COMPUTER TECHNOLOGY

A microcomputer system for controlling classical conditioning experiments

MARKKU PENTTONEN

University of Jyväskylä, Jyväskylä, Finland

and

MATTI SALMI, PASI HAMALAINEN, and JUHA MERILUOTO

University of Joensuu, Joensuu, Finland

A microcomputer-based laboratory system for controlling stimulus presentations and data acquisition in classical conditioning experiments is described. The system comprises an Intel 386/486-based microcomputer and a commercially obtained low-cost counter/timer board with input/output lines for stimulus timing and external device control. A simple, yet versatile custom-designed structured programming language is provided for performing an unlimited number of stimulus configurations and their sequences. In electrophysiological studies, the system can be flexibly connected to computer-controlled signal conditioning systems for the amplification and filtering of multiunit and evoked field potential responses and to high-speed data acquisition systems for sampling and analyzing the responses. The costs of reserving an entire microcomputer for experiment control are well compensated for by the simplicity and efficiency of programming and transportability of the control protocols between different setups and laboratories. Furthermore, a data acquisition and analysis system most suitable for the aims of a research project can be selected.

We have developed a control system that flexibly performs the timing and delivery of experimental stimuli in classical conditioning experiments. It offers commands for event timing, external stimulus device control, data acquisition system synchronization, and video screen display control. Our system may be a useful investment for experimenters using a wide variety of equipment and systems for behavioral and electrophysiological data collection. It is particularly suitable for experiments requiring continuous A/D conversion with high sampling rates—for example, when multichannel, multiunit, or evoked field response recordings are performed.

Although in classical conditioning experiments only a limited set of stimuli are used and the stimuli are presented independently of the subject's behavior, there is a great variability in their relations and timings (e.g., Gormezano, Kehoe, & Marshall, 1983; Rescorla, 1988). The present experiment control system was therefore designed to be versatile enough to adapt to a wide variety of experimen-

tal situations. To gain maximum efficiency and flexibility in both experiment control and data acquisition, we decided to use a separate computer system for each purpose. The interaction between the stimulus delivery and high-speed data acquisition systems is achieved by providing a trigger signal and a trial-type code from the experiment control system to the acquisition system. The acquisition system can then store the trial-type code together with the actual data, and it can be utilized later in analyzing each trial type separately. The possibility for sorting data reliably by trial type is particularly important when large amounts of data—for example, multiunit and evoked responses—are recorded and averaged.

The control programs are written in a simple-to-learn, structured experiment control language and are converted to an efficient code with a custom-designed compiler. The language has been specifically designed for classical conditioning experiments, so only a reduced set of high-level instructions is used. Thus, even a novice programmer should be able to write, modify, and test a control program in less time than it usually takes to learn the most basic features of a commercially available high-level programming language.

Because only a commercially available low-cost counter/timer input/output board is needed in addition to the entry-level microcomputer, investment in our experiment control system should keep total costs low, even for larger

This research was supported by the Finnish Academy. Address correspondence to M. Penttonen, University of Jyväskylä, Department of Psychology, P. O. Box 35, SF-40351 Jyväskylä, Finland (e-mail: penttone@jyu.fi).

Note: The author has a direct financial interest in some of the software described in this paper.—*Editor*

laboratories. The sharing of software and expertise in experiment control is also an important factor in reducing costs; a control program written in one laboratory could be easily transported to other laboratories.

Application Examples

We have used our system for several years when studying the classical conditioning of the head-turning response in cats. In our experimental procedure, a tone (conditioned stimulus, or CS) is presented to one or both ears of the cat through miniature loudspeakers attached in front of the ears. Electrical stimulation (unconditioned stimulus, or US) is applied to the medial forebrain bundle at the level of the lateral hypothalamus. The amplitude of the head turn (conditioned response, or CR) is then measured with a movement acceleration transducer attached to the cat's head during paired (Korhonen & Penttonen, 1989) and differential (Penttonen, Korhonen, Arikoski, & Hugdahl, 1993) conditioning. The qualitative features of the CR are observed from videorecordings, and this process is guided by the trial information superimposed on the animal's behavior by the experiment control system. The control system has recently been adapted for classical conditioning of the rabbit eyelid/nictitating membrane response with the simultaneous recording of hippocampal field responses and multiple-unit activity.

