Behavior Research Methods & Instrumentation
1978, Vol. 10 (6), 779-784

A psychological laboratory based on a PDP-11/10 with graphics

WILLIAM PRINZMETAL
Claremont Graduate School, Claremont, California 91711

and

WILLIAM P. BANKS
Pomona College, Claremont, California 91711

This article describes a laboratory computer system that uses a PDP-11/10 CPU and a graphics processor to simulate a multifield tachistoscope and to run a great number of psychological experiments. The potential of the hardware is discussed, and two useful programs are presented.

The system described here was purchased primarily for research in visual information processing and so had to have a very flexible cathode-ray tube (CRT) display. Since the computer had also to control other equipment for this research, we felt it was reasonable to try to build a system that was general enough to run the full spectrum of experimental paradigms used in human experimental psychology in addition to the specific experiments for which it was purchased. The tasks required of the system included simulation of a tachistoscope, timing of external events and subjects' responses to the nearest .1 msec, and interfacing with other laboratory equipment. It was also desirable that the system be able to interpret and generate analog signals. We also wished to have a system that would not involve us in complex software development every time we started a new experiment. The system described here was therefore designed to meet two "benchmark" criteria: flexibility in operating experimental apparatus and minimal requirements for software development. To satisfy both criteria, the system had to be complete in that all the hardware elements we required were an integral part of the system and, equally important, the the hardware components were well supported by software. The system met both criteria, and we were able to conduct our first experiment (which simulated a N-field tachistoscope, timed reaction times to the nearest 1 msec, and used custom-designed graphics characters) only 3 weeks after the system was installed.

The description of our system that follows is divided into hardware and software sections. The reader should keep in mind that the success of the system results largely from the fact that most of the hardware came fully supported with software. We cannot stress too strongly that when a system is being planned, hardware and software should be considered together. It must always be remembered that a piece of hardware, no matter how sophisticated, will not do all it can do unless it is programmed to do it, and programming can often be a major project in itself.[1]

The system described in this article was purchased with funds from National Science Foundation Research Grant BMS 75-20328 to William P. Banks. Preparation of this article was supported by NSF Grants BMS 75-20328 and BNS 78-17442. Requests for reprints should be sent to either author, Psychology Department, Pomona College, Claremont, California 91711.

HARDWARE

Our system is built around a Digital Equipment Corporation (DEC) PDP-11/10 computer. Figure 1 shows the hardware components of the system. The unibus is a common communication path through which the CPU can address any of the hardware elements. The unibus allows the CPU, the graphics processor, and the DECwriter to communicate directly with the other components, as well as with each other.

The AR-11 laboratory interface system is the essential device that allows the computer to collect data and control events in experiments. The AR-11 has a programmable real-time clock, a 16-channel analog-to-digital (A/D) converter, a three-channel digital-to-analog (D/A) converter, and three digital outputs. The AR-11 interface can be controlled by FORTRAN- or BASIC-callable subroutines supplied by DEC or by user-written subroutines (see below). We have used the programmable clock to time presentation of events such as displays on the graphics system, to measure subjects' response latencies, and to control sampling from the A/D channels. We have used the A/D converters to digitalize audio signals that subsequently go through a transformation in the computer and are then output through a D/A converter and into an audio amplifier. We have also used the A/D channels as though they were digital input channels. A subject's keypressing response applies a voltage to an analog channel, and the time at which the

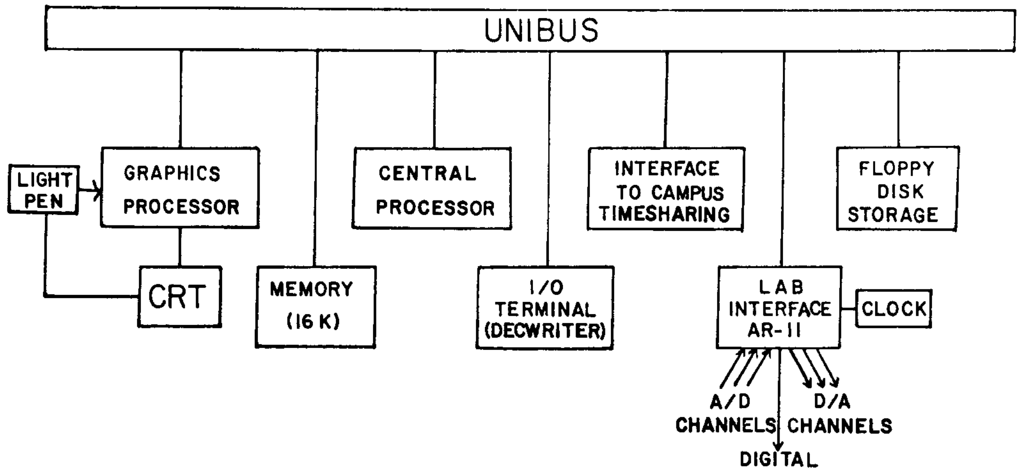

Figure 1. Hardware configuration of Lab PDP-11/10 with graphics.

