Behavior Research Methods, Instruments, & Computers
1987, 19 (4), 383-388

IBM PC Tachistoscope: Text stimuli

SIDNEY J. SEGALOWITZ
Brock University, St. Catharines, Ontario, Canada

Due to cost, speed, and flexibility, the IBM PC and compatibles are good computers for laboratory use. Outlined here is a program that allows a computer from this class of machines to be used as a tachistoscope. The example given is for visual half-field presentation and can be easily modified for other presentation paradigms.

We have been using the IBM PC microcomputer and compatibles in a cognitive-perception laboratory for tachistoscopic presentation of text materials (see Diener & Smee, 1984, for an Apple IIe tachistoscope program), and we have found that programming in BASIC is adequate for standard tachistoscopic presentation. The ease of using interpreted BASIC to try out the details of stimulus presentation makes this mode very useful indeed. A compiled version of the program is used for data collection. In the program presented here (Listing 1), only the routine to read the millisecond clock is programmed in assembly language. This language is needed if the experiment demands millisecond-accuracy reaction times and if the Tecmar clock[1] is used. An alternative loop counter is also provided with slightly better than millisecond accuracy. The disadvantages of *not* using a real-time clock are described.

THE PROGRAM STRUCTURE

The program consists of a main program section and six GOSUB subroutines. Comments are included to help devise variations to suit specific experimental requirements. The example given in Listing 1 is for a visual half-field (VHF) study, where single words are presented on each trial. Minor reprogramming is needed to allow for double stimuli or other changes. The large-print format (40-character screen width) allows for a maximum of eight pages of text (i.e., eight viewing fields). In the program, the first page (SCREEN 0) is used for the fixation point, SCREEN 1 for the stimulus, SCREEN 2 for a postexposure blank field (onto which a mask can be placed), and SCREEN 3 for the end-of-experiment message. These can, of course, be varied.

Stimulus Input

For the configuration in Listing 1, the stimulus input file (STIMFIL$ requested in line 100) should consist of stimulus word items separated by some delimeter, such as a comma, space, or carriage return. In general, however, a single stimulus can be up to a screenful of text (40 characters × 25 lines). It is up to the user to place the stimulus appropriately on the display (lines 360-420 in Listing 1). The program in Listing 1 places a word stimulus to the right or left of the fixation point, and accordingly each stimulus line in the input file contains a "1" or "2" (to indicate whether the LVF or RVF is to be used on that trial) and a word stimulus (e.g., "1 tree"). The stimuli are read in at line 130.

Screen Persistence

One problem of microcomputer presentation is that CRT screens have various persistence times, making accurate timing of stimulus presentation difficult. This problem is easily circumvented by using the reverse video mode: the screen remains on except for those pixels that correspond to the stimulus. Thus, when the stimulus is removed, lights are turned back on, removing any chance of screen persistence. Using a backward mask also reduces the screen persistence problem, especially when displaying in normal video.

Response Output File

If a verbal response is to be scored for content, clearly no reaction time (RT) information is needed, and therefore no output file is required (delete lines 140 and 560-600). If the response is a keyboard keypress (e.g., same/different, yes/no, lexical decision, etc.), then the output file (FOUT$) storing the data can contain several pieces of information: stimulus number, RT, the ASCII code of the key pressed, the designated VHF, and the stimulus word itself.

Sync Pulse

It is important with CRT presentation to time the stimulus onset to coincide with the return of the raster for a new screen scan, so that the RT can be linked consistently to the stimulus onset. The IBM color graphics card refreshes the screen 60 times per second. Of the 16.7 msec required for each screen refresh, approximately 1 msec is taken up by the raster returning to the top of the screen. It is during this period that the active screen page is switched (line 460) to the one with the stimulus and the clock timer is reset. The sync pulse, registered in bit 3 of PORT 986, is high when information is being sent to the screen.

This research was supported by a grant from the Natural Science and Engineering Research Council of Canada. Please address correspondence to the author at: Department of Psychology, Brock University, St. Catharines, Ontario L2S 3A1, Canada.

