Behavior Research Methods & Instrumentation
1982, Vol. 14(4), 409-411

COMPUTER TECHNOLOGY

Experimental control and data acquisition with BASIC in the Apple computer

FREDERICK RAYFIELD
Roosevelt University, Chicago, Illinois 60605

Laboratory experiments can be precisely controlled and data can be collected using the BASIC language on an Apple II+ computer with 48-KB RAM, disk drive, and two timer-I/O cards. The Apple BASIC makes machine language output control routines necessary, but quite convenient. By compiling BASIC, 50-100 inputs/sec are handled. Applications range from operant research and analog data collection to use of the Apple color display capability for stimulus presentations and response recording.

Rayfield and Carney (1981) described a general technique for using BASIC to control experiments with 6502-based microcomputers. The specific system exemplified was the Rockwell AIM-65. Machine language routines that operated on an interrupt basis transparent to BASIC were used to poll inputs and keep track of time. This allowed the use of BASIC as the main control language; BASIC is otherwise too slow for the numerous tasks.

This article describes the implementation of the previously described technique on an Apple II+ with disk drive and added timer/parallel interface cards. The system has 16 input and 48 output lines. In addition to discrete laboratory recording and control, it can be utilized for real-time recording and averaging of analog signals (e.g., evoked potentials) with the addition of an analog-to-digital (A/D) converter. Essentially the same routines described for the AIM-65 are used in the Apple for input polling and passing to BASIC, and timekeeping is based on counting 100-Hz interrupts. Familiarity with the Rayfield and Carney (1981) article is assumed.

HARDWARE

The system, shown in Figure 1, uses an Apple II+ computer with 48-KB RAM and one disk drive. The system also works with a diskless Apple, using the cassette recorder interface to store programs. An Apple with less than 48-KB RAM could be used, but the machine language software described here would have to be relocated. An Integer BASIC card, which has a mini-assembler, is helpful for entering machine language routines, but it is not required.

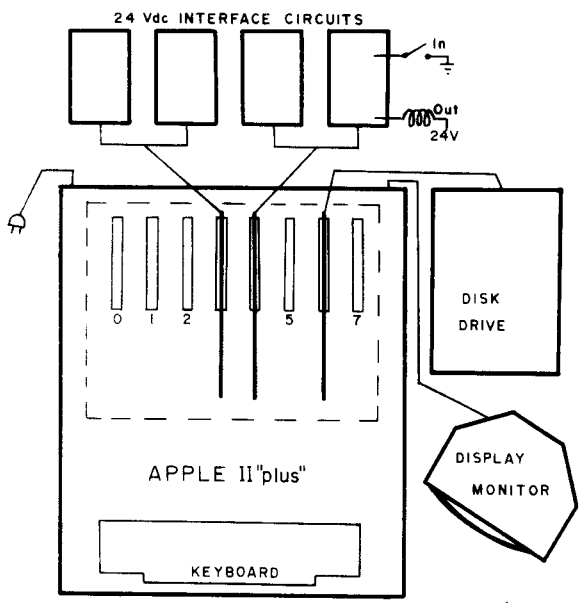

Figure 1. Apple-based system for psychology laboratory recording and control, shown with timer-I/O expansion boards (32 lines each) in Slots 3 and 4, with associated 24-V dc interfacing, and disk controller in Slot 6.

The Apple, unlike the AIM-65, does not have an on-board timer or I/O ports. These must be added to the expansion slots. The simplest and least expensive solution is to install two expansion boards, each with two 6522 versatile interface adaptor (VIA) chips identical to the timer-I/O chip on the AIM-65. (These boards are available from John Bell Engineering, P.O. Box 338, Redwood City, California 94064, Model 79-295, $69.95 assembled.) With Slots 3 and 4 taken by the timer-I/O boards and slots used for the disk drive, and perhaps a printer driver, three empty slots, plus Slot 0, are left

Requests for reprints with hard-copy listings of all routines and other inquiries may be directed to the author, Psychology Department, Roosevelt University, 430 S. Michigan Avenue, Chicago, Illinois 60605. The author wishes to thank Carl Jensen, Roger Schnaitter, and David Braught at Illinois Wesleyan University for assistance with early versions of this system. Apple II and II+ are trademarks of Apple Computer, Inc.; AIM-65 is a trademark of Rockwell International, Inc.

0005-7878/82/040409-03$00.55/0

for other expansion (e.g., language card, modem, operating system). Other clocks available from Mountain Hardware and California Computer Systems have been used successfully. However, each requires its own expansion slot, has no I/O lines, and costs more than the combination timer-I/O card. Other parallel cards have been used successfully (e.g., California Computer Systems Model 7720A or B), but these cards typically have no timer and only two 8-bit ports, yielding a total of 16 programmable I/O lines per expansion slot, compared with 32 lines per slot with the card used in the system described here.

