reader,[2] and one involving software design of the time-sharing monitor[4]. Current work in progress involves hardware design and construction of the RAT, and software design of the BASIC time-shared interpreter. In the Fall work will begin on interfaces for the NCR thermal printer and removable-media disk, and on operating-system software for the disk.

Undergraduate class projects have included software for paper-tape editing, card-to-tape conversion, and relocatable assembly; and hardware for drive and interface circuits for a salvaged Teletype 60 cps paper tape punch, as well as preliminary work on the RAT. In our experience, satisfactory completion of such projects requires more time than students can devote to a single class project. Additional time may be provided for interested students to complete their projects through independent-study or senior-thesis courses.

CONCLUSIONS

WE BELIEVE THAT this facility is making a substantial contribution to the education of engineers at the University of Oklahoma: First, through providing a flexible and innovative approach to undergraduate experimentation; and second, by providing graduate and advanced undergraduate students experience in designing hardware and software for on-line computing within both economic and time constraints.

REFERENCES

1. NSF Instructional Scientific Equipment Grant GY 6467.
2. J. Egbert, M.S. Thesis, University of Oklahoma, Norman, 1970.
3. J. I. Norris, M.S. Thesis, University of Oklahoma, 1970.
4. A. V. Kalyansundar, M.S. Thesis, University of Oklahoma, Norman, 1970.

At the University of Florida

A REAL TIME COMPUTER CONTROL FACILITY

A. W. WESTERBERG
R. C. ESCHENBACHER
University of Florida
Gainesville, Florida 32601

IN 1968 THE CHEMICAL ENGINEERING Department at the University of Florida considered various alternatives by which it could introduce computer control into its undergraduate laboratories. The option available ranged over rather large systems at $150,000 or more to a relatively inexpensive remote terminal system in the under $20,000 range.

The choice ultimately made was for the remote terminal (See Figure 1), an IBM 1070 terminal, principally for its low cost. We also determined that, by designing our own interface equipment, we could have equipment which would generally satisfy our laboratory requirements. One of these requirements was that the total equipment cost no more than $30,000 as that amount was available. If the equipment were more expensive, outside financing would be needed with all its inherent delays.

We also desired to have an easily programmable system which would not first require significant software development on our part. The terminal would tie directly to a large scale scientific computer, an IBM 360/65, and could be operated by FORTRAN calls. With not too extensive a software system design, it was apparent that we could have a very easily programmed system. The power of the scientific computer would also permit complex control and/or analysis algorithms to be tried. We would obviously need the cooperation of the computing center for the quick computer response necessary for control. The programs we would write would use little actual time, but, when the process required attention,

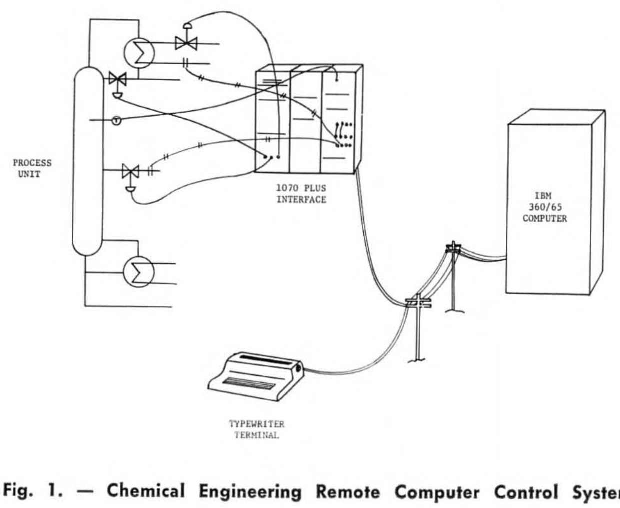


Fig. 1. — Chemical Engineering Remote Computer Control System.

instantaneous response without interruptions would be needed. We were promised the highest priority in the computer while operating but with the penalty that we could only operate at limited hours (about 3 prime time hours plus the late-late shift) each day.

We also found with the terminal that we could only have low speed input and output, about four random inputs or outputs per second or up to 20 sequentially scanned inputs or outputs per second. These rates are more than adequate for most undergraduate experiments. Such things as direct digital control of flow loops were of course ruled out. The equipment could be purchased with a 13 bit analog to digital converter permitting approximately 1 part in 8000 resolution. We determined that we could measure thermocouple signals with a resolution of about 7 microvolts.