A signal generator (FG5000, Wavetek, San Diego, CA) is used for auditory stimulation, and an opto-isolated pulse stimulator (Model 2100, A-M Systems, Everett, WA) is used for electrical stimulation of the brain. In nictitating membrane conditioning, the airpuff, which is directed to the cornea through a plastic tubing, is timed by turning an electronic valve on and off.

The electrophysiological signals are preamplified with a custom-designed differential amplifier that is fitted to a connector located on the animal's head. The signals are then further amplified and high- and lowpass filtered with a programmable signal conditioner (Cyber-Amp 380, Axon Instruments, Foster City, CA). At the start of each recording session, the experiment control program automatically adjusts the gains of the amplifiers and cutoff frequencies of the filters to predefined values for each of the eight channels separately. The signal conditioner is controlled by the computer through a standard RS-232 serial interface.

The custom-designed data acquisition system records neural activity from eight channels with a 15-kHz sampling rate, resulting in an aggregate sampling rate of 140 kHz (DT2831 data acquisition board, Data Translation, Marlboro, MA). Such a high sampling rate is needed, as neuronal spikes are separated from the multiple-unit activity by software off line. The control system triggers the acquisition system at the start of each trial and provides a separate trial-type code for paired, CS-only, and US-only trials.

Comparison With Other Systems

One widely used approach for collecting behavioral and electrophysiological data in conditioning laborato-

ries has been to use specific analog circuits or pulse generators for controlling stimulus presentations and for providing a trigger pulse to the recording equipment for synchronizing stimulus presentations and data acquisition. In our microcomputer-based control system, these analog devices have been replaced by a single, low-cost timer/counter input/output board. As a result, the often lengthy and error-prone manual adjustment of stimulus timings and sequences is replaced by a simple programming environment. Separate programs can easily be written to construct different experimental protocols, and each of them can be thoroughly tested before use. Each experimental protocol can then be simply executed by recalling the corresponding program.

Another common practice in classical conditioning laboratories has been to implement both stimulus control and data acquisition on a single computer (Lavond & Steinmetz, 1989; Scandrett & Gormezano, 1980). This approach may have represented the only feasible choice when expensive minicomputers or early microcomputers were used. With the recent considerable decrease in the costs of microcomputers, the necessity of implementing both experiment control and data acquisition in the same system can now be questioned. If experiment control is implemented in a single, low-cost microcomputer system after testing in one data acquisition environment, it can then be used in other environments without modification. Furthermore, because microcomputers are in use in practically every laboratory and the counter/time input/output board is relatively inexpensive, exchanging experimental protocols between laboratories should be relatively easy. Our programming language is a very high level one and easy to read, so it could even be used for the formal representation of ideas to new procedures, even if the actual experiment control system is implemented in some other way.

Although it provides the same flexibility as general-purpose commercial data acquisition and analysis software, our system is more compact. A control program sufficient to conduct a basic eyelid conditioning experiment consists of only a few pages of high-level, easy-to-read code. Widely used commercial systems, such as ASYST (Keithley Instruments, Rochester, NY), are, in general, well documented and extensively tested. The documentation supplied with these expensive systems may, however, consist of hundreds of pages, with no specific examples as to how classical conditioning experiments could be more efficiently controlled. Furthermore, even though timing and input/output control functions are supported for a wide variety of data acquisition boards, only a few such boards have actually been tested in classical conditioning experiments.

A comparable system could also be implemented with a high-level language, such as BASIC or C. The counter/timer input/output board used in our system comes with driver routines that can be called from BASIC or C, providing elementary control of timing and input/output. An experienced programmer can, in fact, design a control system with comparable features by using these dri-

vers. We estimate, however, that programming and extensive testing of such a control program would take several weeks.

