A general-purpose computer system for behavioral conditioning and neural recording experiments

GENGXIN CHEN and JOSEPH E. STEINMETZ
Indiana University, Bloomington, Indiana

A computer system designed to control the delivery of stimuli and data acquisition during behavioral conditioning and neural recording experiments is described. This IBM-PC-compatible system is programmed in C++ and is capable of controlling stimulus presentations to four independently operating conditioning chambers while collecting one channel of analog and two channels of digital data from each of the chambers. This system is currently being used in a variety of learning and memory experiments involving rats, rabbits, and humans.

As personal computers have become more popular, affordable, and easy to operate, they have become ideal tools for maintaining experimental control and collecting data in the laboratory. Over the last several years, a number of systems have been developed that use a variety of strategies for delivering stimuli and collecting data during classical and instrumental conditioning experiments (e.g., Lavond & Steinmetz, 1989; Scandrett & Gormezano, 1980). For example, Palya and colleagues have described a powerful, yet inexpensive, experimental controller that can be programmed in either a BASIC-like or an ALGOL-like procedure specification language. This system was designed to function as a remote peripheral processor in a network with any computer acting as the network supervisor and each serial port on the host computer supporting several simultaneous experiments (Palya & Walter, 1993; Palya, Walter, & Chu, 1995). Other examples of systems that have been recently designed to control the delivery of stimuli or the acquisition of data during simple conditioning experiments include (1) an interface described by Markham (1993) for controlling external devices via the parallel port of IBM-PC-compatible computers; (2) a system designed by Penttonen, Salmi, Hamalainen, and Meriluoto (1994) for controlling classical conditioning experiments; (3) a WorkbenchMac- or PC-based system designed by McGregor (1996) for use in animal learning, psychopharmacology, and behavioral neuroscience experiments; and (4) an IBM-PC-compatible system for use in classical eyeblink conditioning experiments (Thompson, Moyer, Akase, & Disterhoft, 1994).

These systems have certainly proved useful and reliable, but they have some weaknesses with respect to initial setup, ongoing maintenance, and operation. For example, the Apple II system developed by Scandrett and Gormezano (1980) uses a unique control language (First) and requires that the user build hardware interface boards for the computers. The PC-based system designed by Lavond and Steinmetz (1989) uses Forth, a more universal programming language, but also requires that the user manufacture an interface board. The 386/486 system developed by Penttonen et al. (1994) uses commercially available hardware but is programmed with custom-designed, special-purpose software. In addition, although this system can be used for computer control of stimuli, a second computer is needed for data acquisition.

In part because of the limitations of these existing systems, we designed a new computer system that would be relatively easy and inexpensive to set up and somewhat easy to program and maintain. Specifically, we desired a system that could (1) use commercially available hardware and a popular programming language; (2) generate stimuli used in a variety of behavioral tasks such as classical, instrumental, and operant conditioning experiments; (3) record analog and/or digital data related to behavioral responding; (4) record analog and/or digital data representing brain activity; (5) use a single computer to collect data from multiple experimental chambers, with each chamber capable of being programmed independently from the other chambers; and (6) be used by researchers with minimal computer experience.

The system described here was designed for use on IBM-PC-compatibles and was originally used for classical eyeblink conditioning experiments involving rabbits and humans. With minor modifications, the same system was adapted for use in rat instrumental conditioning studies that are ongoing in our laboratory (see Applications section below).

Development of this system was funded in part by NIMH Grant MH51178 and funds from the Indiana University College of Arts and Sciences. We thank Donald Katz for comments concerning this manuscript. Correspondence should be addressed to J. E. Steinmetz, Department of Psychology, Indiana University, Bloomington, IN 47405-1301 (steinmet@ucs.indiana.edu).

Figure 1 shows a graphical user interface titled "MAIN". It contains four sections, one for each chamber. Each section has labels for "animal:", "paradigm:", "switch:", and "data file:". Chamber 1 and 2 have their "switch" fields set to "[ON]", while Chamber 3 and 4 are set to "[OFF]". The "animal" and "paradigm" fields contain text describing the experimental setup, and the "data file" fields contain file paths. To the right of each chamber section are several buttons or checkboxes, some of which are highlighted.

Figure 1. Example of a Main Dialogue box screen in the user interface. Here, the user enters information concerning the experimental procedures to be run, including the chambers that are active, a description of the session, the paradigm to be selected, and the name of the file into which analog and digital data are to be stored. In addition, other user options can be selected from this screen (see text for details).