Millisecond Clock Timer

Tecmar provides a millisecond clock timer on many of its multifunction cards with a BASIC routine to read the clock times. With this routine, there may be some millisecond inaccuracy during the reading process because of rollover, where a millisecond value of 9 is read and then augmented, forcing a carry in the hundredths, before the rest of the time is read. The assembly language subroutine CLOCKRD.ASM (Listing 2) reads the clock in approximately 0.029 msec and, if the time has changed during the reading process, the system reads it again. The result in this procedure is that the RT is rounded up when it is within 29 μsec of the next millisecond.

If a real-time clock is unavailable, the user can substitute for one with a looping subroutine. For example, to time how long it takes for the subject to press a key, substitute Listing 3 for lines 650-670 in Listing 1. Note that line 655 requires a maximum reaction time (TIMEOUT) defined earlier in the program. The formula in line 670 converts the units to milliseconds, but the raw score LOOP can, of course, be used for higher resolution. Similarly, Listing 4 should substitute for lines 820-850 to provide the wait subroutine. The formulas converting the loops to milliseconds are derived from compiling the code using the IBM PC BASIC Compiler, Version 1.00.

Please note an important limitation to not using a separate real-time clock. The timing of the subject's response cannot begin until the stimulus is off, assuming the experimenter wants the stimulus exposed for a brief period. In other words, the timing cannot start until all screen handling is done. Since VHF presentation must be for less than 200 msec to be effective and since RT is normally greater than 400 msec, this poses no problem in the paradigm presented. However, this may not always be the case, and purchase of a real-time clock with millisecond reporting should be considered.

Stimulus Duration

The stimulus can be left on for times in multiples of 16.7 msec, referred to in the program as NSCANS, read in by the user (line 90). The intertrial interval is set in lines 525-526. The fixation-point duration is set in lines 320-340.

Fixation Point

The fixation point is currently set to appear with the stimulus. To disable it, remove line 380. The postexposure field is currently blank. To maintain a fixation point continuously, remove the apostrophe in line 220.

Running the Program

Since the stimuli and results are stored in RAM until the session is finished, a RAM disk is not needed. The DIMension statement (line 70) allows up to 256 stimuli. This can be increased as needed.

The program can be halted by pressing the space bar between trials and then continued by hitting any key. Pressing "R" between trials restarts the program from the beginning. Pressing "S" between trials stops the program and stores the data gathered thus far. Pressing "Q" between trials stops the program without saving the data.

HARDWARE REQUIREMENTS

To run the program, one needs an IBM PC or compatible computer with IBM color graphics card, one disk drive, and 96K of RAM.

SOFTWARE REQUIREMENTS

The following are needed: BASICA and BASIC compiler, MacroAssembler (optional), and Tecmar clock (optional).

REFERENCE

DIENER, D., & SMEE, W.P. (1984). Apple tachistoscope. *Behavior Research Methods, Instruments, & Computers*, **16**, 540-544.

NOTE

1. Tecmar, Inc., 6225 Cochran Rd., Cleveland, OH 44139. I have not been able to find another real-time clock for the IBM PC that permits millisecond access.

LISTING 1
The BASIC Main Program

```
10 'VHF.BAS   --  VHF.BAS using raster checks. Single stimuli.
20 '            by S. J. SEGALOWITZ
30 '               DEPT. OF PSYCHOLOGY
40 '               BROCK UNIVERSITY
50 '               ST. CATHARINES, ONTARIO L2S 3A1
60 '
70 DEFINT A,T:DIM A$(256),S(256),RESP(256),RT(256)
80 SCREEN 0
90 INPUT "Number of scans (@ 16.7 ms. each) ...",NSCANS
100 INPUT "Stimulus file ...",STIMFIL$
110 OPEN "i",#1,STIMFIL$:I=0
120 IF EOF(1) THEN 140
```

