Behavior Research Methods & Instrumentation
1974, Vol. 6 (6), 557-564

COMPUTER TECHNOLOGY

An automated system for primate instruction for behavioral neurophysiology

SAMUEL L. MOISE, JR., DAVID E. OLSEN, and STEVE W. HUSTON
Space Biology Laboratory, Brain Research Institute, University of California, Los Angeles, California 90024

This paper describes a flexible hardware/software system developed for use with a PDP-8I computer for conducting research in behavioral neurophysiology. A real time monitor was designed to facilitate development, debugging, and modification of programs to run experiments. It relieves the programmer of the burden of dealing with hardware dependent functions such as interrupt handling and input/output. In addition, it provides the user with a large library of callable routines to perform functions commonly needed for conducting experiments. The monitor is modular in design and could be expanded or modified for use with many configurations of the PDP-8 family of computers.

In recent years, there have appeared numerous descriptions of hardware and software systems for facilitating the use of minicomputers for psychological and physiological research (Doll, 1972; Gips, Pfefferbaum, & Buchsbaum, 1971; Millenson, Kehoe, Tait, & Gormezano, 1972; Moise & Jarrad, 1969; Wallsten, 1972). In spite of the proliferation of these reports, no satisfactory systems had been developed in 1972 that filled our needs for a PDP-8I based system for primate training and data acquisition. Our requirements were for a system which would (1) act as a task controller for a large number of experimental paradigms (operant conditioning to complex decision making behavior), (2) collect behavioral data in a well organized format for subsequent analysis, (3) monitor and record physiological data (EEG, EKG, EMG, etc) for on-line analysis during the experiment, and (4) permit computer control of a potentially large number of experimental parameters (i.e., brain stimulators, noise generators, house lights).

As a consequence, we proceeded with the development of an Automated System for Primate Instructions (ASPRIN). This hardware/software system is quite general and has served in a wide variety of experimental designs from simple buttonpress training to complex "feedback" experiments in which changes in stimulus presentation in a memory task are made dependent upon analysis of moment-to-moment changes in the animal's EEG.

The hardware reflects our need for control of many pieces of equipment and a complete physiological data acquisition system to supplement on-line analyses. The software, while written for our particular hardware configuration, provides a general framework for experimental control and data acquisition that could easily be adapted for use on any PDP-8 or -12 that supports OS/8 or OS/12[1] (8K of core and a mass storage device).

This investigation was supported in part by the Public Health Service Research Grant GM-16058-08 and in part by the AF Office of Scientific Research of the Office of Aerospace Research under Contract F44620-70-C-0017.

HARDWARE

The ASPRIN hardware (Figure 1) consists of a primate test chamber with a behavioral test panel, a system console, oscillograph and tape recorder, and a PDP-8/I computer with associated peripheral devices.

The primate test chamber is a wooden enclosure containing space for a restrained animal, a test panel, house lights, and a ventilation fan. The test panel has a 3 x 3 array of stimulus/response windows consisting of transparent Plexiglas response switches. Behind each window is a 1-plane digital projection unit (Industrial Electronic Engineers) that projects any combination of 7 symbols and 5 colors through the switches. Below the windows is a receptacle for delivering food pellet rewards. The test chamber door swings open to permit entry and exit for the animals and has a large one-way glass window for observation of the animals during the experiment.

While the test equipment is configured primarily for monkeys, it is easily adapted to any animal capable of making appropriate motor responses. Prototype test panels have been attached to the home cages of chimpanzees. The present panel may be used for man (adults or children) by removing and placing it on a table and outfitting the pellet feeder to deliver M&M's or other suitable rewards.