HARDWARE AND SOFTWARE SELECTION

This system has been designed to run on a 386 or 486 IBM-PC-compatible computer with a VGA card and 640×480 resolution color monitor. The software runs on the DOS operating system. The interface board linking the computer and the peripheral hardware used for stimulus delivery and data acquisition is the DAS-1602 (Keithley MetraByte, Inc., Tauton, MA). This board is installed in one of the standard PC expansion slots. We chose this interface because it had a number of features that were necessary for our design, including (1) 12-bit resolution analog-to-digital (A/D) conversion with a sampling rate of up to 100 Ksamples/sec; (2) as many as 16 single-ended or 8 differential switch-configurable A/D channels; (3) data transfers that can be performed by program control, interrupts, or direct memory access (DMA); (4) 32 bits of available digital input/output (I/O) (4 bits designated as input, 4 bits designated as output, and 24 bits that can be programmed as either input or output); and (5) a Borland C++ compatible function library available for interface board programming and operation that allowed the system to be implemented without the need for an abundance of low-level code. With the DAS-1602, our system is capable of controlling four independent chambers in real time. The computer can send 6 digital outputs to each chamber (e.g., to control stimuli presentations) while receiving one channel of analog data and two channels of

digital data simultaneously. If fewer chambers are used, the number of inputs and outputs that are available can be increased. For example, if only one chamber is involved in training, up to 24 digital outputs, 4 analog inputs, and 8 digital inputs are available for programming.

The software for this system is written in Turbo C++ (Version 3.0), an object-oriented programming environment. We chose to use C++ in the development of this system in part because it is a widely used programming language and in part because the interface board came equipped with a large library of C++ compatible functions for low-level routines.

THE USER INTERFACE

A major consideration in designing this system was the user interface—that part of the system with which the user interacts during its operation. Because it was our goal to design a system that was very easy to operate and that did not require extensive computer experience or expertise to use, we incorporated a user-oriented interface into the design of the system. This user interface is designed in a style similar to that of the dialogue boxes of Borland products, with each dialogue box containing up to five types of items: text, button, label, switch, or input line. All but text items can be selected (highlighted) by user input. Text is used for description and to make the dialogue boxes more informative (e.g., putting units after numbers

Configure: Chamber 2

System Gain:	650	mV/mm
Sampling Interval:	3	msec
Bin Size:	3	msec (must be multiple of sampling interval)
CR Criterion:	0.5	mm
Bad Trial Criterion:	0.5	mm
Training Criterion:	8	CR's in a row of 9 trials
Display Units:		
Behavior:	8	pixel/mm K
Unit Recording:	1	pixel/count
Sum of Unit Recording:	1	pixel/count []

Figure 2. A variety of parameters are set for Chamber 2 from this Configuration screen, including sampling rate, behavioral criteria, training criteria, and display settings.

that are entered). Buttons are used to run commands. Once a button is selected, a <RETURN> keypress results in the execution of command associated with that button. Labels are always associated with switches or input lines to identify the names of the associated items, and a label and its associated item are always selected at the same time. A switch has only two states (ON and OFF) and can be toggled with the <SPACE BAR> when selected. When selected, input lines allow users to input and edit names, comments, or numbers. Items are selected by using the <TAB> key to move through the items one by one or by using the <ALT> plus one of several "hot" keys.

When the program is started, a Main Dialogue box appears (Figure 1). At this point, the user can select any combination of the four chambers for operation, enter comments regarding the subjects to be run, select the paradigm to be run, and enter the file name under which the data are to be stored. A set of buttons appears on the right side of the display, including (1) Configure, for setting up the configuration for each chamber; (2) Paradigm, for selecting paradigm parameters; (3) Load/Save, for allowing users to load or save configuration files to automatically configure the system; (4) Run, for beginning a session; and (5) Exit, for terminating the setup process.

When the Configure button is selected, the user can choose a chamber to program. Selection of a chamber for use then causes a Configure Dialogue box to appear (Figure 2). This box contains system parameters, training parameters, and display parameters. The system parameters include variables such as A/D gain, length of the A/D

sampling interval, and bin size for sorting digital data. These parameters are used for calculations involved in scaling recorded responses (e.g., converting voltage of a dc signal from an integrated electromyographic record to millimeters of eyelid movement). The training parameters include amplitude criteria for defining response detection and excessive pretrial activity in the A/D input. The display parameters are a series of values that control the relative size of the A/D and digital data displayed in real time during the run. After the parameters in the Configure Dialogue box have been set, the user chooses the OK button or discards the changes by operating the Cancel button.

