Behavior Research Methods & Instrumentation
1980, Vol. 12 (2), 120-125

SESSION V
LABORATORY SYSTEMS

STEVEN E. POLTROCK, *University of Denver, Presider*

Microprocessor control and A/D data acquisition in classical conditioning

JOHN SCANDRETT
Washington University, St. Louis, Missouri 63130

and

I. GORMEZANO
University of Iowa, Iowa City, Iowa 52242

An Apple II/FIRST system has been developed to control classical conditioning experiments, collect analog data, and to extract dependent variable measures of conditioning. With our selection of the Apple II microprocessor and an added hardware floating-point processor, we have been able to establish independent computer systems for each of our three conditioning laboratories at a fraction of the cost of our DEC PDP-8/e (which was interfaced to only one of our laboratories). Moreover, our software system, FIRST, an interactive, high-level, dictionary-based language, is a programming and control system whose flexibility and ease of programming far exceeds that experienced with our DEC PDP-8/e system (Millenson, Kehoe, Tait, & Gormezano, 1973; Tait & Gormezano, 1974). In our judgment, the Apple II/FIRST system is of unprecedented efficiency and versatility for the control, data acquisition, and data analysis of analog responses in classical conditioning experiments.

The application of computer technology to classical conditioning preparations from the Iowa laboratory has required: control software that will repeatedly generate discrete stimulus events, analog-to-digital (A/D) capabilities for recording analog data from uniphasic and multiphasic response systems, and the ability to extract a number of dependent variable measures from the digitized response topography. Briefly, classical conditioning experiments involve the manipulation of the intertrial interval and observation interval. The intertrial interval is generally long and variable in duration, and the stimulus conditions are relatively constant. On the other hand, the observation interval is generally shorter and fixed in duration, and nested within the interval is a pattern of stimulation involving an unconditioned stimulus (UCS), which reliably produces an unconditioned response (UCR), and a conditioned stimulus (CS) that has been shown by test not initially to produce a response resembling the UCR. The CS and UCS are then presented to the organism in a specified order and temporal spacing, and the experimenter examines responding during a specified observation interval. In CS-CR paradigms involving the direct measurement of conditioned responses (CRs) (see Gormezano & Kehoe, 1975), a response similar to the UCR develops to the CS that is called the CR, while CS-IR paradigms involve the indirect measurement of purported CRs through the effects of CSs on instrumental response (IR) baselines. Moreover, for CS-CR paradigms the definition of a CR is restricted to the selection of a target response from among those effector systems elicited as UCRs by the UCS. Generally, in CS-CR paradigms the experimenter examines analog signals from uniphasic or multiphasic response systems in which the extraction of dependent variable measures requires fine-grained measurement of response topography over a relatively brief period of time, as, for example, from those analog signals reproduced in Figures 1 and 2 from permanent (ink-written) oscillograph records.

The response depicted in Figure 1 is the rabbit's uniphasic nictitating membrane response (see Gormezano, 1966; Gormezano, Schneiderman, Deaux, & Fuentes, 1962), taken from an oscillograph record for a single conditioning trial. The upward and downward deflection

The hardware and software development presented in this report was supported, in part, by NSF Grant BNS 76-84561, NIMH Grant MH 16841, and NIDA Grant DA 01759. Requests for reprints should be addressed to I. Gormezano, Department of Psychology, the University of Iowa, Iowa City, Iowa 52242.

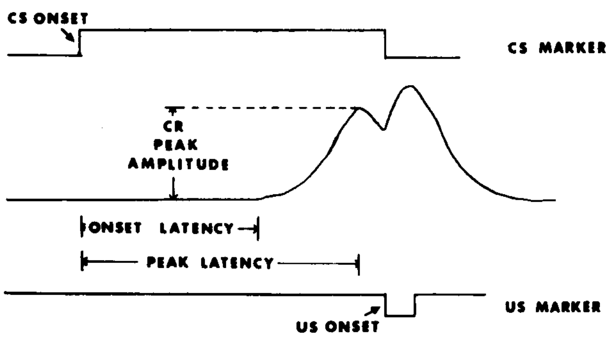

Figure 1. An oscillograph record of CS and US occurrence and the topography of a nictitating membrane CR taken at a paper speed of 200 mm/sec.