The I/O lines are connected, as are the I/O lines on the AIM-65, to an optoisolated 24-V dc interface circuit[1] previously described in detail (Rayfield & Carney, 1981). As with the AIM-65 system, the optoisolation at the external interface card helps to protect the computer from miswiring errors that could do serious damage. The mating of the DIP sockets on the I/O boards in the expansion slots to the optoisolated 24-V dc interface cards is straightforward. The only connection besides the I/O lines and ground is running a wire from 5 V, available in numerous places on the Apple, to power the 7407 integrated circuit and the optoisolators on the interface cards. On the Apple, 5 V is available on the solder pads surrounding each of the Phillips head screws at the rear of the main logic board, but not the screws themselves, or on the forward lead of the capacitor in the left rear corner of the main logic board, or on the trace leading to peripheral expansion connector Pin 25 on either timer-I/O card. Alternately, the necessary 5 V dc can be supplied by an external 5-V dc power supply, sharing the Apple's ground. This is preferred when using more than three or four 24-V dc interface cards, which severely drains the Apple power supply. Each card may draw as much as .15 A.

Ports C300 and C301 (Slot 3) are used as input ports. Output ports are located at memory addresses C380 and C381 in Slot 3 and C400, C401, C480, and C481 in Slot 4. Timer 1, with C30x addressing, in Slot 3 generates continuous interrupts every 10 msec. The timer-I/O card is diagonally jumpered so that the timer interrupts produce nonmaskable interrupts (NMI) on the Apple bus.

MACHINE LANGUAGE ROUTINES FOR TIMING AND INPUT POLLING

It is desirable to place the machine language routines high in memory to leave as much RAM as possible available for BASIC (APPLESOFT) programs and data storage. Because the disk operating system (DOS) resides in high memory, it is necessary to place the machine language programs in RAM just below that. In the 48-KB Apple, the DOS resides above 9600 (hex). Therefore, setting HIMEM=37887, which is equal to 93FF (hex), at the start of the BASIC program leaves adequate RAM for the machine language routines. (A hard-copy listing of the interrupt and initialization routines for the Apple, very similar to those for the AIM-65, is available from the author on request, as is all other machine language and BASIC software described.)

The 6522 timer produces an interrupt every .01 sec. The timing of interrupts is independent of the length of time spent in handling the interrupt. There is a major problem with using software timing loops in BASIC: When events occur that require subroutines, timing is disrupted. This technique avoids that problem. The timing routine is identical to the one used for the AIM-65 except for location. The interrupts are counted, with carry, to keep track of time, and the input polling and passing routines are called. The polling routines, one for each input port (identical to the one shown previously for the AIM-65 except for location), are called each 10 msec. Changes in an input port that persist for more than 2 cycles (20-msec debouncing) are detected, and any individual bit that has changed from a Logic 1 to Logic 0 (switch closure to ground) is extracted with several logic commands and stored with any other inputs waiting for passing to BASIC.

As with the AIM-65, the passing routine checks to see if any inputs are ready to be passed and checks to see if BASIC has picked up the last input. If so, the routine places the identification number of the input in a location for BASIC to PEEK it. BASIC POKEs a zero back out to clear the same location after it has picked up the input identification and then calls a subroutine to handle it.

One difference from the passing routines described previously is that this routine combines the identification numbers for all 16 inputs. Rather than have BASIC check two locations for identification numbers ranging from 1 to 8, BASIC checks only one location for a number from 1 to 16. Inputs to C300 are represented by numbers 1 to 8, and inputs to C301 are represented by numbers 9 to 16. Because operation is transparent to BASIC, it is not critical that the user understand the machine language coding. Combining the two ports helps to shorten the main-line loop in BASIC and makes input identification less ambiguous for the experimenter.

The machine language initialization, called by BASIC in order to start the timer and configure the I/O ports, is similar in form to the one for the AIM-65, but the addresses involved and other details are different. The time counters are reset to zero. The lines that set the data direction registers and access the input and output ports on the 6522 expansion card are similar to those used for the AIM-65. The routine loads the data direction registers C382, C383, C402, C403, C482, and C483 with FF (hex), to set them as outputs, and then loads the output ports themselves (C380, C381, C400, C401, C480, and C481 hex) with FF (hex), to turn the

outputs off. (Logic 1 turns output off; Logic 0—current flows—turns output on.) C30B and C30E (hex) are loaded to select continuous interrupts, and the timer is set by loading the least significant byte into C304 (hex) and the most significant byte into C305 (hex). Thus, a fine and coarse control of the timer are available for adjusting. A larger number slows the timer down; a smaller number speeds it up. Finally, the routine sets the NMI interrupt vector at 03FB-03FD (hex) to call the interrupt routine for counting (timing) and input polling at 9505 (hex). The NMI interrupt is used instead of the interrupt request (IRQ) used in the AIM-65 system. The IRQ interferes with operation of APPLESOFT BASIC.