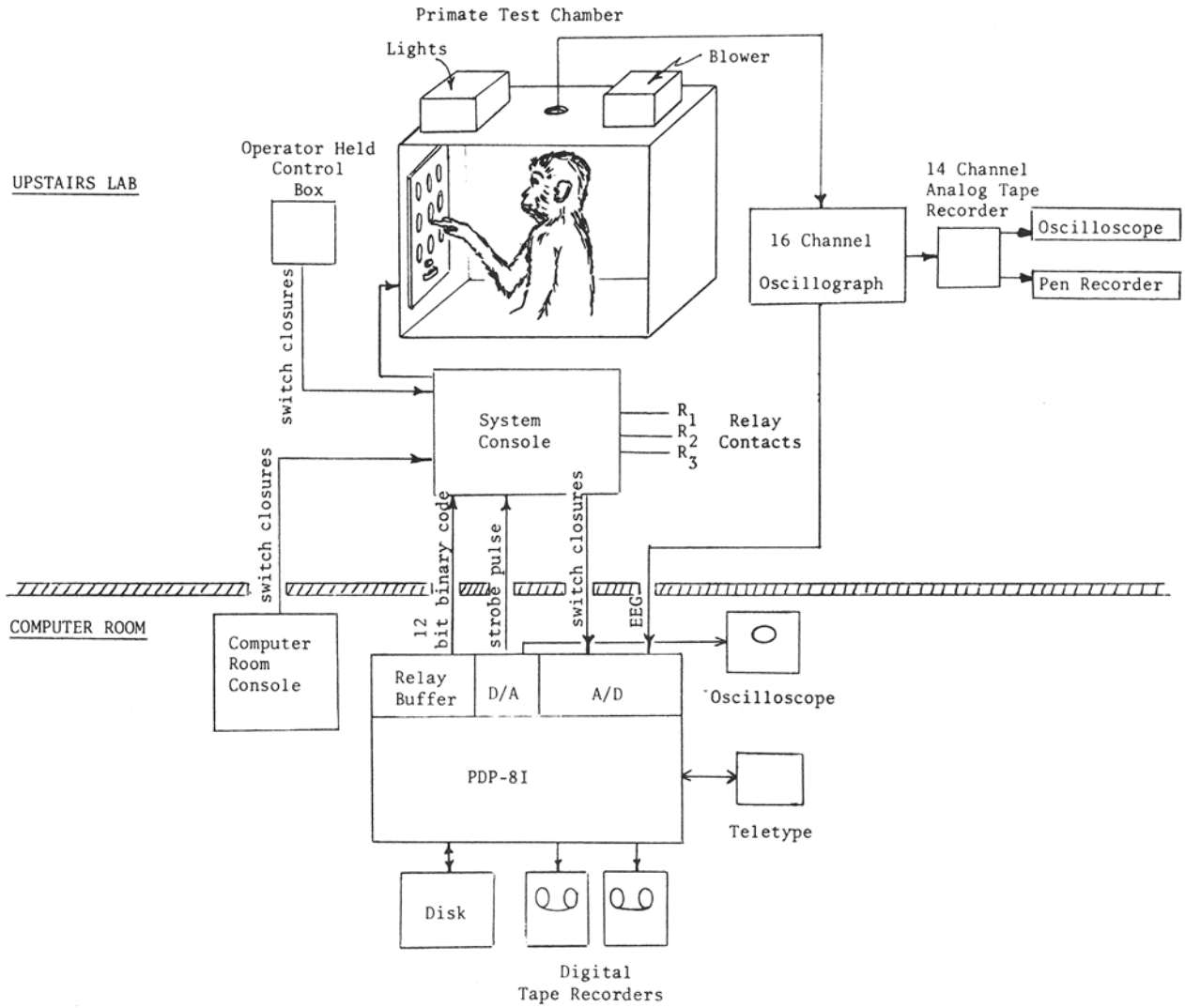

Figure 1. Schematic of the ASPRIN hardware.

The computer is an 8K PDP-8/I with 1.66 million word disk (System Industries, Sunnyvale, Calif.), 7– and 9-track magnetic tape drives, 64-channel analog-to-digital converter, 8-channel digital-to-analog converter, 12-bit relay buffer, Teletype, and experimental apparatus interface. The oscillograph is a 16-channel Beckman Type R dynagraph and the tape recorder a 14-channel Ampex FR1400. Coaxial lines are used to send the amplified signals from the dynagraph to the computer.

In order to be able to select any combination of the 12 stimulus lights in each of the 9 display windows at any given time, the user must be able to select any of 108 unique output lines. Minicomputers commonly have 12 to 24 bits available for output depending upon its word size (the PDP-8 has 12 bits). To control our experimental apparatus a simple, relatively inexpensive, decoder with memory was designed and built using Wyle logic cards (DTL). Eight of the twelve output bits are divided into two groups of four which are decoded separately to provide $2^4 \times 2^4 = 256$ output possibilities. The remaining 4 bits are not presently decoded but are used as four separate output lines for a total of 260 possibilities. Figure 2 is a schematic of the basic decoding logic. Unique output patterns are selected by the computer program by setting the appropriate configuration of the relay buffer, allowing the relays time to settle (approximately 500 μsec in our system) and strobing the pattern into the decoder memory. The strobe in our system is provided by one of the D/A channels, but could just as easily be provided by one of the four undecoded relays. A master relay is provided in the decoder interface so that complex stimulus patterns may be set up in the decoder memory and then presented simultaneously to the subject. In addition,