The final general requirement was that the system could run one large experiment or several smaller ones at the same time. Again, by choosing to build our own interface, we could construct a panel that could be easily patched to any process. The computer equipment had to be adequate to handle only one of our largest experiments, currently a distillation column or a double effect evaporator. If it were totally portable, it could also reach any experiment without the usual worries about microvolt signals from thermocouples traveling long distances.

THE HARDWARE

Figure 2 is a diagram of the remote terminal plus interface equipment. It is all contained in three 19 inch racks 72 inches tall, which are bolted together and mounted on wheels. The non-process connections to the terminal are via a single 110 volt AC plug for power and a single pair of voice grade telephone lines to the computer. Table 1 lists the input/output facilities provided. Inexpensive 2 and 3 prong plugs and outlets permit one to patch a process simply into the terminal using any selection of the available terminal facilities. For thermocouple inputs special commercially available copper-constantan jacks are provided.

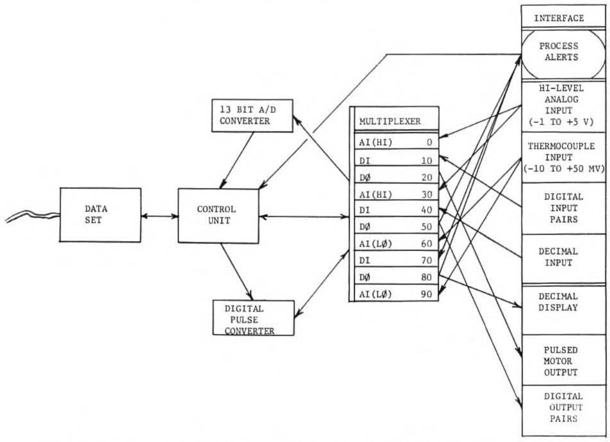


Fig. 2. — 1070 Hardware Plus Special Purpose Interface.

TABLE 1 — INPUT/OUTPUT FACILITIES OF TERMINAL

1. 13 BIT A/D (1 PART IN 8000)
2. 66 CHARACTER/SEC COMMUNICATION LINK
   RANDOM INPUT/OUTPUT, 4 PER SECOND
   SEQUENTIAL INPUT/OUTPUT
   20 ANALOG PER SECOND MAX
   60 DIGITAL PER SECOND MAX
3. DIGITAL PULSE CONVERTER, 96 PULSES/SECOND
4. 7 PROCESS ALERTS
   15 DIGITAL INPUT PAIRS
   12 DIGITAL OUTPUT PAIRS
   10 PULSE OUTPUT CHANNELS
   16 THERMOCOUPLE INPUTS, —10 TO +50 MV
   18 ANALOG INPUTS, —1 TO +5 V
   8 PRESSURE TO VOLTAGE TRANSDUCERS
   4 PRESSURE RECORDERS
   1 DECIMAL INPUT, 6 DIGITS
   1 DECIMAL DISPLAY, 4 DIGITS

The process alerts provide a form of hardware interrupt capability of the computer by the process. A conditional read of the terminal from the computer is available. When used, no terminal response occurs until a process alert contact is closed. During the wait, the computer is free to service other users of the computer.

On the interface the digital inputs and outputs are paired, although they can be used individually. The intended and admittedly redundant use of each pair is to give a positive signal for the two desired states (open or closed) of a digital input or output.

Tied to each output pair are a red and a green light to indicate the state of the pair. Each output is provided in two forms, either as a simple switch closure or as a 110 volt signal, when closed.

The interface is designed to ease control program development. Every input to the computer can be individually simulated by equipment build into the interface. Also a toggle switch permits one to drive manually each output pair, overriding the computer. Thus control programs can be run with the process, any part of the process, or in fact none of the process tied to the terminal. The process alerts can be manually set with momentary push buttons. A process/manual switch on each digital input pair permits one to have the pair tied to the process or to a toggle switch on the interface. Associated with each analog input is a toggle switch and a DC voltage signal passing through an inexpensive potentiometer. The signal is supplied by a power supply. The toggle switch permits the analog signal source to come from the process or from the adjustable internal source.