keypress occurred can be determined to the nearest 1/20 msec. This arrangement allowed us to avoid buying a separate digital input device to collect discrete responses. (Digital responses can also be collected with the DECwriter terminal.) The digital output channels are used to operate relays that control tape recorders, lights, buzzers, bells, whistles, and so on.

The VT-11 graphics system includes a display processor, CRT, and light pen. The CRT display is generated by painting individual points and lines on the screen. The brightness of the display is in part determined by the rate at which points are refreshed. The faster the refresh rate, the brighter the display. The system has two refresh rate options. By the first option, the refresh rate is determined by the amount of material being displayed, so that larger displays will be refreshed at a slower rate than smaller displays. This option is important because it allows displays of any size to be refreshed at the fastest rate possible, and the size of the display is not limited by the refresh rate. By the second option, the refresh rate is set to a constant 60 Hz. This option eliminates variations in brightness of displays with differing sizes. Hence the system is considerably more flexible than a raster-scan or Tektronix-type storage scope. It also has a better brightness contrast ratio than either of these systems.

The graphics system is not a "point plotter," but rather it is a sophisticated microprocessor that uses core memory as a display buffer. The display processor has hardware alphanumeric characters, line, and point generators. The hardware characters and user-programmed software characters and pictures can all be displayed simultaneously. The graphic processor can be controlled by FORTRAN- or BASIC-callable subroutines supplied by DEC. User-designed characters are called up as subroutines and are plotted at about 20 microsec/stroke, with an alphabetic character requiring from two to five strokes. Each of the 128 hardware characters requires 20 microsec to be plotted. Most tachistoscopic stimuli, including simple line drawings, can be plotted in less than 1 msec. The CRT can be used with the DECwriter keyboard as a video terminal, independently of the keyboard as a graphics system, or as both simultaneously. The light pen is an input device. A subject or experimenter can interact directly with the display with the light pen and can, for example, "draw" pictures on the CRT that can later be called up and displayed. Material at any point on the screen can be made light-pen sensitive, and the sorts of complex interactions a subject (or experimenter) can have with the array are limited only by the ingenuity of the programmer.

The CRT was supplied with a P39 phosphor tube. The fading trace produced by this phosphor requires 150 msec to decay to 10% of original brightness and is thus too persistent for tachistoscopic presentation. However, a long-persistence phosphor avoids flicker when text is presented and so is desirable when the CRT is used for software development. In order to have the benefits of both long- and short-persistence phosphor display tubes, we replaced the original tube with a dual phosphor tube we obtained from an independent source (Thomas Electronics, 100 Riverside Drive, Wayne, New Jersey 07470, or 12700 South Main Street, Los Angeles, California 90061). The replacement tube has a dual P40 phosphor that has a short-persistence blue phosphor that decays to 10% brightness in 150 microsec and a long-persistence yellow-green phosphor that decays to 10% brightness in approximately .5 sec. When the CRT screen is viewed through blue filters (Kodak Wratten Filter 47), only the short-persistence blue phosphor can be seen.[2] When conducting experiments where the exposure duration must be short and/or precisely timed, we have subjects view the display through the blue filters. We have built a view tunnel that is attached to the CRT (see Figure 2). During experiments, the tunnel slides out to the appropriate distance and a viewer with the blue filters folds down into place. At other times, such as when using the CRT as a video terminal for program development, the viewer is folded up and the view tunnel is slid back. When the CRT screen is viewed without filters, the light from the two phosphors blends

