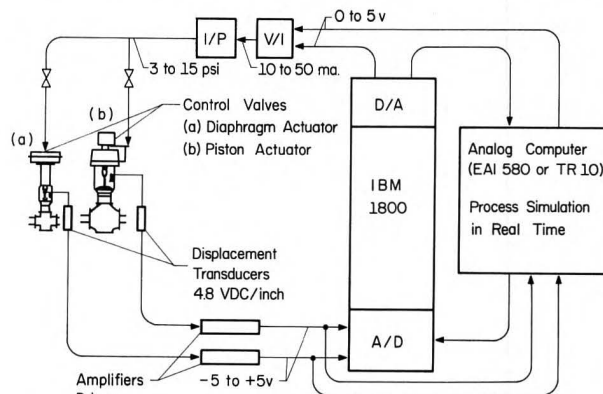


Fig. 4. System for Valve Dynamics Experiments and Real-Time Simulations.

containing ideally simulated valves in one instance and the real valves in another. It will also be possible in advanced studies to interconnect the analog computer, digital computer and any or all of the laboratory stations to form a complex multivariable system containing some simulated components and some real ones. A TR-10 analog computer is available in the laboratory for some small scale experiments, and an EAI 580 is housed at a separate station for more extensive simulations. For either computer, the linkage from the patch panel on the analog machine to the cables from the IBM 1800 is made through a portable interface panel. The panel provides access to all D/A and A/D channels to the IBM 1800 and to all nine lights and switches. The photograph in Figure 5 shows an interconnected system comprising the TR-10 analog computer, the IBM 1800, the interface panel and a pneumatic control valve. This system was set up to demonstrate a real-time simulation of a chemical reactor under direct digital control.

The fourth station in the process control laboratory consists simply of a laboratory bench at which the portable interface panel can be connected to the cables from the computer. The purpose of this station is to provide a versatile general-purpose facility at which a variety of bench-top experiments or demonstrations can be set up and carried out in a short time, some perhaps on a one-shot basis. Usually these experiments will involve studies of the dynamics of individual components or devices. A simple example is a study of the dynamics of thermocouples with



Fig. 5. Photograph Showing a System Consisting of a Pneumatic Control Valve, a Portable Interface Panel to the IBM 1800 Computer and a TR-10 Analog Computer being used in a Real-Time Simulation by an Undergraduate Student.

TABLE I. SUBROUTINE DESCRIPTIONS

Subroutine	Function
ANDIG	Read and print the instantaneous voltage on A/D channel(s) specified in arguments in the calling statement.
ANLOG	Periodically read A/D channels, send voltages on D/A channel(s), (see options below) and store data. The sampling period and channel numbers are specified in arguments in the calling statement. Options: (a) data logging, no D/A signals (b) data logging with proportional D/A signal for feedback control (c) data logging with function generation (D/A voltage) for analog simulations (d) combination of (b) and (c).
DIGAN	Send and hold a voltage on D/A channel(s). The magnitude of the voltage(s) and the channel number(s) are specified arguments.
DISPY	Send voltage data stored in INSKELE common arrays (e.g. stored during real-time operations) for oscilloscope display (i.e. cycle through the arrays). The arrays to be displayed and the variables for the x and y axis are specified arguments.
FLASH	Flash lights and wait for switches (digital inputs) or process interrupt to be set before proceeding. Light number(s) and flashing frequency are specified arguments.
FEROD	Send sinusoidal voltage on a D/A channel and log data from A/D channel(s). The frequency and period of the sine function and the channel numbers are specified parameters. An option in the subroutine permits the user to specify some other periodic function (tabulated) in place of the sine function.
PRINT	Print voltage data stored in INSKELE common arrays (e.g. stored during real-time operations). The arrays to be tabulated are specified in arguments in the calling statement.

various types of shielding, the computer being used to read, store, print and display the thermocouple signals versus time.

As previously indicated, an oscilloscope with an attached Polaroid camera, is available in the laboratory. Thus, by employing a display subroutine, as described in the following section, a student can immediately observe his experimental data and obtain a photograph for insertion in his laboratory report.

