A high-speed computer-compatible event recorder*

JERE BEAUCHAMP and STANLEY R. SCOBIE

State University of New York, Binghamton, New York 13901

Design considerations, construction information, and programming considerations of a high-speed event recorder are presented. The device records discrete events on magnetic tape for later processing by a large computer. Although the design information is general, the instrument has 120 event channels, a time resolution of up to 100 microsec, and can record 60 or more events per second.

Several authors have reported techniques for recording events in real time for later processing by an off-line computer (Barry, Kinnard, Watzman, & Buckley, 1966; Platt & Senkowski, 1968). Rapid changes in instrumentation technology have outdated these earlier systems, which used punched paper tape. It is, for example, difficult to find a computer center that accepts punched tape for processing. Yet, with very large data sets to be manipulated, analyzed, summarized, and presented in readable form, on-line analysis is often inadequate.

Some investigators routinely use magnetic tape for bulk storage of real-time data within the context of a small-to-medium-size laboratory computer. With small laboratory computer prices rapidly decreasing, the use of such a combination appears relatively economical. However, there appear to be several advantages to the sort of system to be described here. A stand-alone data acquisition system can operate independently of any on-line computer, and thus is not inextricably tied to a particular experimental arrangement. Also, in a large system, such as the one to be described, the data input buffers can serve as an interface to a computer controller at the same time they record data on tape for analysis. Thus, construction of a duplicate set of input buffers for the controller can be avoided.

The system design presented here provides careful consideration of the tape read/write characteristics of the computer center facilities used for subsequent processing. The data format reported by Platt and Senkowski (1968) and those used in commercially available units (e.g., Massey-Dickenson Co., Saxonville, Mass.) appear to be difficult to handle with contemporary high-level programming languages. For example, response data blocks of uniform length greatly facilitate subsequent analyses.

The description that follows is intentionally rather detailed; we have tried to present sufficient information to allow implementation of such an instrument by a moderately sophisticated electronics technician.

*This work was supported, in part, by Grants MH-18898 and MH-19961 from the National Institute of Mental Health and a grant from the SUNY Research Foundation. Reprints may be obtained from either author: Department of Psychology, State University of New York, Binghamton, New York 13901.

DESIGN CONSIDERATIONS

Our primary concern was to develop an instrument that could monitor a large number of discrete events, detect their occurrence, and record their time of occurrence on IBM-compatible magnetic tape. Practical electronics considerations indicated a maximum number of channels of 256 or less; general experimental needs appeared to be well satisfied with 120 external events and 8 internal control events.

Time resolution requirements were to be 1 msec or less, with capability to measure time intervals greater than 1 h accurately. High time resolution allows the recording of analog signals converted to binary code, and the capacity to handle long time intervals allows the study of processes over extended periods of time. Preservation of data and maintenance of time resolution in high-speed bursts was necessary. Data and time resolution could not be sacrificed during bursts of activity that exceeded the recording rate of the available magnetic recorders and events shorter than internal cycle times or recorder cycle times. Because we wanted to be able to use this recording system with as great a variety of equipment as possible and also in a number of different physical locations, noise immunity and fault protection were required. Interface to electromechanical equipment required suppression of contact bounce, noise, and protection from high voltages.

Encoding and Recording

We considered several event encoding and recording schemes. The first was to enter the logic states of all events on a fixed time scale. The second was to use a vectored interrupt system similar to those found in more recently designed small computers. The third method was to use a polling technique, where an indication of an event occurrence was generated and all locations were interrogated in order until the first active channel was located.

The first method would have provided a synchronous system. A record clock would record all channels simultaneously, regardless of their activity. To maintain 1-msec resolution would have required that such clocking occur every 1 msec. Allowing for optimum packing on the magnetic tape and 120 event channels would have resulted in a data rate to the recorder of 20 KHz. Considering that an interrecord gap (IRG) would be required after approximately 100 blocks, this would have consumed tape at the rate of 30 IPS (in./sec) at a packing density of 800 BPI (bits/in.). This method also would create some fairly difficult programming problems. There was no simple method for error recovery should the system add or delete a character, nor was event identification readily available without bit manipulations within the analysis program.