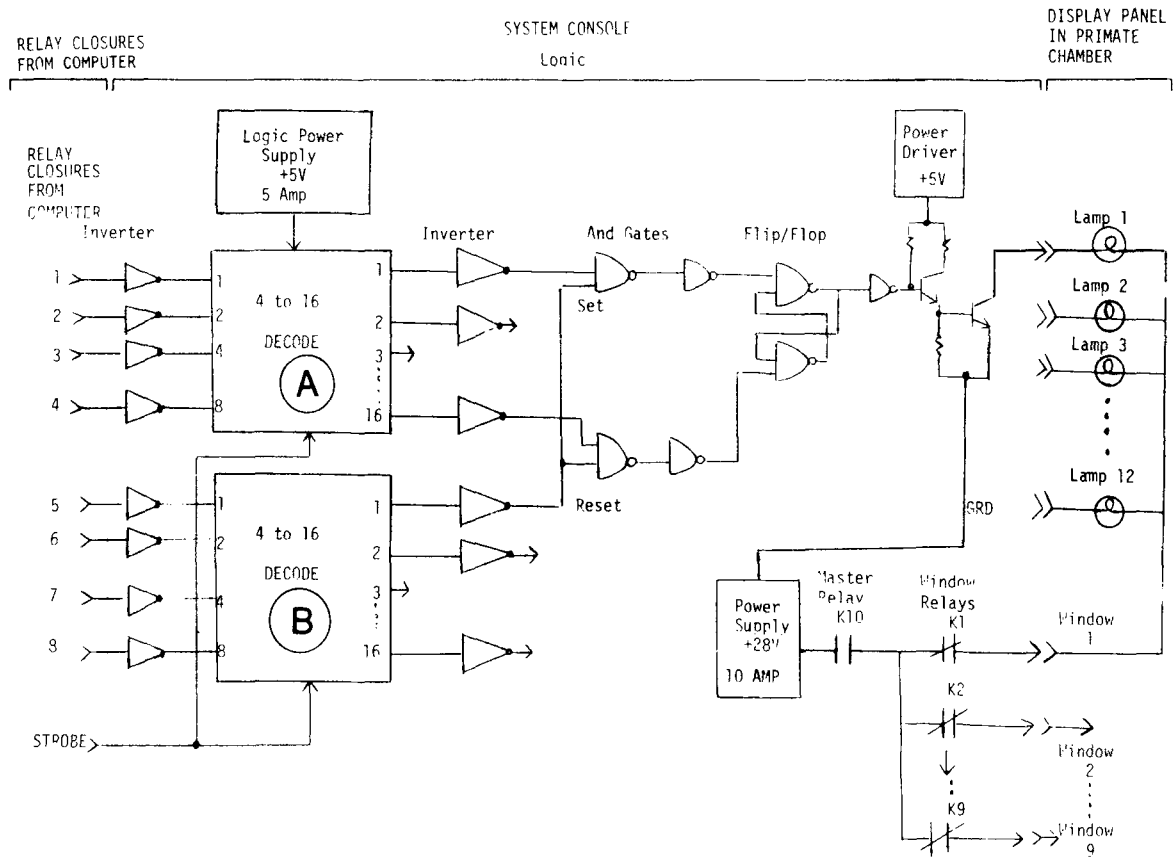

Figure 2. Schematic of typical decoder for activating one lamp on one stimulus-response window in the primate test chamber. The system console circuitry is identical for controlling houselights and additional for stimulators, noise generators, etc.

each stimulus window has a separate relay associated with it so that particular windows may be turned on and off independent of the others. This permits the presentation of frequency ("flickering") stimuli. The interface to the experimental room also controls house lights and provides sets of relay contacts for operation of stimulators, noise generators, etc.

SOFTWARE

The software for a system such as ours must make program development, debugging, and modification as rapid as possible. While several higher-level languages exist for writing real-time experiments (e.g., PSYCHOL, McLean, 1969; SCAT, Grason-Stadler, 1970), they are often inadequate for many experimental designs or are closely tied to machine configurations that were not available to us.