of the top line represents the duration of a tone CS, and the bottom line the duration of a shock UCS. Although the middle curve represents the analog record of the nictitating membrane response for only a single rabbit, we routinely obtain such analog records for six rabbits simultaneously on every trial. Similarly, Figure 2 depicts the rabbit's multiphasic (sinusoidal) jaw movement response (see Gormezano, 1972; Smith, DiLollo, & Gormezano, 1966) on a single conditioning trial for four rabbits simultaneously receiving an auditory CS paired with the intraoral delivery of a water UCS (through a fistula in each rabbit's cheek).

From the "raw" analog oscillograph data presented in Figures 1 and 2, the following information is abstracted on every conditioning trial: (1) the presence or absence of a CR, identified by an upward deflection in the analog signal of more than 1 mm from baseline (representing .5 mm of membrane extension) initiated in the interval between CS and UCS onset; and (2) the latency of CRs through the measurement of the distance between onset of the CS and the tangent point at which the upward deflection appears to begin and, subsequently, conversion of such ruler measurements to time (latency) by the appropriate transform based upon the paper speed of the oscillograph (e.g., 200 mm/sec). While changes in CR frequency and CR latency are routinely obtained and recorded on data sheets, the hand-scored measurement of other dependent variable measures extracted from the analog record becomes an exceedingly laborious, error-filled, and time-consuming task. However, our increasingly sophisticated understanding of classical conditioning has led us to be more and more concerned with being able to specify precisely the changes in CR topography that we observe over training and with the manipulation of a wide variety of independent variables. Specifically, among such additional dependent variable measures (see Figure 1) are: (1) CR peak latency, (2) CR peak amplitude, (3) latency of maximum amplitude, (4) area of the response (CR) in the CS-UCS interval (A + B), and (5) area of the response (CR + UCR) in the interval from CS onset to the end of the observation interval. Moreover, with regard to the jaw movement responses (portrayed in Figure 2), all of the preceding dependent variable measures apply, and, in addition, a determination is made of the number of sinusoidal jaw movements in the: (1) CS-UCS interval, and (2) interval from CS onset to the end of the observation interval.