Elimination of the first method dictated that we

Fig. 1. Overall system block flow diagram.

record a unique identification code along with the time of event occurrence. During periods of no activity on the event lines, no information need be recorded.

A vector interrupt scheme provides for a potentially very high speed of encoding. From the time of a request to encode to the completion of the encoding could easily be accomplished in less than a microsecond. The major difficulty, however, is in the total number of parts required for each event channel. Seven bits are required to encode 128 unique event lines, which requires a minimum of two integrated circuits for bus connection along with additional integrated circuits for the maintenance of a priority scheme and resetting of flip-flops used for temporary response storage. Thus, we also rejected this method.

We chose event channel polling as the best compromise. The search always begins at zero and requires 128 interrogations for the highest numbered event. However, the interrogation and wait for reply can easily be accomplished in 1 microsec, so that the worst case of encoding time is 128 microsec. This also yields a minimum size event channel cell, consisting of three gates and one J-K-type flip-flop with minimal supporting circuitry. Such synchronous operation of the system greatly relaxes the requirements for the digital recorder; information is recorded only as it is generated, which greatly reduces the volume of the records. Creation of the IRGs is also better controlled.

To prevent swamping of the recorder during bursts of data, a small-core memory is used as a buffer. All generated data is loaded into this memory in proper format for the recorder. The memory used consists of 512 words of 8 bits each. An even larger memory would allow for longer bursts before data loss. Any core memory overflow results in the oldest responses being lost and some nonsequential data being entered on the tape, since a newer response is substituted for an older response. As part of each analysis program, the sequence of real time is checked and action taken with any out-of-sequence events. No indication of core memory

overflow has been detected under usual methods of data collection (i.e., data rates of less than 60/sec).

Generation of time information for events is subdivided between hardware-generated information and the information generated by the analysis program. An internal dedicated event line enters an event every 10 sec when the internal clock overflows (cf. Platt & Senkowski, 1968). The final data record entered on the tape consists of a five-digit time code following a three-character event identification. To facilitate error detection, the event code consists of a single alphabetic character followed by a two-digit octal number. If the analysis program finds extraneous information on the tape, it can take immediate corrective action.

Data Input

The event input module consists of two sections: (1) a fault protection, noise immunity, and display section, and (2) an event storage section, shown in the block diagram of Fig. 1 (event input section) and in the detailed schematic of Fig. 2. In Fig. 2, Resistor R1 and Capacitor C1 provide a low-pass network, while Resistors R2, R3, and Gate 1A form a Schmitt trigger. Signals entering must charge C1 to the Schmitt trigger threshold voltage. The noise immunity into this gate is very high. With the values shown, no triggering above 10 KHz was observed with a 0-10-V square-wave input. With standard 5-V logic levels, susceptibility to line-induced noise was very low. Driving the input from another logic gate through 50 ft of wire wrapped tightly around the motor case of an electric hand drill did not generate a spurious output.

Fault protection consists of Resistors R1 and R2 and Zener Diode D1 for positive voltages. Above the Zener breakdown voltage, there is effectively zero impedance contributed by the diode to ground. This clamps the input to 4.7 V; fault protection is then limited to the power rating of the resistors (typically $\frac{1}{4}$ W). This allows for a continuous input voltage of 45 V without exceeding the ratings of the components. For negative

Fig. 2. Partial schematic of the data input module. Component labels follow conventional electronics terminology. Logic element labels refer to explanations in the text.

inputs, the Zener diode is forward biased, limiting the voltage at the gate to the forward voltage drop of the diode, or about .7 V. This circuit can withstand short exposures to normal 117-V ac line voltage without failure.