## PROGRAMMING

In order to keep student programming requirements and de-bugging difficulties at a manageable level, we have written a number of basic programs for the process control laboratory and have stored them permanently on one of the disks. In addition to the initializing and supervisory programs which are transferred into core memory during initialization procedures, these include a number of subroutines whose functions include reading voltages, sending feedback signals, storing, displaying and printing data and governing lights and switches. A list of these routines along with a very brief description of each is given in Table I. A complete description of these routines is not essential here, but perhaps a few comments, in addition to those given in Table I, on some of them are called for. Each subroutine is accessible through a standard Fortran call statement that includes a list of argu-

ments. Three of the routines, namely ANLOG, DISPY and PEROD, handle real-time operations. Those routines subsequently branch to various entry points in an assembly language program, also written by us specifically for the process control laboratory, which resides permanently in the core memory initialization. Thus by means of Fortran calls to these three subroutines, the user has access to a variety of real-time operations. The initiation and termination of the real-time operations of data logging and sending by the computer in ANLOG and PEROD are handled by the student at the experiment site by means of switch settings. For example, when subroutine ANLOG is entered, a light is flashed (the light number is specified in the list of arguments). Periodic voltage readings and sendings begin when the corresponding switch is turned on. The reading and sending end and the execution of the next statement in the student's subprogram takes place when that switch is turned off. In the meanwhile, the user can call for the storing of data over intervals of time during which a second switch (the number of which is also specified in the argument list) is turned on. The usage of subroutine PEROD is very similar.

It should be noted that the subroutines in their present form do not permit the changing of parameters during the course of real-time operations. Thus the values of the proportional band, sampling interval, etc., are specified prior to the call or in the calling statement of ANLOG and cannot be changed during the execution of that subroutine. In order to study various values of these parameters, the student would either have to call his subprogram repeatedly or have inserted his call statement within a loop for repetitive calls.

As mentioned earlier, the student is required to write the Fortran subprogram to which the supervisory program branches when a certain combination of switch settings is entered. Because of the available subroutines, this programming requirement is simple. For example, the following sequence of statements constitutes the set of necessary instructions for executing direct digital control with data logging, followed by an oscilloscope display and printing of stored data:

```
CALL ANLOG (parameters)
CALL DISPY (parameters)
CALL PRINT (parameters).
```

A study of frequency response would require the substitution of "CALL PEROD (parameters)" in place of the call to ANLOG. Of course, the student may make his program quite involved to the extent of providing various experimental options and of carrying out calculations with the experimental data if he so desires.

Though the minimal programming requirement is quite simple, nearly trivial for some exercises, we feel that it is an important part of the student's usage of the system. It gives him a greater overall appreciation and understanding of his experimental goals, of our particular system and of software aspects associated with on-line computing in general. Having to write a computer program for his experiment in advance of the laboratory session forces the student to study the experiment carefully and to plan his laboratory procedure step-by-step. This in itself is a noteworthy benefit of a computerized laboratory in which the students are required to program the experiments.

In the operation of the laboratory, the laboratory assistant or instructor assists the students in disk-storing and de-bugging their subprograms on the afternoon preceding the scheduled laboratory session. Some additional de-bugging is often required during laboratory meetings. As mentioned earlier, the students work in three-man teams, and therefore programming is a group effort. Our curriculum contains a required course in digital computer programming in the freshman or sophomore year. In addition, the students will have used Fortran programming in carrying out assignments in a few courses prior to their enrollment in the process control course.

In our initial or "trial" usage of the system (Fall semester, 1972) the students carried out three computer-aided experiments. We were able to give adequate programming instruction in two 90-minute workshop sessions early in the semester. The sessions included a general discussion of on-line computing and computing control in addition to specific instructions and exercises on the usage of our system. Each student was given a laboratory manual which described the computerized laboratory and contained instructions on the usage of disk-stored subroutines and on the preparation of the required subprograms. It also contained several example subprograms. The workshop sessions were followed by laboratory demonstrations. We found that the preparatory instruc-

tions so given were quite adequate. The students were able to proceed from that stage at the level of independence which we had sought. With some minor revisions in the manual and subroutine functions, we plan to introduce several additional computer-aided experiments for the Fall term, 1973.