Palya and Walter (1993) have presented a similar control system for operant conditioning experiments. In our system the input/output board is installed in the microcomputer, whereas in their system a commercially manufactured input/output board is connected to the parallel printer port of a microcomputer through an adaptor board, with both boards powered by independent power supplies. Their system is specifically designed for operant conditioning procedures, in which the stimulus sequences are dependent on the subject's responses, so the operant responses can be counted on several different input lines. Operant procedures are also feasible in our system because one of the counters of the input/output board is used for detecting TTL-level changes in its input. When a TTL-level output from a response detector is fed to this event counter, the stimulus presentation can be changed according to the subject's responses.

The Walter-Palya system can also be implemented as a network of several commercially obtained processor boards linked to the serial port of a microcomputer (Walter & Palya, 1984). Each processor board, connected to one or two input/output boards, can then control an independent experiment (Palya & Walter, 1993). Because the stimuli in classical conditioning experiments are presented independently of the subject's behavior, our system can also be used for stimulating many subjects simultaneously.

In electrophysiological laboratories, the costs of the experiment control system should be relatively low compared with the total costs of the complete system. The costs of control program development and testing can, however, be substantial. The expense could be less in our system, because the control code is written, tested, and performed in the experiment control microcomputer, whereas the code is transmitted through the relatively slow serial port to the processor board in the networked system.

Implementation Suggestions

Our system should be useful for researchers who are interested in adapting standard or recently developed experimental protocols in a behavioral or electrophysiological laboratory with the ability to select from a wide range of measurement systems. Different protocols can then be tested with a single control system, and then the actual experiments can be conducted by using the data acquisition system best suited to the current problem.

If low-cost, general-purpose laboratory software packages are available, data acquisition should be easy to perform, because our system provides the A/D triggering needed for synchronizing the A/D sampling initiation. If an expensive, general-purpose laboratory software system is available, the programming effort would best be directed simply toward data acquisition and analysis rather than toward trying to incorporate the ex-

periment control procedure into such an extensive and complex system.

A commercial system specifically designed for electrophysiological experiments, with experiment control and data acquisition/analysis opportunities supporting a general-purpose data acquisition board (e.g., DataWave Systems, Longmont, CO; RUN Technologies, Laguna Hills, CA) or a custom-designed interface (e.g., Cambridge Electronic Design, Cambridge, England), is available in some laboratories. If sufficiently detailed experimental control procedures can be implemented by using such a system, and if the data acquisition is performed with the same system, our experiment control system would only slightly increase the total flexibility or efficiency of that system. However, if there are many data acquisition and analysis systems in use in the same laboratory, our system may be useful. If the experiment control procedure is programmed and tested in our system, it would be necessary to learn only one simple and easy-to-use software system for experiment control, thus restricting the need for other software systems for the performance of specific data acquisition and analysis tasks.

In laboratories in which the commercially obtained electrophysiological data acquisition and analysis systems do not provide sufficient data-processing efficiency or capacity, a custom-designed data acquisition system might be easier to program if the inclusion of the experiment control procedure in the same system is not of vital importance. For example, our high-speed data acquisition and analysis system, with a sampling rate of up to 250 kHz, includes extensive on-line data review, as well as off-line data plotting and analysis opportunities. We found that in designing the acquisition system, we were able to concentrate more efficiently on acquisition and analysis when we did not try to include the experiment control process in the same system. Furthermore, both of our acquisition board timers are reserved for the timing of data sampling, so a separate timer board was needed, in any case, for experiment control. A highly complex system was therefore simplified by keeping the experiment control and data acquisition systems logically and physically separate.

System Overview

A low-cost 386/486 SX/DX system with standard memory, VGA video interface, color monitor, and a small hard disk can be used for program development and experiment control. The event timing and external device controls are performed by a low-cost DT2819 programmable counter/timer input/output board (Data Translation, Marlboro, MA). The board is designed around an AMD Am9513A chip and provides five 16-bit counter/timers, a built-in frequency source, and 16 digital input/output lines. The manufacturer supplies an optional screw terminal panel (DT 758-C) for connecting external devices to the board. Alternatively, a custom-designed protective interface with opto-isolation can be used (Hertel & Edgell, 1991).