Setting up a paradigm involves four dialogue boxes that appear serially. The Paradigm Dialogue boxes allow the user to enter a variety of information concerning the paradigm to be run. As currently programmed, our system allows several classes of paradigms to be set up and run, including classical, signaled appetitive and aversive instrumental conditioning, and operant conditioning. After selecting a paradigm to be run, altered, or defined, information concerning the paradigm is entered in the Paradigm Configuration boxes (Figures 3, 4, and 5). Intertrial interval timing (e.g., minimum and maximum values) is set in one of these boxes (Figure 3), as are definitions concerning events that occur during the intertrial interval. Also, timing parameters (i.e., onsets and offsets) for stimuli presented on individual trials are defined in this dialogue box. Two other dialogue boxes are used to provide users with a flexible and simple method for arranging stimuli delivered throughout the session. First, the user

Edit Paradigm

Paradigm Name: **cortstim**

Comment: **H-VI stim in way of conditioning (extinction phase)**

Conditioning Type: **0** (0-2) Context delay: **0** sec

ITI Minimum: **20** sec Shift Range: **10** sec **K**

Blocks: **14** Trials/Block: **11**

Length of Trial: **750** msec

	Name	Onset(msec)	Offset(msec)	Compensat(msec)
CS 1	tone	250	600	
CS 2		750	1500	
CS 3	stimulator	300	350	
SYNC		2	12	
US 1	air puff	500	600	20
US 2		1250	1500	0
Pre-CS Period:		0	249	
Bad Trial Detection Window:		150	294	

Figure 3. The Edit Paradigm screen, an example of which is shown for the cortstim paradigm, allows the user to enter identification information, intertrial interval values, trial blocking schemes, and timing parameters for the trials to be delivered.

is allowed to select the stimuli to be used for each trial type (Figure 4). For example, during classical conditioning experiments, users may wish to deliver two trial types, one with paired presentation of the conditioned stimulus and the unconditioned stimulus and a second with conditioned stimulus-alone presentations. The user can define any combination of the stimuli that are available. After trial types have been defined, another dialogue box allows the user to order the trial types in any manner desired (Figure 5). Typically, the user can begin the run after these dialogue boxes have been completed.

PROGRAM STRUCTURE

Data Structures

Parameter values and other information needed to control the delivery of stimuli and the acquisition of data are stored as lists in the C++ environment. A collection of these lists known as PARADIGM contains all the information and parameters needed to define the training paradigm, including paradigm descriptors (name, comments, paradigm type); intertrial interval parameters; information concerning the number of trials and how the trials are to be blocked; trial timing information such as trial length, interstimulus interval length, and the onset and offset of all stimuli to be presented; an array of stimulus types; and a table defining the order and arrangement of trials within the session.

Another collection of lists, EXP_BOX, contains three general categories of information: channel addresses, system configuration, and run-time variables. Channel addresses for analog input as well as port and line addresses for all of the digital input and output are stored in EXP_BOX. This information is stored in a file on the hard disk named "iomap.sys" and is loaded when the program is started. System configuration information also appears as part of EXP_BOX (i.e., information that was entered by the user via the user interface described above). Included in the system configuration is chamber number and activity switch, names of the paradigms to be run in each chamber, a variety of information concerning data sampling (e.g., A/D gain and sampling rate and digital sampling bin size), criterion for detecting excessive A/D input prior to each trial, and display parameters. Like the PARADIGM list, this information is stored to the hard disk every time a session is initiated as the file "system.cfg". Users can also save this information in configuration files that can be loaded into the system prior to running. Run-time variables stored in EXP_BOX include all of those variables needed to control the running of a session in each of the chambers, such as an identification of the paradigm that has been selected to be run, a text string describing the session to be run, the name of the file in which raw data are to be stored, counters for the intertrial interval, blocks and trials, and calculations for data that are displayed after each trial (e.g., response amplitudes,

Figure 4. The Set Stimulus Type screen allows the user to define as many as 32 trial types (i.e., combinations of stimuli defined on the Edit Paradigm screen).

onset latency, etc.). The operations of the collection of parameters stored in EXP_BOX are actually the core of the program and take care of all of the basic chamber operations (e.g., countdown intertrial interval timer, trial trigger, and on-line displays).