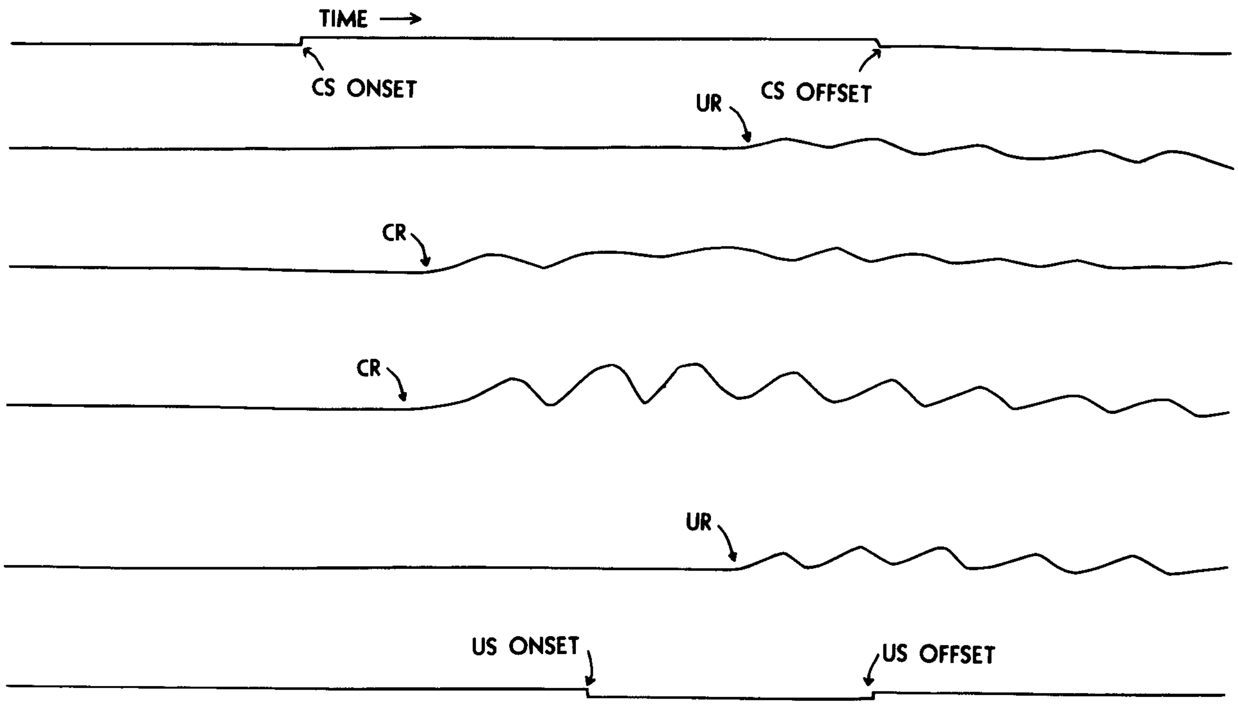

Figure 2. An oscillograph record of CS and US occurrence and the topography of jaw movement responses of four rabbits taken at a paper speed of 200 mm/sec.

Previously, the above dependent variable measures were laboriously determined by ruler measurements, transcribed by hand to data sheets, and subsequently keypunched on cards for statistical treatment by the University of Iowa Computer Center batch processor (e.g., Coleman & Gormezano, 1971). However, the volume of data involved and the need for more flexible and sophisticated manipulations of stimulus parameters led us to acquire a DEC PDP-8/e, and to develop a software system designed to collect a large amount of analog data (Millenson, Kehoe, Tait, & Gormezano, 1973; Tait & Gormezano, 1974). Subsequently, this minicomputer system was engaged in the control, collection of analog data, and extraction of primary dependent variable measures routinely used in the analysis of classical nictitating conditioning experiments from one of the three independent facilities comprising the Iowa Conditioning Laboratories (e.g., Millenson, Kehoe, & Gormezano, 1977). The ultimate hardware configuration of the DEC PDP-8/e system (which interfaced to six sound-attenuated experimental chambers for rabbit nictitating membrane conditioning) had a purchase and construction cost in excess of $50,000. Nevertheless, the system was limited in memory capacity and lacked the mass storage facilities needed for an operating system that would support convenient program development in a higher level language. Even after one substantial reprogramming effort (Tait & Gormezano, 1974) that expanded the capabilities of the control and analysis program, we were left with the formidable and time-consuming task of rewriting assembly language programs whenever we implemented new experimental protocols. Although some increased flexibility in our minicomputer system could have been achieved through additional (and relatively expensive) investments in hardware, we opted for the development of a microprocessor-based system (Kehoe, Frei, Tait, & Gormezano, 1975). We were drawn to this conclusion by our desire to implement computer technology to the other two conditioning laboratories constituting our facilities and by the restriction (substantiated by considerable experience) that each of the three laboratories remain completely independent of one another. Accordingly, cost considerations alone initially served to draw out attention to microprocessor-based computer systems as appearing to offer a relatively inexpensive means for applying computer technology to each of our laboratories. In brief, we are happy to report that with our selection of the Apple II microprocessor, we have been able to establish independent microprocessor-based computer systems for each of our three laboratories at a cost of under $3,000 each. Furthermore, we have established an Apple II microprocessor-based stand-alone system with a high-speed line printer for data processing and analysis for an additional cost of approximately $5,000. Above and beyond the considerations of the cost of a hardware system that would meet our initial needs, our selection of the Apple II microprocessor-based system was based upon the following criteria: ease and low cost of hardware expansion to implement new experimental protocols, availability of software support, ease of software programming, and the capability for real-time operation.

HARDWARE CONFIGURATION

The microprocessor-based computer system for each of the three conditioning laboratories consists of an Apple II with 48K of memory, an AM9511 floating-point processor, an Apple II disk drive, a Sanyo 9-in. video monitor, two 8255 programmable peripheral interface (PPI) chips, each providing a 24-bit digital input/output (I/O) buffer, and 8253 programmable counter/timer configurated to provide a real-time clock, and a 16-channel 8-bit A/D converter (ADC0816CCN) calibrated to yield, for each subject, 16 A/D counts for each millimeter of nictitating membrane extension (or jaw movement). The digital outputs to the stimulus devices in each experimental chamber are controlled by 5-V relays driven by dual positive NAND drivers (75462) connected to the PPI chips (see Grisham & Frei, 1977). In the present systems, connections from the digital outputs to the stimulus devices can be enabled or disabled by a bank of control switches.