Other equipment in the interface includes a hardware poller, eight pressure to voltage transducers, and four pressure signal recorders. We designed and built the poller using a small reversible motor, two micro switches, and three relays. This device periodically closes the first process alert; the period is adjustable from 2 to 60 seconds. The pressure to voltage transducers permit 3 to 15 psi air signals to be fed into the terminal. Associated with the air inlet is an electrical outlet with the equivalent voltage, and this voltage is then easily patched into any of the high level analog inputs.

The final auxiliary equipment is an operator's panel comprising an IBM 1075 four decimal digit display and a six decimal digit manual input device we built using simple 10 position rotary switches. Typical use of the panel involves setting up a six digit number in the rotary switches and pushing the button associated with process alert seven. Responses to the input can be displayed on the digital display. The display is also used to indicate errors as they occur.

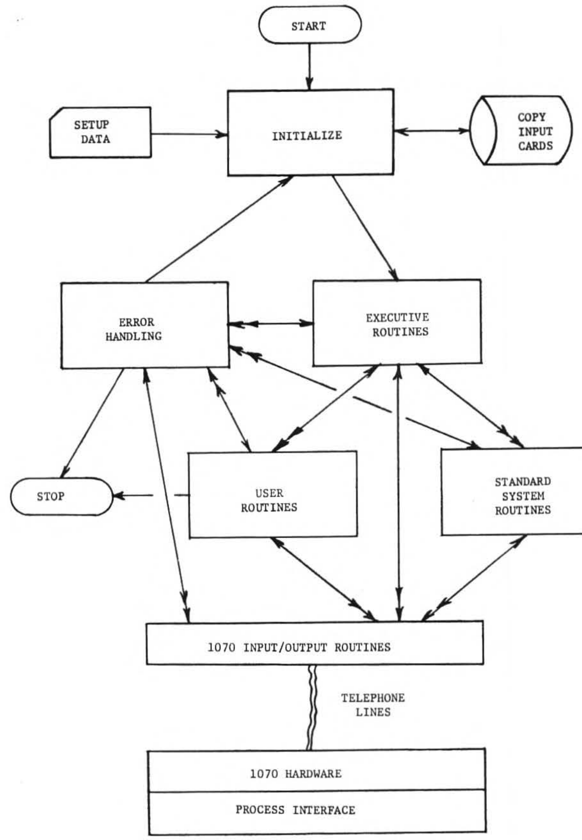


Fig. 3. — Software System for GIPSI.

THE SOFTWARE

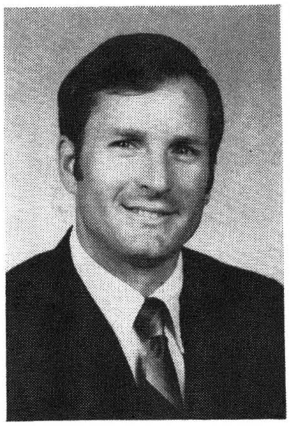
Figure 3 is a diagram of the interrelationships of the major routines developed for process monitoring and control using our terminal. All programs are written in FORTRAN except the IBM provided 1070 input/output routines. The heart of the programs are the executive routines which are given in more detail in Figure 4. The only routines a user needs to supply are the User Routines on Figure 3, part of the Execute program on Figure 4, and a special user data input routine if desired.

The executive routines in Figure 4 are five subroutines which use the indicated three common data tables. The Process Alert Handler issues a conditional read and the control programs stop execution until a process alert on the 1070 is closed. The process alert(s) which is closed is identified and the corresponding response program is put onto the Execution Stack by priority using the program Stacker. Control is then passed to Execute, part of which is user supplied. Any programs on the Execute Stack are removed in order and called. If Process Alert 1 started the sequence, its response program, CLOCK, is on the Execute Stack and is thus called. Its job is to remove programs from the Delay Stack if their time is up and put them on the Execute Stack again using the program Stacker. Times are always compared to the computer clock. Control returns to Execute which continues to remove and cause all programs on the Execute Stack to be executed. When the Execute Stack is empty, control returns to Process Alert Handler which starts the cycle over again.

Data on program priorities, standard time delays for execution, and so forth are kept in the Program Descriptive Data. Program Delay Stacker is used rather than the program Stacker by any routine desiring a time delay before execution.

With each request for execution stored on either the Execute Stack or Delay Stack, a single passed parameter is also stored. This parameter provides the essential communication link between the requesting and the requested programs.

We will illustrate the ease of using the terminal with an example.