Two of the five timers are used as the main timer. The third timer is reserved for delivering a 3- μ sec, fixed-length, TTL-level trigger pulse to the data acquisition system to initiate A/D sampling of analog signals. The fourth timer is connected to a switch by which the experimenter can interrupt and later continue the experiment. The last timer is reserved for counting TTL-level events.

Thirteen of the 16 output lines available are provided for external device control. The 3 remaining output lines are reserved for transmitting a trial-type code to the digital input lines of a data acquisition board in another computer. By using binary coding, eight different trial-type codes can therefore be transmitted to the data acquisition system.

The system can also be equipped with a video output board for superimposing experiment information on video recordings. In our system, the video output board (Dighurst, England) is inserted into a free expansion slot in the computer and is connected to the video feature connector of the computer VGA controller by a ribbon cable. The video output board converts the computer display to a video signal and combines this signal with the output signal of the video camera used for behavioral observations of the experimental subject. During experiments, this composite video signal is transmitted to a TV monitor, and during trials it can also be recorded on a VCR recorder. Since the trial information is overlaid on the behavior of the subject, visual observations of that behavior can be correlated, for example, with the movement transducer signals and neural recordings.

Programming Environment

The user interface consists of a single menu, including commands for editing, compiling, and running the programs. The programs are developed within the menu system in three steps. The program is (1) created and modified with a text file editor; (2) compiled with a custom-designed compiler; and (3) executed. The compilation of a program into an executable code typically takes just a few seconds. Any one of these compiled executable programs can later be selected under the menu for further use.

The user interface and the compiler have been written in the C language (Microsoft C 6.0). For maximum efficiency, the parts of the program directly controlling the counter/time board have been written in assembly language (MASM 6.0). The experimenter, however, needs little knowledge about programming, since the custom-designed programming language is based on only a few rules, and only a limited number of high-level commands are used. The flexibility of the system is further increased by the option of including MS-DOS commands in the programs. Thus, output devices or signal conditioning systems equipped with a serial interface can be directly controlled through the computer serial port.

Control Language

The experiment control programs are constructed by defining lists of commands. Each list contains a set of commands designed to perform a specific task. The

trial-type and intertrial interval definitions constitute the most basic lists. These lists are then used to define sessions. Correspondingly, a list performing a task used in many trial types can be constructed and later included as a part of a trial-type list. Within these lists, individual commands are placed sequentially and are separated by commas. Broadly speaking, the lists correspond to the subroutines of the most commonly used programming languages, such as BASIC and C. Because lists written to control a single experiment are usually short, they can all be included in the same program, and no subroutine library system is needed.

Table 1 shows the general structure of a sample program. Integer and list variables are used in the programs, and they are declared at the start of the program by a *Variables* command. Experimental stimuli are the integer variables most often used. In this example, two stimuli—tone and brain stimulation—are defined. The list variables in this example are *InterTrialInterval*, *CSOnlyTrial*, *PairedTrial*, and *PairedSession*. After variable declarations, the actual values of the integer and list variables are assigned by *Set* commands. An integer value is assigned to each stimulus variable. This value represents the output line of the timer/counter input/output board to which the stimulus device is attached. The commands comprising a list are included in brackets (*{* and *}*), but in this example only empty lists are shown. Finally, when all the lists have been defined, a fixed-name list—*ExperimentControlParam.CommandList*—is defined, collecting the whole experiment into a single entity. This list is then executed by an *Execute ExperimentControl* command.

Commands

Commands are provided for external device control, timing, program flow, and screen output. Most commands require some parameters, and the number and type of parameters vary between commands. Parameters can be integer values, text strings, or other commands. Integer or list variables defined by the experimenter can also be delivered as parameters.