Signal Control and Data Collection

After the program is started and the user executes the RUN command, a software switch to each chamber is checked by the program to determine which of the chambers are to be used during the session. If the software switch is off, the chamber is removed from the "running list" (i.e., those chambers to be serviced during the session). Another empty list, the "finished list," is also created when the program is started. During the run, if a chamber finishes a session, it is removed from the running list and placed on the finished list. The program keeps looping until all of the chambers set up in the running list have finished and been moved to the finished list, so that the running list becomes empty. At this point, the program follows the finished list to print out summary reports and initiate routines to archive data files for each chamber.

The intertrial interval is controlled by a time counter set up individually for each chamber. The counter is initialized with a pseudo-random number within a range designated by the user for each paradigm. The program reads a clock continuously in the looping process. If the clock ticks (1 sec more than the previous reading), the counter of each chamber is updated (decremented). If the counter reaches zero, a trial is triggered in the chamber. For the

intertrial interval countdown routines, the operation of the clock is a normal C++ library function. Values for other functions, such as trial triggering, counter manipulations, and display, are stored among the group of parameters that constitute EXP_BOX.

Trial Control and Data Collection

Upon the triggering of a trial in a chamber, the program first resets some of the parameters on the DAS-1602 interface board using configuration information for the chamber entered by the user before the run. The parameters include sampling rate, A/D channel selection, and number of sampled points. Because the interface board is reset every time a trial is triggered, each chamber can be operated independently from the other chambers. Different paradigms with different stimuli selection and timing configurations can be operated in each of the chambers if the user wishes.

In our application, two sources of data are typically collected; analog behavioral data that are entering the computer via an A/D channel, and digital data, such as TTL pulses, that are input via I/O ports. The analog data normally represent the position of an eyelid during conditioning, whereas the digital data represent concomitantly recorded discriminated action potentials from an area of the brain. Because of interface hardware limitations, we transfer the A/D data with DMA mode while reading the I/O ports with software polling. The polling code is written in a manner that makes the polling as quick as possible. This enables us to read I/O input pulses as narrow as 100–

	Trial										
	1	2	3	4	5	6	7	8	9	10	11
Block1:	3	1	1	1	1	2	1	1	1	4	1
Block2:	3	1	1	1	1	2	1	1	1	4	1
Block3:	3	1	1	1	1	2	1	1	1	4	1
Block4:	3	1	1	1	1	2	1	1	1	4	1
Block5:	3	1	1	1	1	2	1	1	1	4	1
Block6:	3	1	1	1	1	2	1	1	1	4	1
Block7:	3	1	1	1	1	2	1	1	1	4	1
Block8:	3	1	1	1	1	2	1	1	1	4	1
Block9:	3	1	1	1	1	2	1	1	1	4	1
Block10:	3	1	1	1	1	2	1	1	1	4	1
Block11:	3	1	1	1	1	2	1	1	1	4	1
Block12:	3	1	1	1	1	2	1	1	1	4	1
Block13:	3	1	1	1	1	2	1	1	1	4	1
Block14:	3	1	1	1	1	2	1	1	1	4	1

Figure 5. The Set Stimulus Table screen allows the user to define trial type order for blocks of trials defined in the previous two screens.

200 μ sec (a common output for commercial window discriminators such as those manufactured by F. Haer, Inc.). There are six data buffers for each chamber, one for A/D data from the current trial, one for averaged A/D data for a block of training, two for digital data from the current trials (one for each channel), and two for summed unit activity across a block of training trials (one for each channel).

Keithley Metrabyte, Inc., provides a library of functions to monitor the operation of the DMA transfer. Because the data transfer is controlled precisely by a timer on the interface board, the counter can be used as a time index for stimuli delivery signals and also to determine bin onsets and offsets for digital data sampling. Before the DMA data transfer begins, onsets and offsets of stimuli switched on for a given trial in a given chamber are calculated and scaled according to the units defined by sampling interval parameters selected before the run. The counter for transfer is continuously compared with the time for a candidate stimulus operation (onset or offset). Once they are equal, that operation is executed and the next candidate is selected. At the same time, the border of the digital data bin is compared with the transfer counter. Once the counter crosses the border of the bin, a new bin is set as well as a new bin border. In this way, digital data are collected in bins that are established across time (i.e., within a trial). Digital pulses are counted by detecting the rising edge of the pulse—that is, once a high voltage (logic 1) is read following a low voltage (logic 0), the program counts 1 in the current bin. With this scheme, it is important that the polling loop time be less than the digital pulse width, so that no pulses are missed.