Arthur W. Westerberg is currently an assistant professor in ChE at the University of Florida. He obtained his education at the University of Minnesota, Princeton University, and Imperial College, London, finishing in 1964. He then worked two years for Control Data Corporation, in their process control division in La Jolla, California, before coming to Florida in 1967.

His teaching interests include Fortran and numerical analysis, undergraduate and graduate process design, and classical and optimal control. His research is in the area of computer aided design and computer control of processes. (Left photo).

R. C. Eschenbacher is a graduate of the University of Florida (BSChE, MSE, and PhD'70). He is now employed as an engineer with the Humble Oil and Refining Company in Baytown, Texas. His interests are in the area of process modeling, computer control, and optimization.

SAMPLE PROGRAM

For this example we would like to use analog input 1 as a 0 to 5 Volt voltmeter. The signal on this input is to be read and displayed every 10 seconds if digital input pair 1 is on. If digital input pair 1 is off, no updating is desired. Also, program execution is to terminate if the button for process alert 2 is pressed.

The steps to implement this example are first to set the poller to a 10 second interval. Then the FORTRAN subroutines given in Figure 5 are written. Subroutine VLTMTR is essentially self explanatory. Subroutines DELAY, READ1, CALIB, and DISPLY are standard routines in the Executive and Standard System packages.

Subroutine GOTO is the user supplied portion of the Execute program indicated on Figure 4. It provides the system calls to each active subroutine. Routines 1 to 7 are always in response to process alerts 1 to 7 respectively. Only process alerts 1 and 2 will be active here. Other preassigned program ordinals are for demand functions requested via the operator's console, and for changes of state for digital input pairs. Programs 127 and higher are other user or standard system subroutines. In our example program 127 is SETCLB, a system subroutine which periodically reads in standard voltages and then calculates new calibration constants for the A/D input system. Program 128 is our user written subroutine, VLTMTR. All other ordinals are ignored—for example, if the button for process alert 3 is pressed, the system will ultimately pass ordinal 3 to GOTO which will ignore it.

Subroutine QUIT is again self explanatory.

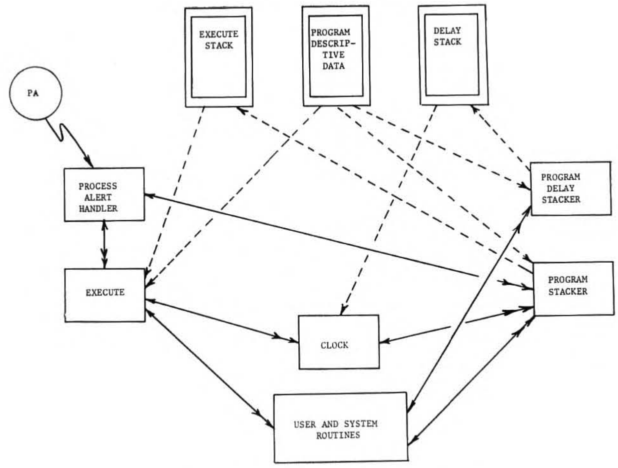


Fig. 4. — Executive Routines for GIPSI.

```
      SUBROUTINE VLTMTR (IPROG, NU1, NU2)
      DIMENSION INPUT (40), INARY (40)
C        RESCHEDULE EXECUTION IN 10 SECONDS
      CALL DELAY (IPROG, 0)
C        READ DIGITAL INPUT 1
      CALL READ1(10, 1, 1, 7, INPUT, INARY)
C        IF DIGITAL INPUT NOT ON, RETURN
      IF (INARY(1).EQ.0) RETURN
C        READ ANALOG INPUT 1
      CALL READ1(0, 1, 1, 7, INPUT, INARY)
C        CALIBRATE SIGNAL TO HIGH LEVEL REFER-
        ENCE VOLTAGES
      CALL CALIB (INARY(1), 1, 1)
C        DISPLAY VALUE
      CALL DISPLY (INARY(1))
      RETURN
      END

      SUBROUTINE GOTO (I, J, K)

      IF (I.EQ.1) CALL CLOCK (I,J,K)
      IF (I.EQ.2) CALL QUIT (I,J,K)
      IF (I.EQ.127) CALL SETCLB (I,J,K)
      IF (I.EQ.128) CALL VLTMTR (I,J,K)
      RETURN
      END

      SUBROUTINE QUIT (I,J,K)
      CALL DISPLY (9999)
      STOP
      END
```

Fig. 5. — Programs for Sample Problem.