The stand-alone microprocessor-based computer system for data processing and analysis consists of an Apple II with the same hardware configuration described above. In addition, however, the system contains a second Apple II disk drive, a Centronics 702 line printer, and an Apple II communications interface card to provide interactive capabilities with the University of Iowa Computing Center through a COMDATA (Model 302A2-13) acoustical telephone modem.

SYSTEM CAPABILITIES

Our software system, FIRST, is an adaptation of FORTH (Moore, 1974), an interactive, high-level language originally developed to make effective use of a small minicomputer with severely limited memory size (i.e., 8K words). FIRST is a programming and control system designed for simple, flexible, and effective experimental control, data acquisition, graphical data display, and data analysis. The basic system, resident in 13,800 bytes in an Apple II computer, contains a monitor, disk operating system, an efficient compiler, macro assembler, and text editor. The system also contains high-resolution graphical plotting capabilities, keyboard input and video output facilities for numerical and character string data, and line printer support for a Centronics 702 printer, as well as hard-copy graphics routines for printing graphical displays on an Axiom 820 microplotter.

FIRST provides computational capabilities for hard-

Table 1

Integer BASIC	FIRST
10 FOR J = 1 to 10000	INT X
20 X = 2*3*4*5	: TEST 1 10000 FOR 2 3 4 5 * * * X ! NEXT ;
30 NEXT J	Time: 4.6 sec
Time: 49.9 sec	
Applesoft BASIC	FIRST
10 Y = 3.1416	VAR Y VAR Z
20 FOR J = 1 to 10000	PI Y F !
30 Z = SQR(Y)	: TEST2 1 10000 FOR Y F@ SQRT Z F! NEXT ;
40 NEXT J	Time: 6.9 sec
Time: 492 sec	

ware add, subtract, multiply, and divide operations on 16-bit integers, 32-bit integers, and 32-bit floating-point numbers. The system also provides hard-wired floating-point evaluation of the usual transcendental functions: direct trigonometric, inverse trigonometric, log, exponent, and square root. It also performs hard-wired general floating-point exponentiation and data stack manipulations. FIRST is not simply an improved dialect of FORTH, adapted for any 6502 microprocessor. Rather, it is a similarly expandable dictionary-based language tailored for most efficient laboratory use of a 48K-byte Apple II system having, in addition to the usual memory-mapped text and graphic video display, an attached AM9511 floating-point processor chip with internal numerical stack. The system makes extensive use of firmware facilities present in the read-only memory (ROM) of the Apple, such as keyboard input, screen display, audible alarm, and disassembler.

Because FIRST is designed for flexibility and very high processing speed, it completely supercedes the functions usually provided by the Applesoft BASIC language processor. FIRST is typically 10 times faster than Apple integer BASIC for 16-bit arithmetic and 12-70 times faster than Applesoft floating-point BASIC. Some specific execution time comparisons are listed in Table 1.

FIRST LANGUAGE STRUCTURE

To the experimenter using FIRST, the system responds interactively to words entered on the display screen. The set of available words present in the system is stored in a dictionary resident in memory. The vocabulary of available words is like the functional keyboard of a calculator; each word, when entered and executed, performs a processing control or display function. For example, the word contained within the slash marks /CLS/ clears the display screen, /BELL/ rings the alarm bell, and /D/ displays all the words currently in the dictionary. When an explicit numerical value is entered, the implied operation is: push the value on the arithmetic push-down stack. The arithmetic operator words do not have general source or destination memory addresses associated with them as in conventional computer instructions. Instead, the arithmetic operator words are like the arithmetic operations in certain calculators of the reverse Polish variety: They always operate on whatever pair of numbers is on the top of the arithmetic stack (called the A-stack). Thus, an expression in BASIC or FORTRAN such as 2 + 3 would be carried out in FIRST as 2 3 + (push 2, then push 3, then add, leaving the result as new top of stack). The word /I./ pops off and prints the top of the A-stack.