Recently Digital Equipment Company released real-time versions of FORTRAN IV and BASIC that take advantage of the facilities of OS-8. The real-time functions of these languages are configured only for the PDP-8E (and PDP-12 in the case of FORTRAN) laboratory peripherals. Adaptation of FORTRAN for nonstandard peripherals is a demanding task, as the run-time system is structure around characteristics of the laboratory peripherals. In the case of BASIC it is possible to write straightforward assembly language routines to drive nonstandard equipment. However, the overlay scheme used by OS-8 BASIC may be limiting for experiments that demand high rates of data acquisition with on-line evaluation.

Given that assembly language programming is still a necessary evil for many situations, the ASPRIN monitor was developed to facilitate writing new experiments by relieving the need for the programmer to deal with hardware dependent functions such as interrupt handling and input/output. It provides for a wide range of user callable functions commonly needed for controlling behavioral tasks and collecting physiological data for on-line analysis. Many of the functions of this monitor were suggested by the excellent set of user functions in FIASCO (Computer Controlled Psychology Laboratory, Carnegie-Mellon University, 1971).

Unlike FIASCO, the ASPRIN monitor is a single task controller, but was written with the idea of converting it for multitasking. However, there are excellent arguments against time-sharing minicomputers used for experimental control (Uttal, 1972). This is particularly true when large quantities of physiological data must be

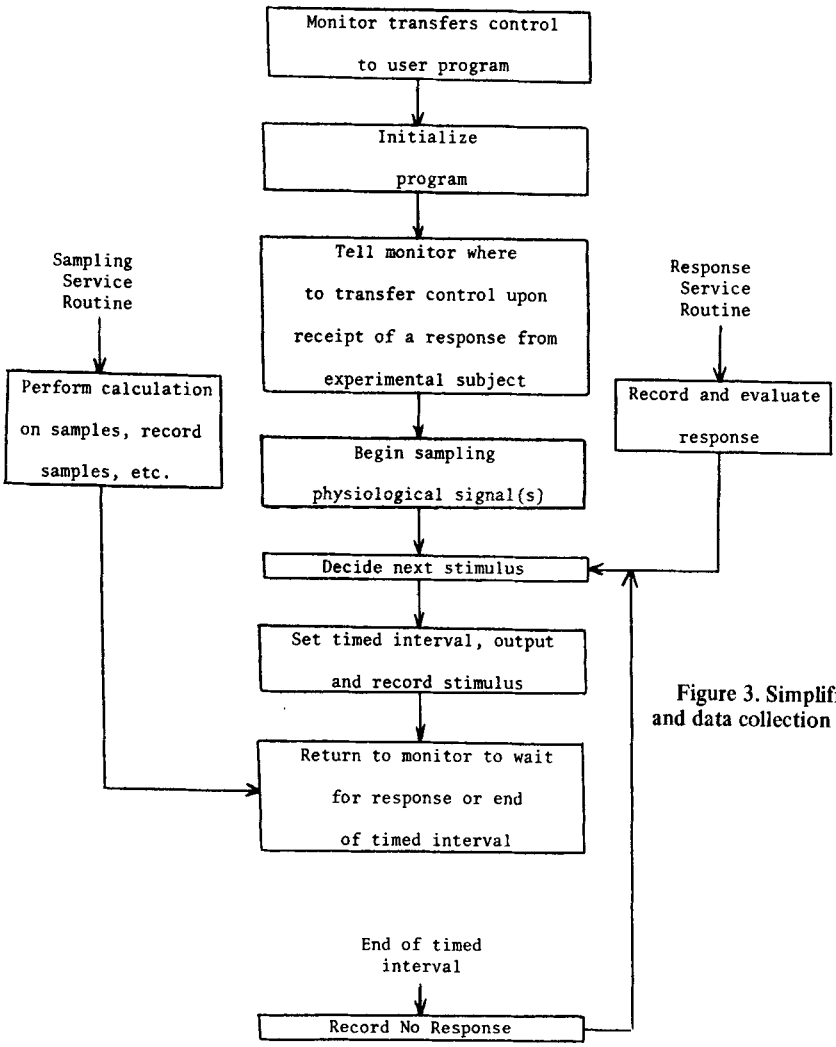

Figure 3. Simplified flow chart of a basic experimental control and data collection program.

collected or if real-time evaluations of even moderate quantities of physiological data must be made. During sophisticated experiments this processing can occupy nearly all the available CPU time leaving very little for the control of additional experiments.