Data File Format

During each session, the program stores the raw A/D and digital data for each chamber in a data file set up by the user prior to the run. The data file consists of two parts, a header containing all of the information concerning the chamber configuration and paradigm parameters for that session and a data part that is organized in a trial-by-trial format. For each trial, a block of A/D data is followed by two blocks of digital data (a block for each digital data channel). Since the A/D resolution on the DAS-1602 interface board is 12 bit, each A/D data point is 2 bytes, with low byte first and high byte second (e.g., IBM-PC short integer format). Each digital data point is 1 byte. The size reserved for the header is 1,024 bytes. The size of the data part is determined by the sizes of the blocks of A/D data and digital data stored with the respective block sizes determined by the total number of trials run, the trial length, and the sampling rate.

DISCUSSION

Applications

The system described here has been installed in several computers in our laboratory and we have been using it for the past 2 years. Our major application of the system has been in classical eyelid conditioning experiments involving rabbits (e.g., Steinmetz, 1996). In these experiments, a conditioned stimulus (CS) such as a tone or light is presented 250–750 msec before an unconditioned stimulus (US), which is typically a blink-eliciting air puff directed at the cornea of the eye. With paired CS and US pre-

Figure 6. A picture of a computer display taken during a rabbit classical eyeblink conditioning experiment. In this example, eyelid position (top horizontal trace) and discriminated action potentials from one brain site (middle horizontal trace) were being monitored from 2 rabbits. The first vertical line for each box depicts trial onset, the second vertical line depicts CS onset, and the third vertical line depicts US onset (interval between CS and US onsets = 250 msec). The text at the bottom of Boxes 1 and 2 provides information concerning the behavioral response executed during the last trial as well as information concerning the upcoming trial. In Box 1, on the last trial a maximum of 0.03 mm of movement was detected between trial onset and CS onset, a maximum of 0.06 mm of movement was detected between CS onset and US onset, and a maximum of 10.9 mm of eyelid movement was recorded after US onset. A 282-msec latency was recorded between CS onset and a criterion of 0.5 mm of eyelid movement. As indicated by the behavioral trace and these data, this rabbit has executed a reflexive response to the air puff but no conditioned response (i.e., the eyeblink response was executed after US onset). The bottom line of text indicates that 5 sec remain before the fourth trial of the first block of training (Trial 4) is delivered. The remaining number shows the current position of the eyelid. In Box 2, 0.02 mm of movement was detected in the pre-CS period, a peak response of 8.35 mm of movement was detected during the CS-US interval, and 10.3 mm of movement was detected after US onset. An onset latency of 162 msec was recorded (i.e., the rabbit in Box 2 executed a conditioned response on the previous trial).

sentations, the rabbits eventually learn that the CS predicts the US and begin blinking to the CS, prior to US presentation. Eyeblinks are normally monitored through electromyographic (EMG) recordings or through noninvasive infrared devices. In both cases, dc signals are input into the computer system via an A/D channel on the DAS-1602. We can monitor the behavior of 4 subjects in four different chambers, each chamber running an independent experimental procedure. With the present system, we have been able to vary the basic eyeblink conditioning paradigm to investigate a variety of classical conditioning phenomena, including discrimination conditioning using two different CSs, interstimulus interval discrimination conditioning using two different interstimulus intervals and one CS, trace conditioning, and, more recently, contextual conditioning that requires tonic changes in stimuli during the intertrial interval. We are also currently using this system in human classical eyeblink conditioning experiments.

Researchers in our laboratory often monitor rabbit brain activity during the conditioning process. In these experiments, electrodes are either implanted chronically or lowered acutely into a region of the brain, and action potentials are recorded during acquisition and performance of the classically conditioned response. Typically, the amplified brain signal is routed to a window discriminator that detects action potentials that exceed a preset level. The present system is capable of monitoring two

channels of discriminated action potentials in real time per chamber. The TTL outputs of the window discriminator are routed to a digital input channel on the DAS-1602, which facilitates counting discriminated unit activity across the duration of each trial. Figure 6 shows a reproduction of a computer monitor display depicting behavioral responding and discriminated unit activity recorded simultaneously from 2 rabbits.

We have also recently installed the Spike2 waveform analysis system (CED, Ltd.) on a second computer to which we route raw unit activity from the brain amplifiers. Sync pulses from our conditioning system control the operation of the Spike2 system. The Spike2 system performs separation of action potentials on a number of brain signals using waveform-fitting routines. These analysis procedures are performed off line.