External device control. Setting and clearing output lines is accomplished by means of simple commands.

Table 1
Program Structure

Variables Tone, Stimulation, InterTrialInterval, CSTestTrial, PairedTrial, PairedSession
Set Tone = 1 Set Stimulation = 2 Set InterTrialInterval = { } Set PairedTrial = { } Set CSTestTrial = { } Set PairedSession = { }
Set ExperimentControlParam.CommandList = { PairedSession } Execute ExperimentControl

For example, the tone generator is switched on using the *SetLine, Tone* command. Multiple operations can be concatenated: To switch on the tone generator and turn on a light, the command *SetLine, Tone + Light* would be used. Each of the 13 available output lines can thus be independently set, cleared, or changed by the *SetLine, ClearLine*, and *ChangeLine* instructions.

The data acquisition system can be triggered by a fixed, 3- μ sec trigger pulse by using the *SetTrigger* command. The trial number counter is simultaneously increased by one, and the three reserved trial-type lines are updated to represent the trial-type code provided by the parameter. The state of these lines is retained until a new *SetTrigger* command is performed, or the experiment is terminated.

Stimulus timing. All stimulus presentations are synchronized by the *Delay* command, which also determines how long a stimulus combination is kept unchanged. The *SetLine, ClearLine, ChangeLine*, and *SetTrigger* commands are thus not executed immediately, but are collected individually until a *Delay* command is issued in the program. When the main timer has counted to zero downward from the parameter value of the previous *Delay* command, a new event starts. The new *Delay* parameter value is then loaded in the main timer and all pending output lines are changed. Although the delay lengths are given as multiples of 1 msec, the timer uses 0.1-msec steps internally. The accuracy over repeated timings is within 5 μ sec.

Display control. A wide variety of information can be simultaneously shown on the computer screen at different locations by *Display* commands. In addition to text strings and variable values, some system information such as time, date, or current trial number can be written to the screen with *GetTime, GetDate*, and *GetTrialNumber* commands. The subject and session numbers, when obtained from the experimenter at the start of the experiment, can also be displayed on the computer monitor by using the *GetSubjectNumber* and *GetSessionNumber* commands. The readability of the text information can be improved by defining screen attributes by *SetBackGroundColor, SetTextBackGroundColor*, or *SetTextColor* commands and clearing the screen by the *ClearScreen* command.

Even though the system allows extensive control of the screen attributes and the location of the text, the program was not designed for accurate presentation of visual stimuli on the computer screen—that is, for tachistoscopic applications. The inability to present accurately timed visual stimuli is due to the fact that the program does not take into account the precise moment at which the video display controller of the computer causes the stimulus to appear on the screen. With the VGA video-controller, a text positioned in a certain row and column will appear on the screen an average of 7.1 msec later (Dlhopolsky, 1989). If these accuracy constraints are not vitally important, the system can also be used for visual stimulation, provided that the stimulus configurations of

interest can be constructed from the standard computer character set.

Program flow control. Instructions are normally executed sequentially. Looping through a set of instructions is possible, however, by using the *Repeat* command with integer starting and ending values and the *EndRepeat* command linked to it. Any number of commands or lists can be included within the loop, and the loops can be nested. The conditional execution of specific events is achieved by *Case, Else*, and *EndCase* commands. Variable-length delays or a variable number of repetitions are achieved by using the *Random* command, which selects a value randomly in the range defined by a low-limit and high-limit parameter.

One counter of the counter/timer board is reserved for a temporary program suspension. Each grounding of the counter input by an interrupt/suspension switch increases the counter. This counter can then be read with the *CheckInterruptSwitch* command. If the counter appears to have received count pulses, the program is temporarily suspended. After this, an additional grounding of the counter input leads the program to continue and the counter to be reset. Since only the specific execution of the *CheckInterruptSwitch* command leads to the temporary interruption of the program, unintended interruptions during trials are avoided. Instead, the experiment can be temporarily suspended during the intertrial interval—for example, due to technical problems or to the discomfort of the experimental subject.