Because some indication of system activity is required by the user, an indicator light for each event channel is provided. Since events of short duration are processed, a method of displaying these events to an O is provided by the drive circuit (Fig. 2), composed of Gate 1B, Resistors R4, R5, and R6, Capacitor C2, and Transistor Q1, which forms a one-shot pulse of duration approximately 1.1 (R4) (C2) sec for short pulses or a signal equal to the input for longer events. The indicators used are Hewlett-Packard 5082-4403 light-emitting diodes; these have no current surge on turn-on, low power consumption, small size, and are compatible with logic circuitry.

The event buffer (Fig. 2) consists of a J-K flip-flop (FF1) and three NOR gates (2A, 2B, and 2C). The operation of the cell is as follows: the resting state of the cell is with the flip-flop reset ($Q = 0$), a logic 1 on the decoder input and reset lines, and a logic 0 from the noise immunity and fault protection circuit. A 0-1 transition from the noise immunity and fault protection module is inverted to a 1-0 transition by Gate 2A. This 1-0 transition causes the flip-flop to change states to $Q = 1$. Further 1-0 transitions on the clock line of the flip-flop are prevented by the Q of the flip-flop disabling Gate 2A until the flip-flop is reset.

The Q of the flip-flop is coupled to a multiple input gate which effectively produces a logical OR function, indicating that there is data in the system to be processed. This is used to initiate a search of the system for the active point. At some later time, the decoder line is set to logic 0. Since Q is at logic 0, this produces a logic 1 on Gate 2C, this 1 is coupled to a multiple input gate which again produces a logic 1, indicating that the cell is set. This causes the search to cease at that point. A logic 0 on the reset line in conjunction with the decoder at logic 0 gives a logic 1 on Gate 2B, which resets the flip-flop. This flip-flop is then free to respond to transitions on the clock line.

Scanning Logic

The scanner (Fig. 1) consists of two 128-input OR gates, a 7-bit binary counter, a 1-out-of-128 decoder, and control logic. The two multiple-input OR gates are connected to all cells. The first is used to initiate a search cycle, the second indicates that an active cell has been found. Normally, the binary counter rests at zero. An indication of an active event causes the scanning logic to become active with the first clock transition. A strobe to allow testing of the event-found gate is generated. If no event is found, the binary counter associated with the scanner is pulsed. This causes the counter to increment and the next location to be decoded. A delay is introduced to allow the counter and decoder to settle. After this, another strobe to test if an event had been found is generated. If no event is found, the cycle is repeated.

When the first event that had been set is found, the counter stops and a signal is generated to the data load control, indicating that an event has been found and further processing is required. The contents of the binary counter provide the event identification information required for event recording. The contents of this counter are recorded in the buffer core memory and the scanning logic is reset immediately thereafter. Occurrences of more than a single event simultaneously pose no major problems in that only the particular cell encountered is reset at the end of a cycle. Remaining cells are immediately encoded in numerical order as soon as the scanning logic has been reset.

Conservative estimations of speed for the scanning and search logic indicate a total settling time of 540 nsec, which allows a search rate of approximately 1.8 MHz, or a worst-case encoding time of approximately 70 microsec. The availability of a precise 1-MHz clock from the real-time clock outweighed the speed increase from a separate scanner clock; thus, the worst-case encoding time is 128 microsec.

Real-Time Clock

Requirements for the system include the availability of a fairly wide range of frequencies for internal control and great stability and accuracy for consistent

time-interval measurement. A crystal oscillator, using an AT cut crystal of 1 MHz (Reeves-Hoffman Corporation, Carlisle, Pennsylvania), is used. Temperature coefficients of this crystal indicate that maintenance of a 15-to-20 parts per million accuracy over the limited (15°C to 50°C) ambient temperature change expected can be achieved without the use of a crystal oven or other compensating techniques.