We also use this system to generate stimuli and record data in a variety of ongoing rat instrumental conditioning experiments (e.g., Steinmetz, Logue, & Miller, 1993). In these experiments, a tone is presented for a period of time and the rat is required to press a response lever to either receive a food pellet reward (an appetitive task) or avoid a mild foot shock (an aversive task). In these instrumental procedures, unlike those in classical conditioning, the behavior of the rat affects the delivery of stimuli since the leverpress produces a food reward or the avoidance of a shock. These paradigms also require operant-based shaping procedures that are under the control of the computer,

and we have successfully used the system in research that required the generation of a variety of operant schedules, including fixed interval and fixed ratio delivery of food reinforcements. The present system controls the delivery of all stimuli and records the leverpressing behavior of the rats on each trial. As in the classical conditioning setup, we can monitor the behaviors of 4 rats being conditioned under four different procedures and can concomitantly monitor two channels of discriminated brain activity from each chamber in real time.

In recent experiments, we have also used this system in autonomic (fear) conditioning experiments, differential rates of low responding conditioning, and passive avoidance (step-down) experiments.

System Requirements

This conditioning system requires an IBM-PC-compatible 386 or 486 computer with a hard disk drive, 4 MB of RAM, and a VGA display. The only limitation that we have encountered to date is a compatibility problem with some computers that have very high speed processors. This compatibility problem is easily solved by selecting a Keithley MetraByte, Inc., interface board that is compatible with the processor on board the computer.

Availability

The hardware interface board used in our system is the DAS-1602, available from Keithley MetraByte, Inc. The board with software function library and documentation currently costs about \$1,000. The software is programmed in C++ from Borland, Inc. The run-time programs can be obtained from J. E. Steinmetz at Indiana University for a small development and handling fee (\$50).

REFERENCES

- LAVOND, D. G., & STEINMETZ, J. E. (1989). An inexpensive interface for the IBM PC/XT and compatibles. *Behavior Research Methods, Instruments, & Computers*, **1**, 435-440.
- MARKHAM, M. R. (1993). An interface for controlling external devices via the IBM PC/XT/AT parallel port. *Behavior Research Methods, Instruments, & Computers*, **25**, 477-478.
- MCGREGOR, I. S. (1996). Using Strawberry Tree WorkbenchMac and Workbench PC software for data acquisition and control in the animal learning laboratory. *Behavior Research Methods, Instruments, & Computers*, **28**, 38-48.
- PALYA, W. L., & WALTER, D. E. (1993). A powerful, inexpensive experiment controller or IBM PC interface and experiment control language. *Behavior Research Methods, Instruments, & Computers*, **25**, 127-136.
- PALYA, W. L., WALTER, D. E., & CHU, J. Y. M. (1995). An inexpensive 1-millisecond experiment control interface for IBM PCs and its user-friendly control language. *Behavior Research Methods, Instruments, & Computers*, **27**, 129-130.
- PENTTONEN, M., SALMI, M., HAMALAINEN, P., & MERILUOTO, J. (1994). A microcomputer system for controlling classical conditioning experiments. *Behavior Research Methods, Instruments, & Computers*, **26**, 447-453.
- SCANDRETT, J., & GORMEZANO, I. (1980). Microprocessor control and A/D data acquisition in classical conditioning. *Behavior Research Methods & Instrumentation*, **2**, 120-125.
- STEINMETZ, J. E. (1996). The brain substrates of classical eyeblink conditioning in rabbits. In J. R. Bloedel, T. J. Ebner, & S. P. Wise (Eds.), *The acquisition of motor behavior in vertebrates* (pp. 89-114). Cambridge, MA: MIT Press.
- STEINMETZ, J. E., LOGUE, S. F., & MILLER, D. P. (1993). Using signaled barpressing tasks to study the neural substrates of appetitive and aversive learning in rats: Behavioral manipulations and cerebellar lesions. *Behavioral Neuroscience*, **107**, 941-954.
- THOMPSON, L. T., MOYER, JR., J. R., AKASE, E., & DISTERHOFT, J. F. (1994). A system for quantitative analysis of associative learning: Part 1. Hardware interfaces with cross-species applications. *Journal of Neuroscience Methods*, **54**, 109-117.

(Manuscript received November 13, 1996;
revision accepted for publication April 8, 1997.)