The monitor performs all internal calculations in floating point by means of a modified 27-bit floating point package (Rothman, 1970). Modifications include double and single precision float and fix routines, and floating point to ASC11 conversion with integer formating. The entire package is 5-1/2 pages long and may be used in a stand-alone version appended to the user's program.

User Interaction With the Monitor

Programs to run under the monitor may be assembled with any of the PDP-8 Pal or Macro assemblers along with definitions for the monitor callable routines. At run time, the monitor occupies all of field 0 and location 20-50 of each field above 0 which contains a user program. User programs reside in Field 1, or higher if more than 8K of core is available.

When started, the monitor types its name, version, and the current date. It then types MKC> and waits for a user response. A number of debugging, and core modification functions are available to the user for testing programs, modifying them, or preparing to run the experiment. Many of these functions are similar to those in OS/8 ODT (Octal Debugging Technique). Table 1 lists the keyboard commands and their functions. The Q command can be used to initialize the random generator to a new value before starting an experiment and the T command provides a quick check that the experimental apparatus interface is working properly. Once the user passes control to the experimental program via the S or G command, the monitor ignores all keyboard input except ↑X (control X). Receipt of ↑X at any time halts the user program and immediately returns control to the monitor keyboard controller.

User programs typically consist of a series of monitor calls with appropriate logic for presenting stimuli, determining if responses are correct, evaluating response latencies, processing physiological data, or whatever is

Table 1
Monitor Keyboard Controller Commands

Command	Function
↑C	Return to OS-8 (control C)
↑X	Return control to the keyboard monitor (control X)
/	Examine and open current core location
< CR >	Store new value for currently open location, if any (carriage return)
< LF.>	Examine and open next location (line feed)
A	Examine contents of accumulator at breakpoint
B	Set or clear breakpoint (FNNNB sets a breakpoint at location NNNN of field F)
C	Continue from breakpoint
D	Dump octal locations (XXXXX-YYYYY; D dumps locations X to Y)
F	Examine field register
G	Go to specified location (FNNNNG will go to location NNNN in field F)
L	Examine link
M	Examine search mask
W	Search for specified word via mask (XXXXX-YYYYY; ZZZZW does a word search for Z from X to Y. Each word searched is ANDed with the mask before being compared with Z)
–	Separator for lower limit
;	Separator for upper limit
S	Start user program at location Ø2ØØ of field 1
T	Test I/O to experimental apparatus
Q	Call random generator (NNNNQ calls the random generator NNNN times)

required by the experiment. Figure 3 is a simplified flow chart of the fundamentals of a typical user program. When the user program finishes immediate processing and must wait for a response, a specified interval, or further acquisition of physiological data, it must pass control to the monitor via the monitor routines XMONITOR or XWAIT. XWAIT eventually passes control to XMONITOR which performs a null background task while waiting. At present the null task is simply a pattern of moving lights in the accumulator but could easily be made into a more complex program such as FOCAL, if an additional teletype and at least 12K of core were available.

Interrupt Handling

The heart of the monitor is the clock interrupt service routine. Interrupts from other devices generally lead to acquisition of data points or identification of external events. Initiation of A/D sampling, evaluation of the status of WAIT intervals, and evaluation of the status of input from the experimental apparatus is managed by the clock interrupt service routine which is activated every millisecond. Figure 4 is a flow chart of the clock interrupt handler.

The initial version of the monitor was designed to begin behavioral training of animals as quickly as possible with minimal collection of analog data (e.g., EEG) until the animals reached a given level of training.