LISTING 1 (Continued)

```
125 '                       S is the side of presentation (LVF/RVF)
126 '                       A$ is the list of stimulus words
130 I=I+1:INPUT #1,S(I),A$(I): GOTO 120                    'read in stimuli
140 CLOSE(1):INPUT "Name of output file ...",FOUT$:OPEN "o",#2,FOUT$
150 NSTIM = I
160 KEY OFF:WIDTH 40:COLOR 0,7,7                           'reverse video
170 DELAY =100:CLS:LOCATE 10,10,0
180 PRINT "Press space bar when ready":GOSUB 900           'ready to go?
190    OUT 893,21:OUT 895,1                              'set timer to 0.0 seconds
200    CLS:WATE = 2000: GOSUB 820                          'wait 2 seconds
210 LOCATE 11,20,0:SCREEN ,,0,0:PRINT CHR$(254)            'fixation point
220 SCREEN ,,2,0:CLS ':LOCATE 11,20,0:PRINT CHR$(254)      'fix pt on screen 2
230 '
240 '                                    *********** start ***********
250 FOR J=1 TO NSTIM
260 A$=INKEY$                                              'stop program
270    IF A$="q" OR A$="Q" THEN END
280    IF A$=CHR$(32) THEN GOSUB 870:GOTO 260              'halt and continue
290    IF A$="R" OR A$="r" THEN 250                        'restart
300    IF A$="S" OR A$="s" THEN 540
310 SCREEN ,,0,0                                           'fixation point
320                                                      'intertrial interval
330 OUT 893,21:OUT 895,1                                 'set timer to 0.0 seconds
340 WATE = 1000: GOSUB 820                                  'wait 1 second
350 '
360 '                              write stimulus word to screen 1
370 SCREEN ,,1,0:CLS
380 LOCATE 11,20,0:PRINT CHR$(254)              'fixation point (optional)
390 LG = LEN(A$(J))
400 IF S(J) = 1 THEN 410 ELSE 420
410       LOCATE 11,19-LG:PRINT A$(J):GOTO 430          'LVF
420       LOCATE 11,22:PRINT A$(J)                      'RVF
430 '
440 DEF SEG=0: POKE 1050, PEEK(1052)          'CLEAR KEYBOARD BUFFER
450 GOSUB 780                               'wait for raster off
460 SCREEN ,,0,1                            'flip on stimulus word - < 1.2 ms.
470 OUT 893,21:OUT 895,1                    'SET TIMER TO 0.000 SECONDS
480 GOSUB 680                               'count # of scan cycles
490 SCREEN ,,0,2                            'stimulus off
500 GOSUB 650                               'get RT and response
510 RT(J)=TIM                               'store RT
520 RESP(J)=ASC(A$)                         'store response
530 NEXT J                                  'next stimulus
540 SCREEN ,,3,3                            'write data to new screen
550 '
560     LOCATE 10,10:PRINT "Storing data"
570     PRINT #2,"Stimulus file is ";FIN$,"# of screen refreshes/stim =";NSCANS
580     PRINT #2,"Stim #        RT(ms.)      RESPONSE "
590     FOR J=1 TO NSTIM
600     PRINT#2,J,RT(J),RESP(J),S(J),A$(J):NEXT            'write to disk
610 '
620 LOCATE 10,10: PRINT "Run program another time? (Y/N) ..."
630 GOSUB 870: IF A$ = "Y" OR A$="y" THEN CLOSE(2): GOTO 100
640 SCREEN ,,0,0:WIDTH 80:END              '************  END ***********
650 'get clock reading
660 A$=INKEY$: IF A$="" THEN 650
670 CALL CLOCKRD(TIM):RETURN
680 'count the scans
690 NUMSCAN = 0
700 IF NUMSCAN = NSCANS THEN 750
710 A=INP(986) AND 8:IF A=0 THEN  710      'wait while rastor is off
720 A=INP(986) AND 8:IF A=8 THEN  720      'wait till rastor finishes scan
730 NUMSCAN = NUMSCAN + 1                  'count # of raster returns
740 GOTO 700
750 SCREEN ,,0,2
```