So far, we have described a calculator that happens to have 600 "function keys." The programming flexibility of FIRST arises from one's capability to invent new words as combinations of old words. The basic compile operation creates a new word as a specified execution sequence of already existing words and adds the new word to the dictionary. /:/ means start compile, and /;/ means end compile. Thus /: NEWWORD CLS BELL;/ expands the dictionary to include a new entry /NEWWORD/, which, when entered, clears the screen and rings the bell. We could read the defining sequence above as: "create/NEWWORD/, which means /CLS/, then /BELL/, then return." Thus, /NEWWORD/ is a newly compiled subroutine. After a new word has been added to the dictionary, it can, in turn, become a constituent part of a still newer word being defined. In fact, the only possible formal programming error in the FIRST system is to refer to a word that does not yet exist (the error-message response to /QWERT/ is QWERT?).

A sizable, complex programming task such as stimulus control and A/D data logging of nictitating membrane responses is carried out in a hierarchical manner by starting with the existing vocabulary as a programmer's "kit of parts" used to build words tailored to carry out the individual control actions (tone on, tone off, shock on, shock off, start A/D sampling, display data, wait for intertrial interval, etc.). Using these words, still more comprehensive words are written that describe a conditioning trial, and finally, one last word is created, /RUN/, which specifies the repetitions of trials for one complete experimental session. The total number of added words in the pyramid culminating in /RUN/ is typically 180.

The word definitions comprising a particular program (control and acquisition, nictitating membrane response latency analysis, CR percentage, etc.) can be thought of as a specialized vocabulary relevant to a particular problem area. There is no formal distinction between the operating system, the compiler, and the experimenter's program. In effect, one creates for each program the most suitable new language with which to carry out the control and computational elements that make up the program. The word definitions comprising a program are written on disk-storage screens using a text editor (/45 EDIT/ means "read in Screen 45 from disk to display screen and then edit"). The compiler is so fast that one ordinarily does not bother to save compiled object code on disk. The source language typically occupies 25 screens (960 bytes each) on disk and requires 14 sec to compile (the nucleus of 600 FIRST words bootstrap-loads from disk in 8 sec).

CONTROL HARDWARE AND SOFTWARE

Stimulus control is carried out by storing or writing output control bits into PPIs. In general, an interface built for an Apple computer has a small range of internal device-port addresses at a base address determined by the connector slot into which the interface is inserted. For example, the PPI port whose output bits control tone and light stimuli is inserted in Slot 5 and has $COD8 as its hexadecimal address. The words /4 $COD8 B!/ store zeros in Bits 0, 1, 3, 4, 5, 6, 7, and a 1 in Bit 2, which gates on the tone generator (the syntax is: "push value, push address, perform a byte-store operation"). Here, then, are some useful intermediate word definitions: /: TON 4 $COD8 B! ; / and /: TOF 0 $COD8 B! ;/. We test these new words by executing them directly from the keyboard to verify that /TON/ makes the tone come on and that /TOF/ turns if off. Once we have carefully checked the output bit that performs a specific function, we are no longer concerned with the "bit level" of software and interface detail, and we only need to remember the function associated with the word. Thus /TON/, /TOF/, /LON/, /LOF/, /SHN/, /SHF/ are words that control the onsets and offsets of tone, light, and shock stimuli.

The next programming level in a stimulus control program involves the timing of stimuli and the multiplexed A/D conversion of nictitating membrane response signals. The timekeeping function is implemented by a programmable interval timer (PIT) that contains three independent 16-bit pulse counters, of which we use two. The first PIT timer has the Apple 1.02-MHz pulse train as the input and operates in a "repeating-divide-by-N" mode to produce a square-wave output whose state can be examined by reading Bit 2 of a PPI whose address is $C112. The output of Counter 1, whose pulse period is typically 4 msec during a trial interval and 10 msec during the intertrial interval, feeds the input of Counter 2, which then counts "ticks of the clock." The intertrial interval, typically 60 ± 10 sec as generated by our random number generator, is timed by looping and reading the contents of Counter 2 until the desired count has elapsed. The timing of stimuli and A/D samplings requires the use of direct 6502 assembler language because of critical speed requirements. For each tick of the sampling clock (2-5 msec), a sequence of words must be executed that measures six or eight A/D channels and compares the elapsed time with a sequence of control-function times, programmed with logical-IF constructions. Portions of assembly language can be freely interspersed within a new word definition. Macrodefinitions, which generate convenient in-line groups of assembler instructions, are also available.