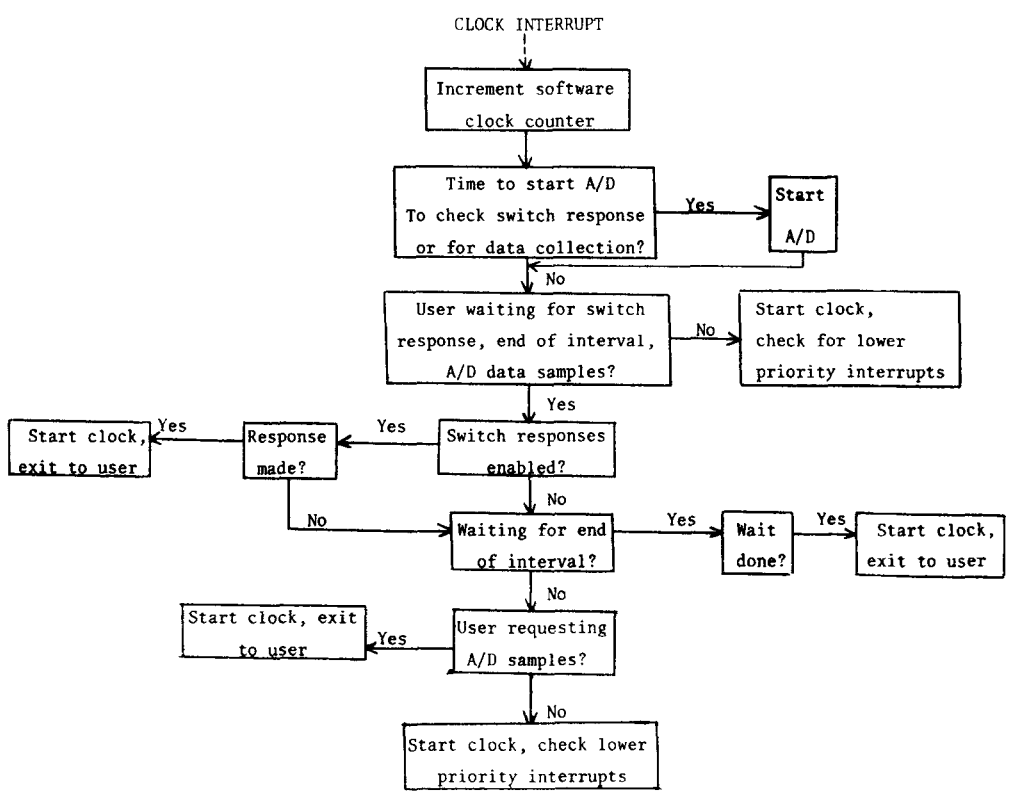

Figure 4. Flow chart of the clock interrupt handler.

Since no interrupting digital module was available at that time, the A/D converter was used to monitor the 12 response lines from the experimental apparatus as well as for digitizing up to four channels of analog data. Because of this restraint, timing was accurate to the nearest millisecond but response servicing was only accurate to the nearest 10 msec. We have since acquired an interrupting digital module that allows a side range of user selectable sampling rates and number of channels to be sampled (up to 20,000 samples/sec), and improves response servicing accuracy to ±1 msec.

Analog Data Acquisition

The user instructs the monitor to begin collection of analog data by calling the routine XSAMPLE. Arguments for this call are the address of a user routine to process the collected data, the address of a ring buffer where the samples are to be placed, the size of the buffer, and the sampling rate. If the number of samples/sec (number of channels X sampling rate) exceeds the system capacity, an error message is printed and control returned to the keyboard monitor. The monitor uses the buffer pointer to fill the buffer with samples as they are collected. The pointer is available to the user so he knows when the latest sample has been put. When the end of the buffer is reached, the pointer is reset and the buffer refilled.

When the user passes control to the monitor to wait for a response, or end of interval, the clock interrupt routine determines if sampling is in progress and exists to the user to process the samples if there are no responses or an end of interval to service. The user service routnine must extract samples from the ring buffer for processing. When finished, the user routine returns to the monitor via XMONITOR.

During involved processing of samples the user may be making calculations at the time a wait interval ends or a response is made. Therefore, two switches are provided in the monitor portion of the field containing the user program (Location 0-50). These switches are normally 0 but are set to 1 if responses are enabled and one is detected (XINTSW) or if a timed interval is over (XWASW). These switches may be interrogated periodically during user computations. If the user finds a switch set and wishes to process the response or end of interval he must transfer control back to the monitor via XMONITOR. Collection of analog data is continuous until the user calls the routine XSAMP1. The user must be sure that his processing of the analog data is on the average complete before the next sample is taken or data will be lost.

Data Output

Output to storage devices (disk, tape, paper tape, etc.) is extremely flexible utilizing the device independent facilities of OS/8 device handlers. The user program opens an output device and file (if file structured) by calling routine XOPEN. This routine calls the OS/8 command decoder for the user to enter the output device and file name according to the usual OS/8 conventions. If an attempt is made to output data before a file is opened, the monitor prints an error message and calls the command decoder for the user to enter the output specifications.