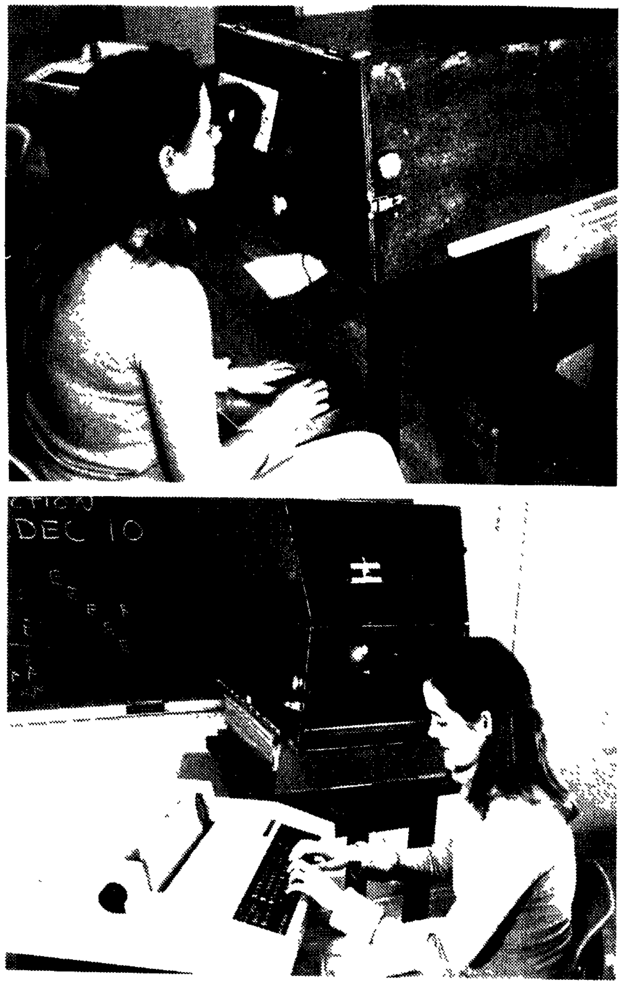

Figure 2. (a) The CRT is viewed through the view tunnel with filters while conducting a tachistoscopic experiment. (b) The view tunnel is not used when using the CRT as a video terminal.

to make a white light that flickers very little even when the screen is full of text.

As data is collected in an experiment, it is temporarily stored on a floppy disk. At the completion of an experiment, the data is sent through the extra serial interface to a large campus-wide timesharing DEC 10 computer system for analysis. In addition to saving time and labor costs transferring data to the large computer, the interface makes it possible for us to use peripheral devices on the large computer, such as a card reader/punch, magnetic tape, and so on.

SOFTWARE

Our system software includes the RT-11 operating system, BASIC, FORTRAN IV, and several special-function FORTRAN libraries. RT-11 is a fast-operating single-user system. RT-11 includes both single-user and foreground/background monitors. Because the foreground/background monitor requires more memory than the single-job monitor, leaving less memory for user programs, we have mainly used the single-job monitor.

Most of our programs are written in FORTRAN. The FORTRAN is supported by several special-purpose libraries of subroutines supplied by DEC that make it very powerful for controlling experiments. The SYSLIB library includes over 100 subroutines that provide access to all features of the RT-11 monitor, making assembly language programming unnecessary for most uses. Some of these functions include accessing the contents of particular memory locations, changing system input-output parameters, controlling and communicating information between foreground and background jobs, and providing character string handling capabilities. The SYSLIB subroutines provide the FORTRAN programmer complete control of the RT-11 monitor. These subroutines enabled us, for example, to write a program that controls communication between our PDP-11 computer and the campus timersharing system completely in FORTRAN. (This program is available from the authors on request.)

The GTLIB library includes FORTRAN-callable subroutines that control the graphics system. Pictures consisting of lines, dots, and/or alphanumeric characters can be created, displayed on the CRT, and stored on the floppy disk for later use. Libraries of pictures or text can be created in this manner. Pictures can be copied in different locations on the screen or moved to create dynamic stimuli ("cartoons"). The coordinates of the light pen on the screen can be determined by putting a "tracking object" on the screen that always centers itself on the light-pen location. The tracking object can be moved around the screen to, for instance, draw pictures. Also, pictures or text on the screen can be made light-pen sensitive, so that the computer registers a hit when the picture or text string is touched with the light pen. Subjects can thus use the light pen to indicate a response. The graphics display is extremely easy to control with the FORTRAN GTLIB subroutines.

The LPSLIB library of FORTRAN-callable subroutines controls the AR-11 laboratory system, including timing of events with the five-frequency crystal clock, A/D input conversions, and D/A output conversions. The LPSLIB subroutines are somewhat more complex than those in the other subroutine libraries. We did experience some difficulty in developing a method of timing external events with the programmable clock. Appendix A contains a sample tachistoscope program as well as a subroutine we developed for collecting discrete response reaction times in milliseconds through analog channels by using both LPSLIB and SYSLIB subroutines.