RESULTS

The three University of Iowa conditioning laboratories are now controlled by individual Apple-FIRST control computers, and the analog response data for all experiments are routinely logged on floppy disks for later signal processing on a stand-alone Apple-FIRST system having a line printer and hard-copy graphic facilities. Most experiments are run under a generalized classical conditioning program that has provisions for the presentation of CSs and UCSs, with onset and offset times specified as variables. The program records all sampled A/D values for all subjects and displays the digitized nictitating membrane response for each subject, successively, during the intertrial interval of the experiment, along with nictitating membrane response topography measures extracted by the analysis program. Next, the salient response measures are stored in a memory buffer until filled, at which point the data are moved to disk files reserved for data storage. Currently, the signal analysis program extracts the following measures from the digitized nictitating membrane response topographies: (1) a RMS estimate of the quality of the baseline signal prior to CS onset, (2) CR and UCR latency, (3) CR and UCR amplitude, (4) peak CR and UCR latency, and (5) total area under the CR and UCR topographies.

The panels of Figure 3 depict the video display for one subject's nictitating membrane response topography during the course of training. The vertical line at the bottom of each panel specifies the program's determination of CR or UCR latency, and the second horizontal line from the bottom of each panel defines the response baseline. While the nictitating membrane response was digitized by a 4-msec A/D sampling rate, the topographies displayed on the video monitor are the result of plotting every other data point (i.e., every 8 msec). Examination of the panels in Figure 3 reveals the signal analysis program's determination of response latency. The program first determines if the peak amplitude of the nictitating membrane response is greater than the (.5-mm) criterion. If the amplitude of the nictitating membrane response fails to attain criterion, a −1 is stored and a beeper alarm is sounded to appraise the experimenter of the subject's failure to make a CR or UCR on that trial. Second, the program averages 200 msec of sampled baseline preceding the CS and

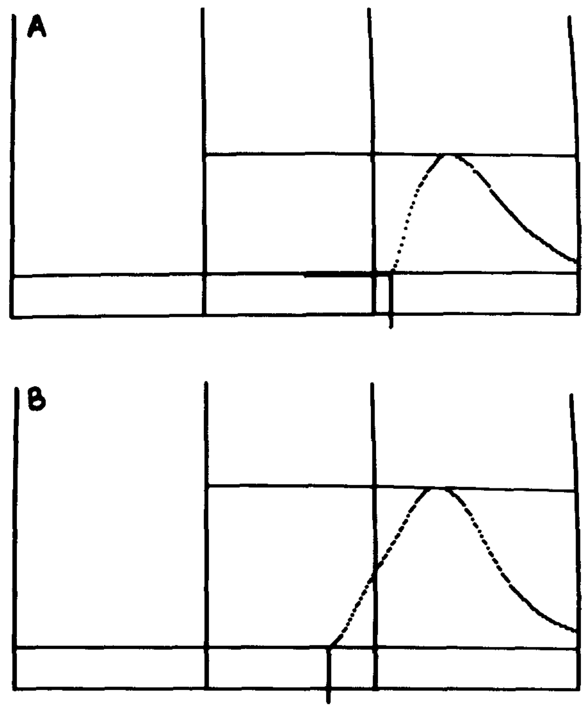

Figure 3. Panel A reveals the digitized topography of the nictitating membrane UCR and Panel B that of the CR, as it is displayed on the video monitor for a single subject. In each panel the vertical lines to the extreme left and right mark the observation interval, while the second vertical line from the left indicates CS onset and the second vertical line from the right denotes UCS onset. The second horizontal line from the bottom specifies the baseline (which is rescaled for each subject by a subroutine), while the small vertical line crossing the baseline indicates the program's determination of response latency.