One of the counters of the board can be used for counting the TTL-level changes in its input. This counter can be reset by the *ClearEventCounter* command. If a *Wait-Event* command is then issued, the program is suspended until a predefined number of events has been accumulated in the event counter. The counter can also be read with the *GetEventCounter* command, and the value can be used for conditional program execution.

The experiment is usually terminated when all the trials have been performed. Termination can also be forced by pressing the *Escape* key on the keyboard after first pressing the interrupt switch.

A Sample Program

A sample program listing is shown in Table 2. The variable, stimulus, and list definitions are the same as those in the example provided in Table 1. Here, the construction of the intertrial interval, two trial types, and complete session are shown in detail.

Inspection of the lists shows that the commands and parameters are separated from each other by commas. Because the number and validity of the parameters are checked during the compilation of the program, all syntactic and most semantic programming errors are avoided.

At the start of the *InterTrialInterval* list, the current trial number is displayed on the screen, starting at column 6 of row 3. Next, an intertrial interval of 25–45 sec is selected randomly and executed. This is achieved by looping a 1,000-msec delay 25–45 times. To allow a tem-

Table 2
A Sample Program

```

Variables Tone, Stimulation,
InterTrialInterval,
CSTestTrial, PairedTrial,
PairedSession
Set Tone = 1
Set Stimulation = 2

Set InterTrialInterval = {
  Display, 3, 6, GetTrialNumber,
  Repeat, 1, Random, 24, 42,
  CheckInterruptSwitch,
  Delay, 1000,
EndRepeat }

Set CSTestTrial = {
  InterTrialInterval,
  Display, 23, 1, "CS-Only",
  SetTrigger, 1, /* Pre-period */
  Delay, 250,
  SetLine, Tone, /* CS-period */
  Delay, 1000,
  Delay, 500, /* US-period */
  ClearLine, Tone, /* Post-period */
  Delay, 500 }

Set PairedTrial = {
  InterTrialInterval,
  Display, 23, 1, "Paired",
  SetTrigger, 3, /* Pre-period */
  Delay, 250,
  SetLine, Tone, /* CS-period */
  Delay, 1000,
  SetLine, Stimulation, /* US-period */
  Delay, 500,
  ClearLine, Tone + Stimulation, /* Post-period */
  Delay, 500 }

Set PairedSession = {
  Repeat, 1, 9,
  CSTestTrial,
  Repeat, 1, 9,
  PairedTrial,
  EndRepeat,
EndRepeat }

Set ExperimentControlParam.CommandList = {
  PairedSession }
Execute ExperimentControl

```

porary program suspension at the start of each 1,000-msec delay, the program also checks whether the interrupt switch has been pressed.

The two trial types are then defined. The CSOnlyTrial starts by the execution of the earlier defined InterStimulusInterval list. Then a "CS-Only" message is displayed. The SetTrigger command is then issued to trigger the A/D sampling in the data acquisition system and to send the trial-type code. The next event, which consists of delivering a tone stimulus for 1,000 msec, follows after a delay of 250 msec. To preserve a structural similarity with the PairedTrial, in which the tone is also 1,500 msec long but coterminates with a 500-msec US, a 500-msec US period is added without any changes in output lines.

After this, the tone is switched off, and a 500-msec delay follows before the start of the next intertrial interval.

The PairedTrial is similar to the CSTestTrial, except that a different trial-type code is sent to the acquisition system with the SetTrigger command. In addition, at the start of the US period, the stimulator is switched on, and at the end of the US period both tone generator and stimulator are switched off.

The PairedSession list is formed by defining a 10-trial block composed of 1 CSTestTrial and 9 PairedTrials. This trial block is then repeated nine times, resulting in a total of 90 trials. In this example, only the PairedSession list is included in the ExperimentControlParam.CommandList and is finally executed by the Execute ExperimentControl command. However, other lists similar to the PairedSession list—for example, an UnpairedSession list—can also be constructed. At the start of the experiment, the experimenter can be asked to choose one of them for execution.