The BASIC interpreter is also supported with graphics and AR-11 functions that are almost identical to the FORTRAN libraries. However, because of the longer execution time and larger storage requirements of BASIC programs, programs to control experiments are generally written in FORTRAN. Since the graphics and AR-11 functions are almost identical in FORTRAN and BASIC, it is convenient to use BASIC to learn how to use these functions and to develop program ideas. In addition, a display file (i.e., a picture) can be drawn by a program in BASIC and later used by one in FORTRAN (or vice versa). Appendix B contains a sample BASIC program that allows one to draw pictures with the light pen. These pictures can then be stored and later called up in either a BASIC or a FORTRAN program.

Only a few of our programs have had to be written in assembly language (MACRO). These programs sample A/D input channels and/or output D/A channels at extremely fast rates to process and/or create auditory signals. Good-fidelity sound signals are created by sampling and outputting at the rate of once every 30 microsec.

In conclusion, the system is able to control nearly all experiments that might be required of a psychological laboratory. Most of these functions have been implemented quickly and easily without sophisticated user hardware or software development. We recommend the graphics system described here as an alternative to a point plotter for computer simulation of a tachistoscope. It is true that a point plotter (i.e., a computer-controlled D/A converter driving an X-Y-Z CRT display) is capable of creating many of the displays the graphics can generate, but the point plotter requires a great deal more programming skill. The point plotter is also slower than the graphics system, particularly in drawing lines. Finally, accurate real-time control is, in most cases, easier to achieve for the graphics displays than for point plotters.

NOTES

1. When shopping for our system, we found it useful not only to ask salespeople what functions a particular computer could possibly perform (hardware), but also to ask them to give us specific information on how those functions could be implemented (software). We also asked for (and usually received) names of researchers who were using the hardware in question for purposes similar to the ones we planned. Discussions with these researchers were invariably informative. Some pointed out severe limitations in their systems even though they were able to perform the assigned function. All had good advice on alternate hardware configurations, sources of software, and inexpensive ways to solve various problems.

2. We also tried a P7 phosphor, which is similar to the P40 phosphor. However, we had difficulty completely filtering out the yellow-green trace of the P7 phosphor.

Appendix A

```
C
C                 SAMPLE TSCOPE PROGRAM
C
          DIMENSION IBUF(500),IORDER(100)
          CALL INIT (IBUF,500)                     !INITIALIZE DISPLAY BUFFER.
          TYPE 100
          ACCEPT 101,NTRIAL                        !INPUT NUMBER OF TRIALS.
          CALL RANDOM(IORDER,NTRIAL)               !GET PERMUTATION OF NTRIALS.
          CALL RSTR('STIM.DAT')                    !READ DISPLAY FILE.
          DO 1 I=1,NTRIALS
          CALL ON(IORDER(I))                       !TURN ON STIMULUS # IORDER(I).
          CALL INPUT(IANS,RTIME)                   !WAIT FOR RESPONSE.
          CALL OFF(IORDER(I))                      !TURN OFF STIMULUS.
          WRITE(7,102)IORDER(I),IANS,RTIME
1         CALL ZZZ(1000)              !WAIT 1 SEC.
100       FORMAT(' ENTER NUMBER OF TRIALS.')
101       FORMAT(I3)
102       FORMAT(I4,5X,I1,5X,F6.0)
          CALL EXIT
          END


                 A SUBROUTINE TO COLLECT REACTION TIMES (SEE TEXT)



          SUBROUTINE INPUT(ITI,RTIME)
C
C         SUBROUTINE TO WAIT FOR INPUT FROM A/D CHANNELS 1 OR 2
C         RETURNS CHANNEL NUMBER AND ELAPSED TIME IN MS
```

Appendix A (Continued)

```
C
        EXTERNAL QSEC             !CLOCK COMPLETION ROUTINE
        COMMON/CLOCK/IQSEC                  !NUMBER OF QUARTER-SECONDS
        PRESET=250.0
        IQSEC=0
        ICMF=0
        CALL SETR(4,1,PRESET,ICMF,1,QSEC)
10      CALL IADC(1,4,ICH1)
        CALL IADC(2,4,ICH2)
        IF (ICH1.GT.12600 .OR. ICH2.GT.12600) CALL SETR(-2,,,)
        IF (ICMF.EQ.0) GOTO 10
        ITIME=IPEEK("170416)                !GET NUMBER OF MILLISECONDS
        RTIME=(ITIME-7)+250.0*IQSEC         !ADD IN QUARTER SECONDS
        ITF=2                     !ASSUME FALSE
        IF (ICH1.GT.12600) ITF=1            !CHANGE IF TRUE
        RETURN
        END


        SUBROUTINE QSEC
C
C       CLOCK COMPLETION SUBROUTINE -- COUNT QUARTER-SECONDS
C
        COMMON/CLOCK/ IQSEC
        IQSEC=IQSEC+1
        RETURN
        END
```