calculates the RMS, which is used to normalize the graphical display of the nictitating membrane response and also serves as a measure of baseline quality for the latency-finding algorithm. Thus, if some remnant of a baseline nictitating membrane response is present during the 200-msec baseline sample, the RMS will be many times greater than the typical value of .2 counts (16 counts corresponds to 1-mm membrane deflection). Accordingly, if the RMS is greater than a criterion level, indicating that the rabbit was responding prior to CS onset, a −3 is stored and, again, a beeper alarm is activated to alert the investigator. The detection of nictitating membrane response latency (CR or UCR) is determined by searching the digitized response to the right, from CS and/or UCS onset, until a displacement from baseline exceeds .5 mm, then sampling to the left until the displacement (found by successive averaging of a specific data point with the values of the neighboring point on either side) is reduced to 1/16 mm, which is approximately 5 standard deviations above the baseline noise level. Algorithms also exist for extracting the other dependent variable measures.

In practice, subsequent to an experimental run, the digitized nictitating membrane responses are copied from the disk file onto a sequential library disk, which holds all the data for a sequence of runs. This library disk then serves as the input to programs that classify each individual subject's response as a UCR or CR, and they (rarely) exclude trials because of unacceptable baseline quality. These CR percentages are then suitably formatted for transmission via the telephone modem to a timesharing system on which ANOVAs and other statistical package programs are run. Other final analysis programs plot and display histograms of time distributions of latencies.

In conclusion, we now have affordable, independent control and data acquisition systems on which each experimenter can write and test his own program for each new set of conditioning parameters under investigation. The ease and flexibility of writing analysis programs in FIRST also allows each experimenter to write the specialized counting, classifying, and plotting programs with which to acquire and display final results. In our judgment, these systems are of unprecedented efficiency and versatility for the control, data acquisition, and data analysis of classical conditioning experiments.

REFERENCES

Coleman, S. R., & Gormezano, I. Classical conditioning of the rabbit's *(Oryctolagus Cuniculus)* nictitating membrane response under symmetrical CS-UCS interval shifts. *Journal of Comparative and Physiological Psychology, 1971,* **77,** 447-455.

Gormezano, I. Classical conditioning. In J. B. Sidowski (Ed.), *Experimental methods and instrumentation in psychology.* New York: McGraw-Hill, 1966.

Gormezano, I. Investigations of defense and reward conditioning in the rabbit. In A. H. Black & W. F. Prokasy (Eds.), *Classical conditioning II: Current theory and research.* New York: Appleton-Century-Croft, 1972.

Gormezano, I., & Kehoe, E. J. Classical conditioning: Some methodological-conceptual issues. In W. K. Estes (Ed.), *Handbook of learning and cognitive processes* (Vol. 2). Hillsdale, N.J: Erlbaum, 1975.

Gormezano, I., Schneiderman, N., Deaux, E., & Fuentes, I. Nictitating membrane: Classical conditioning and extinction in the albino rabbit. *Science,* 1962, **138,** 33-34.

Grisham, M. G., & Frei, L. J. An optically isolated digital interface for the SKED system. *Behavior Research Methods & Instrumentation,* 1977, **9,** 215-218.

Kehoe, E. J., Frei, L. J., Tait, R. W., & Gormezano, I. On microprocessor-based computers. *Behavior Research Methods & Instrumentation,* 1975, **1,** 183-186.

Millenson, J. R., Kehoe, E. J., & Gormezano, I. Classical conditioning of the rabbit's nictitating membrane response under fixed and mixed CS-US intervals. *Learning and Motivation,* 1977, **8,** 351-366.

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 & Instrumentation,* 1973, **5,** 212-215.

Moore, C. H. FORTH: A new way to program a minicomputer. *Journal of Astronomy and Astrophysics Supplement,* 1974, **15,** 497-511.

Smith, M. C., DiLollo, V., & Gormezano, I. Conditioned jaw movement in the rabbit. *Journal of Comparative & Physiological Psychology,* 1966, **62,** 479-483.

Tait, R. W., & Gormezano, I. A minicomputer program for stimulus control and analog data for discrete trial paradigms in biological preparations: Classical conditioning. *Behavior Research Methods & Instrumentation,* 1974, **6,** 295-300.