Once the output file is opened, the monitor will accept ASCII and numeric data from the user program to be queued for output. All output is treated as ASCII characters by the monitor. Numeric quantities are converted into ASCII character strings by routines XINTEGER and XFPOUT before outputting them. As characters are received, the monitor packs them in OS/8 format (three characters per two computer words) in a buffer and initiates output when the buffer is full.

When the experiment is finished a call to XCLOSE closes the output file and makes it permanent. If the user returns to OS/8 via a ↑C in response to the monitor keyboard controller, the monitor will close any open output file, clear all hardware flags, and pass control to OS/8 at 7600.

The advantages of having data stored in OS/8 format is sizeable. As soon as the experimental run is complete, the data is immediately available for modification, summary or analysis by any of several OS/8 editing programs (EDIT, TECO) or computational language (FORTRAN IV, FORTRAN II, BASIC, FOCAL-OMSI version). For users with 12K of core, or more, a batch job can be set up to perform long sequences of data storage, cataloging, and analysis. This whole approach can drastically reduce the time for handling, maintaining and analyzing large quantities of data.

No special routines have been provided for the output of 12-bit binary numbers. The process of outputting these numbers via XINTEGER requires time for conversion of its ASCII equivalent, which is far too time consuming if large numbers of A/D samples are being collected and output. In this case, the user can divide each sample into two 6-bit halves and output each half through the XOUT or XSTRING routines. These routines treat each word to be output as an 8-bit ASCII character. A special program has been written to read the data back and reformat it into OS/8 ASCII for further analysis by FORTRAN, BASIC, or FOCAL. This program accepts data files with any mixture of binary and ASCII output. Each 6-bit binary halfword is identified by the presence of 0s in bits 7 and 8 of each 8-bit character. One or both of these positions is always a 1 in OS/8 ASCII data.

Since the monitor leaves the interrupt on at all times, the user must be sure to only use handlers that run with the interrupt on. In general, disk, standard magnetic tape, Dectape or Linctape peripherals are satisfactory because they can be set by their OS/8 device handlers not to generate an interrupt when I/O is complete. Other devices like paper tape punches and line printers always generate interrupts and the interrupt handler would have to be modified to service them.

Since OS/8 device handlers generally issue a hardware transfer data instruction then loop until I/O is done, the user program essentially halts until the transfer is complete. This could cause delays of 2 msec to 500 msec (for disk or magnetic tape, respectively). Data from all interrupting facilities continues to be acquired during this time but the user program cannot act on this data until the output operation is over. This may be a consideration of importance if large quantities of data are to be processed continuously. In this case, an interrupting double-buffered handler would be needed. Such a handler for the magnetic tape TC58 controller (7 or 9 track drive) has been written and used with one version of the monitor for nearly a year.

User Callable Functions

The following sections describe monitor functions available to user program.

Timing and control routines. These routines read the real-time clock, measure latencies, request suspension of the program for specified periods, and return control to the monitor. XCANCEL: Terminates any wait period when called. This is usually called after a response is received from the experimental apparatus. XCLOCK: Initializes the real-time clock counter for timing events. XCLOK1: Reads the value of the real-time clock counter. Elapsed time since the last call to XCLOCK is returned to the user as a floating point number starting at the location specified in the accumulator. XEXIT: Provides a permanent exit from the user program to return control to the monitor keyboard controller. XMONITOR: Provides an exit to the monitor to wait for a response from the experimental apparatus. When a response occurs, control is passed to the user location specified by XENTER (see next section). XWAIT: Allows the programmer to have execution of his program stopped and then restarted after a specified time lapse. Control is returned at the location following the call of XWAIT. The programmer can decide to service responses from the experimental apparatus by setting the accumulator nonzero. A zero value will inhibit acknowledgment of these responses. After servicing a response during an XWAIT interval, control may be returned to the monitor via XMONITOR to wait for the next response or the end of the XWAIT interval.

Experimental interface input and output routines. These routines control input and output operations for the interface to the experimental apparatus. XENTER: Establishes an entry location in the user program to which the monitor will transfer upon receipt of a response from the experimental apparatus (i.e., a switch press). XROOM: Outputs the bit patterns following the calling address to the relay buffer for control of the experimental apparatus.