A BASIC program calls for initialization of the machine language routines, initializes variables for use by BASIC, and polls inputs and keeps track of time in seconds since the session's start. The sample program available on request from the author demonstrates simple counting, output control, and timing functions in BASIC.

OUTPUT CONTROL

In the AIM-65, output control from BASIC is accomplished by generating a mask that is applied with an AND or OR to the contents of the output port in order to change single bits. The AND and OR functions in APPLESOFT, the Apple computer's version of BASIC, do not operate the same as AND and OR in the AIM-65 version of BASIC. They do not return the masked contents, but simply a 1 or 0 for true or false. Therefore, another efficient way of changing individual output bits is needed. Machine language routines use the AND and OR functions in the Apple 6502 microprocessor. The POKE command in BASIC is used to specify which of the 48 outputs (x = 0-47) will be changed (POKE 38084, x) and then simply CALLs either of two machine language subroutines: one that turns the referenced output on (CALL 38112) and another that turns it off (CALL 38128). Detailed understanding of the machine language routines that handle output control is not critical to using the system.

The turn-on and turn-off routines are CALLed by BASIC. Both call a routine that establishes the output port and the particular bit on that port to be affected. This is based on the identification number of the output (0 to 47) POKEd from BASIC. After the port and bit are determined, the turn-on or turn-off routine loads the port and applies an AND or OR mask for the desired change, as with AIM-65 BASIC. The difference is that the Apple routines use masks stored in memory. Application depends on which bit is to be controlled, rather than on generating the masks mathematically as with AIM BASIC. The masked port contents are then sent back to the port, producing the desired change in one output and leaving the others unaffected.

This output technique is very convenient and requires less BASIC programming than the AIM-65 system. The user needs only a list of the various outputs, their associated identification numbers, and the two CALL commands (turn on, CALL 38112; turn off, CALL 38128). Thus, a typical line to turn an output on might read: 100 POKE OUT, 3: CALL OP, where OUT has been set equal to 38084 and OP (for operate) has been set equal to 38112.

APPLICATIONS

The system can be used for a wide variety of laboratory applications. Decisions are made at each input or at any time (in seconds since session start) based on counts, time, or any mathematical or logical decision possible in APPLESOFT BASIC. The system has been used to run a variety of operant schedules simultaneously in different chambers. A typical operant experiment requires approximately 1 KB of BASIC per input. Thus, memory limitations in a 48-KB Apple system are unlikely. The system has also been converted to average evoked potentials and to produce a graphic display on the Apple monitor. This requires additional A/D conversion hardware, which is read and controlled from BASIC.

If more than 16 inputs and 48 outputs are required, the software components can be iterated and timer-I/O cards (with the timers not used) can be added until the Apple's available slots are filled. The hardware described here can be used as 64 inputs or 64 outputs.

Finally, BASIC is slow. The system described here handles 10 responses/sec in operant research. The usual suggestions that constants be made variables, that remarks lines be omitted, and that often used subroutines be placed first in the program apply to APPLESOFT as they do to other implementations of BASIC. Several compilers will compile Apple's BASIC (e.g., Microsoft's TASC, about $180, and On-Line's Expediter, about $100). Once compiled, a BASIC program runs 10-20 times faster than a noncompiled (interpreted) program. A compiled BASIC program handles 100-200 responses/sec. In that case, a user may wish to change the system timing for 1-msec interrupts to obtain finer grained timing. Stimulus presentations with a second display monitor are easily accomplished with a Y connector in the video output.

REFERENCE

RAYFIELD, F., & CARNEY, J. Controlling experiments with BASIC on 6502-based microcomputers. *Behavioral Research Methods & Instrumentation*, 1981, 13, 735-740.

NOTE

1. The optoisolated 24-V dc interface circuit is available on a circuit card with 8 inputs and 6 outputs, 12 outputs, or 16 inputs from the author, 5300 S. Shore Drive, No. 97, Chicago, Illinois 60615. Price is approximately $100.

(Received for publication February 18, 1982; revision accepted April 22, 1982.)