### CONCLUSION

In this paper we have described briefly our system for computer-aided experiments in an undergraduate process dynamics and control laboratory at the University of Illinois. We have included some description of its usage, but will elaborate further on the course description and on the specific laboratory experiments and demonstrations in future publications.

In designing the apparatus and the computer programs for the system, we placed much value on retaining basic simplicity so that students, in an introductory course, would be able to understand the operations of an on-line computing facility and be able to use it to advantage on a hands-on basis. We also strived for versatility so that new experiments could be incorporated easily and that utilization of the system could be extended to more advanced topics, to very complex networks and even perhaps to courses on other subjects, such as applied kinetics.

We also felt that it was important to leave the programming of experiments to the student, but we have made available a number of subroutines which make this task relatively simple. There is a strong temptation for instructors in such laboratories to program the experiments completely and reduce the role of the student to that of executing the programs. While such experiments may be elegantly programmed and virtually fail-safe, they may also stifle most of the student's thought input. With the student involvement in programming, even at a very simple level, a greater appreciation of process-computer interfacing is imparted, a greater degree of openness is automatically provided and presumably more thought input, creativity, originality and interest on the part of the student will be realized.

Our facility was employed for the first time in an introductory undergraduate course on process dynamics and control in the Fall semester, 1972. Three computer-aided experiments conducted by the students and two demonstrations carried out by the author of this paper, were

used, and the concensus of opinions of students, assistants and instructor was that the endeavor was highly successful. The student interest and enthusiasm for the laboratory were noticeably greater than they had been in previous years when similar experiments were conducted without the aid of the computer. We plan to introduce several additional experiments and demonstrations for the next offering of the course (Fall, 1973). It will be possible to discuss more definitive results and to present realistic evaluations after a few more semesters of experience. Hopefully, the system we have developed as well as our experiences in implementation will be helpful to others embarking on similar projects.

### ACKNOWLEDGEMENTS

This project received most of its financial support through a grant from the National Science Foundation. Departmental help was provided mainly in the form of graduate teaching assistants and laboratory instructors. Most of the electronic interfacing hardware was designed and built in the electronics shop in the School of Chemical Sciences. The staff of the School's computer lab provided help with much of the basic programming. The author would like to acknowledge particularly the valuable assistance of Mrs. Pat Anderson with programming and the work of Dr. Ming Fang who designed and installed much of the hardware and apparatus.

### BIBLIOGRAPHY

1. Christensen, J. H. and P. M. Vargo, "Education in Real-Time Computing", *Chem Engr. Ed.* 5, 30 (1971).
2. Fisher, D. G., "Real-Time Computing in the University", *Chem. Engr. Ed.*, 5, 24 (1971).
3. Westerberg, A. W. and R. C. Eschenbacher, "A Real-Time Computer Control Facility", *Chem. Engr. Ed.*, 5, 32 (1971).
4. Wright, J. D., "Education in Computer Control: How to Make Your Real-Time Clock Tick", *Pulp and Paper Magazine of Canada*, 72, 4, 29 (1971).
5. Idier, M. and D. A. Mellichamp, "Computer Monitoring and Control of a Process Dynamics Laboratory", Paper No. 4b, 71st National Meeting of AIChE, Dallas, Feb., 1972.
6. Wissler, E. H., "Computer Aided Methods for Chemical Engineering Laboratories", Paper No. 4a, 71st National Meeting of AIChE, Dallas, Feb., 1972.
7. Moore, C. F., "A General Purpose Data Acquisition and Control Utility," 7th Annual Conf. on Use Digital Computers in Process Control, Louisiana State University, Feb., 1972.
8. Elzy, E., L. B. Evans, R. C. Weaver and A. W. Westerberg, "Real-Time Digital Computer Systems in Undergraduate Education, Paper No. 47c, 72nd National Meeting of AIChE, St. Louis, May, 1972.
9. Secrest, Don, "Time-Sharing Experimental Control on a Small Computer", *I&EC* 60, 6, 74, 1968.