Availability

Provided that one unformatted, high-density, 3.5-in. floppy disk (1.44Mb) is included in a request letter, a text file consisting of a complete listing of an experiment control program for classical eyeblink conditioning will be sent to any interested researcher. This program contains procedures for conducting paired, unpaired, and backward eyeblink/nictitating membrane conditioning. A detailed documentation of the control language is also included. The complete programming system, not including the MS-DOS operating system and a text file editor, can be purchased for \$140.

Conclusions

Because of the large installation base of Intel 386/486-based microcomputers and the relatively low purchase price of the counter/timer board, the system is quite affordable. At the least, it includes the same control features that have been previously implemented on the now commercially unavailable PDP-8 (Getty, 1975) and KIM-1 (Solomon, Weisz, Clark, Hall, & Babcock, 1983) computers. Since our custom-designed, special-purpose programming language is easy to read and economic in expression, and since programs are easy to modify and test, it features the most desirable characteristics of a real-time control language (Balsam, Deich, O'Connor, & Scopatz, 1985). In addition, sets of frequently used commands can be grouped to form lists, which can later be reused. Most important, the language provides facilities for the easy and efficient handling of external devices, screen output, and event timing in conditioning experiments.

REFERENCES

- BALSAM, P. D., DEICH, J., O'CONNOR, K., & SCOPATZ, R. (1985). Microcomputers and conditioning research. *Behavior Research Methods, Instruments, & Computers*, 17, 537-545.
- DLHOPOLSKY, J. G. (1989). Synchronizing stimulus displays with mil-

- lisecond timer software for the IBM PC. *Behavior Research Methods, Instruments, & Computers*, **21**, 441-446.
- GETTY, D. J. (1975). The PEPL system for control of experiments by a PDP-8 computer. *Behavior Research Methods & Instrumentation*, **7**, 131-136.
- GORMEZANO, I., KEHOE, E. J., & MARSHALL, B. (1983). Twenty years of classical conditioning research with the rabbit. In J. M. Sprague & A. N. Epstein (Eds.), *Progress in psychobiology and physiological psychology* (pp. 197-275). New York: Academic Press.
- HERTEL, S. A., & EDGELL, S. E. (1991). Input protection for the laboratory computer. *Behavior Research Methods, Instruments, & Computers*, **23**, 387-394.
- KORHONEN, T., & PENTTONEN, M. (1989). Behavioral and neural characteristics of short-latency and long-latency conditioned responses in cats. *Behavioral Neuroscience*, **103**, 944-955.
- LAVOND, D. G., & STEINMETZ, J. E. (1989). An inexpensive interface for the IBM PC/XT and compatibles. *Behavior Research Methods, Instruments, & Computers*, **21**, 435-440.
- 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.
- PENTTONEN, M., KORHONEN, T., ARIKOSKI, J., & HUGDAHL, K. (1993). Effects of lateralized US and CS presentations on conditioned head turning and bilateral cingulate cortex responses in cats. *Behavioral & Neural Biology*, **59**, 9-17.
- RESCORLA, R. A. (1988). Behavioral studies of Pavlovian conditioning. *Annual Review of Neuroscience*, **11**, 329-352.
- SCANDRETT, J., & GORMEZANO, I. (1980). Microprocessor control and A/D data acquisition in classical conditioning. *Behavior Research Methods & Instrumentation*, **12**, 120-125.
- SOLOMON, P. R., WEISZ, D. J., CLARK, G. A., HALL, J., & BABCOCK, B. A. (1983). A microprocessor control system and solid state interface for controlling electrophysiological studies of conditioning. *Behavior Research Methods & Instrumentation*, **15**, 57-65.
- WALTER, D. E., & PALYA, W. L. (1984). An inexpensive experiment controller for stand-alone or distributed processing networks. *Behavior Research Methods, Instruments, & Computers*, **16**, 125-134.

(Manuscript received August 31, 1992;
revision accepted for publication March 1, 1994.)