Data output routines. These routines provide for output of data collected during an experiment. Output may be to any OS/8 device for which there is an interrupt compatible handler. XCLOSE: Closes the output file and device. Makes the output file permanent. XFPOUT: Converts a floating point number at a user specified address to its ASCII representation and outputs it. The output field width is specified by the location following the call to XFPOUT. No decimal places are output in this version. XINTEGER: Converts a 12-bit octal integer to its ASCII representation and outputs it. The output field width is specified by the location following the call to XINTEGER. XOPEN: Calls the command decoder for input of device and file name of the output file, then opens it for output. XOUT: Outputs the single ASCII character in the accumulator. XSTRING: Outputs a string of alphanumeric ASCII characters stored one character per word. Output is terminated on a 0 word.

Console input/output routines. Routines to interact with the system console. XCONSOLE: allows the user program to print ASCII text on the console device. XREAD: Allows the user to input a string of alphanumeric characters from the console device. Up to 80 characters may be input.

Analog data collection routines. These routines allow the user to perform analog to digital conversion on up to 16 channels data. XSAMPLE: Starts the sampling of analog data for the number of channels specified by the argument sequence of the call. Other arguments in this call are the address of a ring buffer in the user program where sampled values are to be placed, the size of this buffer, the address of a pointer to this buffer, the number of channels to sample, and the sampling rate. XSAMP1: Terminates sampling of analog data.

Miscellaneous routines. These routines provide various functions useful for most experiments. XPERM: Randomly permutes a set of N numbers ($1 < N < 50$) starting at a location specified by the call. XRAND: Generates a random number on the range specified by the call ($0 < \text{RANGE} < 2047$). XSCOPE: Displays 1 or 2 channels of data on any standard oscilloscope via a AA005 D/A converter. This routine automatically scales each data value as if it were a digitized value from the AA01 A/D converter. XSCOP1: Displays 1 or 2 channels of data without scaling.

CONCLUSION

The authors will provide further details of the monitor or the hardware system to anyone interested. By simply substituting appropriate clock, relay, and analog-to-digital converter IOTs with appropriate parameters, the present monitor can run on many PDP-8I or PDP-12 configurations. Modifications of the clock and analog-to-digital service routines permit use of some of the sophisticated features of the PDP-8E and PDP-12 laboratory peripherals. We are willing to suggest modifications of the monitor to help adapt it to other PDP-8 configurations. Interested persons should contact

the first author with details of their computer configuration.

REFERENCES

Doll, T. J. A 4-K computer language for experimentation with human subjects, Behavior Research Methods and Instrumentation, 1972, 4, 27-31.

FIASCO, Computer-controlled psychology laboratory user manual, Department of Psychology, Carnegie-Mellon University, 1971.

Gips, J., Pfefferbaum, A., & Buchsbaum, M. ERL - A language for implementing evoked response and psychophysiological experiments, Behavior Research Methods an Instrumentation, 1971, 3, 199-201.

Grason-Stadler, SCAT user's manual, Grason-Stadler Company, Inc., West Concord, Massachusetts, 1970.

McLean, R. S. PSYCHOL: A computer language for experimentation, Behavior Research Methods and Instrumentation, 1969, 1, 323-328.

Millenson, J. R., Kehoe, E. J., Tait, R. W. & Gormezano, I. A minicomputer program for control and data acquisition in classical conditioning, Behavior Research Methods and Instrumentation, 1973, 5, 212-217.

Moise, S. L., Jr. & Jarrard, L. E. A computer-controlled system for training and testing primates. Behavior Research Methods and Instrumentation, 1969, 1, 234-236.

Rothman, Richard. 3 page floating point package with floating output. DECUS Program Library. DECUS No. 8-375B, 1970.

Uttal, W. R. Misuse, abuse, overuse, and unuse of on-line computer facilities by psychologists, Behavior Research Methods and Instrumentation, 1972, 4, 55-60.

Wallsten, T. 8TRAN language and the PDP-8 facility at the L. L. Thurstone Psychometric Laboratory, Behavior Research Methods and Instrumentation, 1972, 4, 107-108.

NOTE

1. OS/8 and OS/12 are Digital Equipment Company's operating systems which permit the use of a wide range of peripherals and all available core. It offers a versatile Keyboard Monitor that supervises a comprehensive library of system programs.

(Received for publication July 8, 1974,
revision accepted September 24, 1974.)