The 1 MHz from the crystal oscillator passes through a five-decade divider chain, which provides all divisions of 2 and 10 to a minimum of 1 Hz. The remaining circuitry is the five-decade counter and associated open collector NAND gates for recording the clock time on command from the data load control. The input to this counter is connected to the appropriate frequency from the divider chain, and the overflow from this counter is logically inverted and connected to the overflow event channel.

Data Load Control

After an event is encoded, the output record is created. This includes formatting of the event identification and attachment of the current value of the running time clock. The general logic of the data load control (Fig. 2) is composed of a series of open collector NAND gates for data reading, a binary counter and decoder for control of the character being read, and logic to provide timing and control of the reading cycle.

When a signal from the scanning logic has been generated, indicating that an event is encoded, the data load control becomes active. The first data location, containing the most significant bit of the scanning logic binary counter along with two nonnumeric control bits, is loaded into a temporary register. A request to the core memory buffer for a clear and write cycle is then made. The binary counter is then incremented to address the next sequential character to be recorded. The order of data recording, after the first character, is (a) the next three most significant bits of the scanning logic binary counter, (b) the three least significant bits of the scanning logic binary counter, (c) the most significant digit of the running time clock, and (d) the remaining four digits of the real-time clock in descending order of significance.

The data load control logic and sequence length is constructed to allow additional characters of data to be entered from external devices. For example, if an event is the completion of a count cycle or an analog-to-digital conversion, this information can be entered through the data load control by lengthening the sequence and using the scanning logic decoder to address the external device.

Core Memory Buffer

All information to be recorded is loaded into the 512 8-bit-word ferrite core memory buffer and retrieved from it (Fig. 1). Total memory cycle time is 10 microsec

for a full cycle consisting of a read-restore operation or a clear-write operation.

The additional circuitry supporting the core memory buffer includes: memory cycle control logic, two independent address counters, an address selector and register, and an address comparator. The memory cycle control logic provides the dynamic timing to control the core memory and to determine the cycle to be executed. The timing constraints on the memory include: (a) the time interval that an address must be stable prior to the initiation of a cycle, (b) how soon after the end-of-cycle pulse generated by the memory another cycle can be started, (c) how long signals must be valid, and (d) when data being read from memory is valid. Since two relatively independent subsystems can request memory cycles, a priority system arbitrates simultaneous requests to: (a) load data into memory or (b) to read data already loaded. As implemented, the read-from-memory request is given the higher priority.

The two independent memory address counters have a designed length of 12 bits each, allowing the addressing of up to 4,096 unique memory locations, should a memory of that size become necessary in the future. The two address counters are connected to a comparison circuit which gives a logic 1 level when the two address registers are equal. The address counters are also connected to a selector and address register. When a memory cycle is initiated, the proper address counter is selected by a signal from the memory cycle control. The contents of this counter are loaded into the address register and automatically advanced to the next count. The contents of the memory address register thus remain unchanged during the memory cycle.

Data Output Control and Digital Recorder

The major concern was to create an output system that would operate with as wide a range of recording devices as possible, even though the primary concern was to generate IBM-compatible magnetic tape. The general scheme of operation of the data output control was to monitor the address counter comparator. If the input address does not equal the output address, new data has been entered into the buffer memory. When such an inequality is detected, an immediate request for a read-restore cycle is made to the memory control logic. The result of the read-restore cycle is loaded into the first of two temporary registers and a flag is set, indicating that it has been loaded. If the second temporary register is not loaded, the contents of the first temporary register are loaded into the second register and a flag, indicating that the second temporary register is set, and the first temporary register flag is reset. If an inequality still exists in the address counters, another request for a read-restore cycle is initiated to load the first temporary register. With data in the second temporary register, the next available complete clock pulse is used to generate a record command to the recording device. At the completion of the record

command, the second temporary register flag is reset, permitting data to be loaded from the first temporary register.