Appendix B

```
   .
   .
   2 REM TRAK.BAS DRAWS PICTURES WITH
   3 REM THE LIGHT PEN.
   4 REM FOR DOCUMENTATION, SEE "BTRAK.DOC"
   5 REM THIS IS VERSION 6
   6 REM IF YOU HAVE QUESTIONS OR COMMENTS
   7 REM PLEASE CONSULT THE AUTHOR,
   8 REM CHARLES PETERSON, PSYCHOLOGY DEPT.,
   9 REM POMONA COLLEGE, CLAREMONT, CA 91711
  10 L=-1\F=-1\T=1\I=5
  15 Q=0
  21 PRINT "THIS PROGRAM DRAWS LINES WITH THE LIGHT PEN"
  22 PRINT "LIGHT PEN MOVES TRACKING OBJECT"
  23 PRINT \PRINT "INPUT  EFFECT"\PRINT "VALUE"\PRINT
  24 PRINT "P     DRAWS A POINT"
  25 PRINT "L     DRAWS A LINE"
  26 PRINT "<CR>  REPEATS PREVIOUS L OR P COMMAND"
  27 PRINT "U     LIFTS LINE"
  28 PRINT "S     SAVES DISPLAY"
  29 PRINT "E     ERASES TRAKING OBJECT"
  30 PRINT "C     CLEARS AND RESETS DISPLAY"
  31 PRINT "IN    CHANGES INTENSITY(N:1-8)"
  32 PRINT "ON    CHANGES PEN INTERRUPT STATUS(N:1=ON,-1=OFF)"
  33 PRINT "FN    CHANGES FLASH MODE(N:1=ON,-1=OFF)"
  34 PRINT "TN    CHANGES LINE TYPE(N:1=SOLID;2=L,3=S,4=D DASHED)"
  35 PRINT "D     REPEATS THESE INSTRUCTIONS"
```

Appendix B (Continued)

```
36 PRINT "X      EXITS OUT OF PROGRAM"
37 PRINT "R      RESTORES A SAVED DISPLAY"
38 PRINT "AT     INSERTS AN ALPHAMERIC STRING (T)"
50 IF Q=1 THEN 130
100 X=500\Y=500\N=0
110 CALL "INIT"
120 CALL "TRAK"(X,Y)
130 INPUT #0:N$
131 Q=1
132 T$=R$
135 IF LEN(N$)=0 THEN 140 \T$=SEG$(N$,1,1)
140 IF T$="X" THEN 500
141 IF T$="I" THEN 400
142 IF T$="F" THEN 420
143 IF T$="O" THEN 440
144 IF T$="T" THEN 460
145 IF T$="P" THEN 240
150 IF T$="L" THEN 200
151 IF T$="D" THEN 23
154 IF T$="U" THEN 480
155 IF T$="C" THEN 100
156 IF T$="S" THEN 300
157 IF T$="R" THEN 300
158 IF T$="A" THEN 360
160 IF T$<>"E" THEN 190 \CALL "ERAS"
190 GO TO 130
200 REM DRAW LINE HERE
205 R$="L"
210 IF N=0 THEN 275
220 X2=X\Y2=Y
225 CALL "VECT"(X2-X1,Y2-Y1,L,I,F,T)
230 X1=X2\Y1=Y2
235 GO TO 130
240 N=0
245 R$="P"

250 X1=X\Y1=Y
260 CALL "APNT"(X1,Y1,L,I,F,T)
270 GO TO 130
275 X1=X\Y1=Y
278 N=1
280 CALL "APNT"(X1,Y1,L,-1,F,T)
285 GO TO 130
300 PRINT "INPUT DESIRED FILE NAME"
310 PRINT "(DEV:)FILNAM(.EXT)"
320 INPUT F$
325 IF T$="S" THEN 330
326 CALL "RSTR"(F$)
327 GO TO 130
330 CALL "DSAV"(F$)
350 GO TO 130
360 S2=LEN(N$)
361 A$=SEG$(N$,2,S2)
362 X1=X\Y1=Y
363 CALL "APNT"(X1,Y1,L,-I)
365 CALL "TEXT"(A$)
366 GO TO 130
400 I=VAL(SEG$(N$,2,6))
415 GO TO 130
420 F=VAL(SEG$(N$,2,6))
430 GO TO 130
440 L=VAL(SEG$(N$,2,6))
450 GO TO 130
460 T=VAL(SEG$(N$,2,6))
470 GO TO 130
480 N=0
485 GO TO 130
500 CALL "INIT"
1000 END
```