At Florida a time sharing and remote job entry terminal system (Figure 1) is implemented on the University Computer. Our department has two IBM 2741 selectric typewriter terminals associated with this system. Our normal method of entering the subroutines just discussed would be to type them directly into the computer using a 2741 terminal. A phone call to the center is needed to reserve top priority space for our programs and then via the typewriter terminal, we have the computer compile our user programs and add them to our standard 1070 software. They are then linked and loaded into core and execution commences.

At the 1070 terminal and interface equipment, we then start the hardware poller whenever we are ready to begin and our terminal is now a voltmeter. The poller can be turned off anytime we wish to put the software program into hold.

DEMAND FUNCTIONS

Several standard programs exist whose execution can be requested via the six decimal digit input device. All input is decoded in a standard form. The first two digits indicate the program ordinal and the last four are data to be passed to the program when it is called. The response routines are called demand functions, and Table 2 lists some of those provided.

TABLE 2—TYPICAL DEMAND FUNCTIONS

1. STOP PROGRAM
2. ERROR RESPONSE
   A. STOP B. CONTINUE C. RESTART
3. CHANGE VALUE OF CORE WORD
4. DISPLAY VALUE OF CORE WORD
   A. ONCE B. PERODICALLY UPDATE
5. PUT ROUTINE ON DELAY STACK
6. TAKE ROUTINE OFF DELAY STACK

DISCUSSION

Table 3 gives a brief summary of the system costs and typical core requirements. These costs are quite small. The operating costs will increase when we are charged for core space used, a charge not now implemented. The core required would be equivalent to a 50,000 word minicomputer with 16 bit words, certainly a large minicomputer. It is however only 10% of the University Computer's core. We can reduce the requirements substantially by removing portions of the software not needed for a particular control program.

TABLE 3—SUMMARY OF COST DATA

EQUIPMENT FROM IBM	
DATA SET	
CONTROL UNIT	$17,000
13 BIT A/D CONVERTER	
DIGITAL PULSE CONVERTER	
100 MULTIPLEXER POINTS	
DIGITAL DISPLAY	
NONSTANDARD EQUIPMENT	
CABINETS	
RELAYS, SWITCHES, LIGHTS	$ 4,000
POWER SUPPLIES	
TRANSDUCERS	
OPERATING STATISTICS	
CONTROLLING A DISTILLATION COLUMN BY ADJUSTING 3 SETPOINT CONTROLLERS	$3-5/HR
CORE SPACE	100,000 8-BIT BYTES

A detailed description of the system is available from the department in the form of a pair of manuals on the system [1,2].

REFERENCES

1. Eschenbacher, R. C., Software Manual for GIPSI, Department of Chemical Engineering, University of Florida (1969).
2. Eschenbacher, R. C., Hardware Manual for GIPSI, Department of Chemical Engineering, University of Florida (1969).

SMITH: Photodecomposition

(Continued from page 22)

ν	frequency of radiation, sec^{-1}
λ	wave length
μ_λ	attenuation coefficient of pollutants in water, cm^{-1}
Ω	rate of reaction, g $moles/cm^3$-sec
η	overall efficiency of utilization of energy input to lamp

REFERENCES

1. Joyce, R. S. and Sukenik, V. A., "Feasibility of Grannular, Activated Carbon Adsorption for Wastewater Renovation," A.W.T.R.-10, Public Health Service Pub. No. 999-WP-12 (1964).
2. Masse, Arthur N., "Removal of Organics by Activated Carbon," presented at 156th National Meeting, Amer. Chem. Soc, Atlantic City, N. J., Sept. 1967.
3. Painter, H. H., Viney, M. and Bywaters, A. J., Inst. of Sewage Purification, Part 4, p. 302 (1961).
4. Matsuura, T. and Smith, J. M., A.I.ChE J. 9, 252 (1960).
5. Calvert, J. G. and Pitts, J. N., Jr., "Photochemistry," John Wiley and Sons, New York (1966).
6. Schorr, V., Boval, B., Hancil, V. and Smith, J. M., "Photo-oxidation Kinetics of Organic Pollutants in Municipal Wastewater," submitted to Industrial and Engineering Chemistry.