A requirement for creating IMB-compatible magnetic tapes is that interrecord gaps (IRGs) be inserted in the data. Also, program execution speed is increased by maintaining fixed-length records. Control of the IRG is maintained by counting the number of events recorded on the tape. Clear indication of a new event is found by the presence of a logic 1 in Bit Positions 6 and 7. These are used to increment an IRG counter. When the IRG counter is full, the next event causes the generation of a standard IRG. During the IRG interval, the clock pulses for recording are suppressed and the output lines to the recorder are set to logic 0. A standard record length of 256 events (or 2,048 characters) was selected as being sufficiently long for programming and short enough for adequate system performance.

The specific recorder used is the Digi-Data Corporation 1300 series incremental magnetic tape recorder. At the unit's maximal recording rate of 500 char/sec, an effective rate of over 60 events/sec can be maintained continuously. Record clock pulses for the tape drive are generated from the system real-time clock (Fig. 1).

System Control and Indicators

Toggle switches, which allow connection to an external data signal for normal recording or to a front panel pushbutton switch for manual testing, are used for each event. Visual indicators (light-emitting diodes) for all event channels are provided.

The contents of the 7-bit event scanner binary counter are stored in a temporary register and displayed visually; this provides the event code for the last encoded event (Fig. 1). As an indication of system activity, the address comparator logic signal is also displayed. The system is idle as long as this signal is on. Also, the two memory address counters are displayed.

The operational controls contain pushbuttons to reset the two address counters and the IRG counter to zero, and toggle switches to prevent data from being loaded into the buffer, to inhibit clock pulses to the recorder, and to complete a standard 256-event block with null events and halt ("go to IRG" for tape loading and unloading). To allow entry to control information on the tape for later retrieval and identification, six operator-only channels are provided. To prevent accidental operation, entry of a control event requires simultaneous or overlapping closure of the enable switch and event mark switch.

CONSTRUCTION¹

The bulk of the electronics is on 5.3 x 7.0 in. double-sided printed circuit boards. The etched card edges mate with an ELCO Corporation 6061 card edge connector with .200-in. spacing. Circuit interconnection is done with 24-ga wire wrapping.

Table 1
Approximate Costs for the System as Described

Power supplies: +5V, 12A; +12V, 3A; -12V, .5A	\$ 200
Resistors, capacitors, transistors, and diodes	500
Wire and cable	50
Printed circuit boards	100
Light-emitting diodes and switches	250
Digi-Data Model 1300 seven-track recorder, 200 BPI	\$2400
The following items were obtained through Federal surplus sources and the cost here is based on approximate commercial equivalents.	
Ferrite core memory	\$ 900
Digital integrated circuits (Series 7400 and Signetics Corporation Series SP300 and SU300)	800
Instrument case, chassis slides, power supply mounting, and ventilation	250
Wire-wrap panel, 72 card locations	250
Relay cabinet	150
	\$5850

The event input and scanning logic for 120 event channels requires a total of 27 printed circuit boards, using five different card designs. The data load control logic, the real-time clock, the data output control, and memory cycle control logic are contained on one circuit board each. Temporary storage registers require a total of five circuit boards of the same design. The address counting, selecting, and comparison require 4 circuit boards; and miscellaneous functions, including connections to front panel, incandescent indicator drivers, connection of external events, etc., require a total of 11 other card locations.

The system electronics (excluding the tape drive) requires 5 V at 12 A, +12 V at 3 A, and -12 V at 1 A. Forced air ventilation, especially on the power supply section, is necessary. (Information as to cost and construction is given in Table 1 and in the footnote.)

PROGRAMMING

Two major programs for the analysis of data generated by the system have been written. The first is for data from eight Ss in an avoidance training situation, the second is for cardiac rate data collected for pairs of Ss in a classical conditioning experiment.

The avoidance training program is organized to handle up to eight different stimulus events, eight S response events, and eight shock events. The program can recognize different groups of Ss over days or sessions. This program provides three different kinds of output: (a) an expanded record of all stimuli and responses as they occurred; (b) a collapse of S responses by stimuli, including latency of first response, total responses during the stimulus, and number of shocks accumulated, and totals of latency, responses, shocks, and probabilities; and (c) a sort by S and by group of response data.

In the cardiac conditioning analysis program, the following is done: (a) the validity of each heart beat for each S is tested by percentage limits of the average of the last five heartbeats recorded for that S, (b) an expanded record of each heartbeat is printed during

```

SAMPLE: PROC OPTIONS(MAIN);
STMT LEVEL NEST
1
2 1 SAMPLE: PROC OPTIONS(MAIN);
3 1 DCL RENUMBER(1:8) FIXED(1) INITIAL(1);
4 1 DCL FIRST CHAR(1);
5 1 DCL CORR FIXED(2);
6 1 DCL TIME FIXED(5);
7 1 DCL APC(16) FIXED(9) INITIAL(0);
8 1 DCL RTG FIXED(13,4);
9 1 DCL IO FIXED(1);
10 1 GET DATA;
11 1 READ: GET FILE(INPUT) EDIT(FIRST, CODE, TIME, RTG, CORR, FIRST);
12 1 IO = CODE * 100 / FIRST * 1000;
13 1 RTG = ACCUM + TIME / 100000;
14 1 IF IO < 101 THEN GO TO CONTROL;
15 1 IO = RENUMBER(1:8);
16 1 IF IO < 101 THEN GO TO READ;
17 1 IF IO < 101 THEN GO TO RESPONSE;
18 1 IF IO < 201 THEN GO TO STIMULUS;
19 1 IF IO < 301 THEN GO TO SHOCKS;
20 1 IF IO < 401 THEN GO TO CONTROL;
21 1 IF IO < 501 THEN GO TO READ;
22 1 IF IO < 601 THEN GO TO RESPONSE;
23 1 IF IO < 701 THEN GO TO SHOCKS;
24 1 RESPONSE:
/* HANDLING ROUTINES FOR SUBJECT RESPONSES */
STIMULUS:
/* HANDLING ROUTINES FOR STIMULUS DATA */
SHOCKS:
/* HANDLING ROUTINES FOR SHOCK DATA */
CONTROL:
/* THE UPDATING OF THE ELAPSED TIME OF EVENT IS AS FOLLOWS */
25 1 IF IO < 11 THEN ACCUM = ACCUM + 100;
26 1 GO TO READ;
/* ADDITIONAL PROGRAMMING FOR INFORMATION ENTERED BY EY */
27 1 END SAMPLE;

```

Fig. 3. Sample analysis program for avoidance conditioning data (refer to text).

stimulus intervals and pre- and poststimulus intervals, (c) heart rate data is sampled during the intertrial interval, (d) graphs of heart rate are produced for the stimulus defined as the conditioned stimulus, and (e) means and standard deviations of both interbeat intervals and beats-per-minute are calculated for each stimulus.

The analysis programs do not assume any specific event channel assignments, but rather allow for relatively simple modification of the assignments through passed data from the SYSIN file.

A sample program (Fig. 3) was taken from the avoidance conditioning program for illustrative purposes. The coding is in PL/I level F (IBM, 1970a, b). The initial statements (2-8, Fig. 3) establish the program variables in terms of magnitude, number representation, and initial values. The range of the numbers is established according to the anticipated range, and the number representation is decimal. While fixed binary and binary floating point would have been less time consuming for arithmetic processing, large amounts of output data are created. The loss of efficiency in the arithmetic processing was more than compensated for in increased speed of output operations using decimal arithmetic.

A single statement "GET DATA" (Statement 9) allows the setting of all required variables in the program, especially "RENUMBER" (Statement 2). The information to accomplish this is located in the standard "SYSIN" file. The tape created by the event recording system is found in the file "INPUT" (Statement 10). The data description to link the tape to the program is done in the job control language (JCL) supplied with the program at run time.

Input operations from the data tape are performed by edit-directed stream transmission as opposed to record transmission. Any errors occurring on the tape cause a program interrupt to occur, which can be identified and

dealt with by an appropriate "ON" unit. Very few tape errors are typically found. However, without some programmed method of dealing with errors, the standard system action is to raise an error condition and terminate the program immediately.

After reading eight characters of tape data (one event block), the initial alphabetic character delineating event records is converted along with the remaining event code to a single number giving the event identification (Statement 11). The time occurrence of the event is computed, using both the information taken from the tape and the information generated by the program (Statement 12). The magnitude of the program time clock is a 13-digit number, capable of well over 250,000 h.

The control events within the system are given the first eight event channel positions; thus, their calculated identification numbers are from 0 to 7. Event 0 is a null event, used only to complete records. Event 1 is connected to the overflow of the system clock. Events 2-7 are operator entry events, whose significance varies from experiment to experiment. A test is made of all events to check for control information. If a control event is found, an immediate branch to the control section is made.

The general tactic taken for an experimental program was to create a renumbering array (Statement 2) that would number the Ss, responses for the S, and stimuli for the S. Thus, a response from S 1 from the arbitrary event channel assigned to S 1's response would be renumbered as 1, stimuli to S 1 would be renumbered as 11, shocks to S 1 would be renumbered as 21, etc. (Statements 15-22). After entry to the proper area of the program, a constant (i.e., 10, 20, etc.) was subtracted to identify the S.

Subroutine calls or subprocedures are not used, since a high degree of modularity was obtained by using the techniques described. Further internal compiler-generated bookkeeping and other overhead associated with subroutines is not incurred.

SUMMARY AND CONCLUSIONS

This description of a data recorder is intended as an aid to workers who must either build or provide design specifics for such an instrument. A combination of a small laboratory computer and a digital magnetic tape recorder could be similarly used. However, much of the same input buffering would have to be done as described here. We should note that users and builders of systems such as this face problems of obsolescence. Just as the paper tape of earlier systems is no longer accepted at computer centers, magnetic tape specifications can also change. Close cooperation with directors of computer centers can ameliorate, but not eliminate, such problems.

REFERENCES

Barry, H., III, Kinnard, W. J., Jr., Watzman, N., & Buckley, J. P.

A computer-oriented system for high-speed recording of operant behavior. *Journal of the Experimental Analysis of Behavior*, 1966, 9, 163-172.

Beauchamp, J., Donovan, P. J., & Burright, R. G. An automated system for long term and continuous computer compatible recording of feeding, drinking, and general activity behaviors of rodents. *Physiology & Behavior*, 1972, 8, 765-768.

Ellen, P., & Wilson, A. S. Digital recording of simultaneous events in multiple operant chambers. *Journal of the Experimental Analysis of Behavior*, 1964, 7, 425-429.

Herrick, R. M., & Denelsbeck, J. S. A system for programming experiments and for recording and analyzing data automatically. *Journal of the Experimental Analysis of Behavior*, 1963, 6, 631-635.

IBM. System/360 operating system: PL/1 (F) language reference manual. International Business Machines Corporation, 1970a.

IBM. System/360 operating system: PL/1 (F) programmer's guide. International Business Machines Corporation, 1970b.

Platt, J. R., & Senkowski, P. C. A computer-compatible

technique for recording binary events in real time. *Behavior Research Methods & Instrumentation*, 1968, 1, 79-81.

NOTE

1. Major documentation for this system is in two parts: (1) schematics and printed circuit negatives of the various printed circuit boards, and (2) a wire wrap list indicating card locations and pin-to-pin connections. These items, along with a brief description of the function and component population of the printed circuit boards, can be supplied at cost (estimated at \$150). Complete engineering information on the system does not exist in a form sufficient to allow construction by an independent electronic job shop.

(Received for publication April 2, 1973;
revision received June 19, 1973.)