LISTING 1 (Continued)

```
760 RETURN
770 '
780 '                                          wait for raster return (=0)
790 A=INP(986) AND 8:IF A=8 THEN 790
800     RETURN
810 '
820 'wait for WATE milliseconds
830 GOSUB 670:T1 = TIM
840 GOSUB 670: IF TIM-T1 < WATE THEN 840
850 RETURN
860 '
870 A$=INKEY$: IF A$="" THEN 870              'wait for keypressed
880 RETURN
890 '
900 A$=INKEY$: IF A$="" THEN 900              'wait for cue
910 IF A$=CHR$(32) THEN RETURN ELSE 900
```

LISTING 2
Assembly Language Subroutine to Read the Time in Milliseconds from the Tecmar Clock

```
TITLE   CLOCKRD.ASM   --  reads the time on the TECMAR clock
;       The CALL from BASIC takes 212.2 microseconds.
;       To read the time takes 29.09 microseconds.
;
;       by S. J. SEGALOWITZ
;          DEPT. OF PSYCHOLOGY
;          BROCK UNIVERSITY
;          ST. CATHARINES, ONTARIO L2S 3A1


SSEG    SEGMENT STACK
        DW      80H DUP(?)
SSEG    ENDS

CSEG    SEGMENT
        ASSUME CS:CSEG,SS:SSEG

        PUBLIC  CLOCKRD
CLOCKRD PROC FAR

        PUSH    BP
        MOV     BP,SP
        PUSH    AX
        PUSH    BX
        PUSH    CX
        PUSH    DX

MUL_10  MACRO                   ;MACRO to multiply AX by 10
        SAL     AX,1            ;multiply by 2
        MOV     SI,AX           ;add to SI
        SAL     AX,1            ;multiply by 2 = 4
        SAL     AX,1            ;again = 8
        ADD     SI,AX           ;8 times + 2 times = 10 times
        MOV     AX,SI           ;put result back in AX
        ENDM

        MOV     DX,893          ;clear the rollover bit
        MOV     AL,20           ;  takes 28 clock cycles
        OUT     DX,AL           ;  = 5.88 microseconds
        MOV     DX,895
        IN      AL,DX

READ:   MOV     DX,893          ;collect seconds
        MOV     AL,2            ; Read time = 87 clock cycles
```

LISTING 2 (Continued)

```
        OUT     DX,AL           ;           = 18.17 microseconds
        MOV     DX,895
        IN      AL,DX
        MOV     CL,AL           ;save it in CL

        MOV     DX,893          ;collect 10ths and 100ths
        MOV     AL,1
        OUT     DX,AL
        MOV     DX,895
        IN      AL,DX
        MOV     BH,AL           ;save it in BH

        MOV     DX,893          ;collect 1000ths
        SUB     AL,AL
        OUT     DX,AL
        MOV     DX,895
        IN      AL,DX
        MOV     BL,AL           ;save it in BL

        MOV     DX,893          ;check the rollover bit
        MOV     AL,20           ; takes 24 clock cycles till
        MOV     DX,895          ; CMP is done = 5.04 microsec.
        IN      AL,DX
        CMP     AL,1            ;has a change occurred?
        JE      READ            ;read the time again if yes

; time is now collected.

        SUB     AH,AH           ;clear AH
        MOV     AL,CL           ;seconds in CL
        MUL_10
        MOV     CX,AX           ;store in CX

        MOV     AX,00F0H        ;high nibble for 10ths
        AND     AL,BH
        SAR     AX,1            ;put it in low nibble of AL
        SAR     AX,1
        SAR     AX,1
        SAR     AX,1
        ADD     AX,CX           ;add in seconds
        MUL_10
        MOV     CX,AX           ;save total in CX

        MOV     AX,000FH        ;low nibble for 100ths
        AND     AL,BH
        ADD     AX,CX
        MUL_10

        SUB     BH,BH           ;clear BH
        SAR     BX,1            ;move 1000ths into low nibble
        SAR     BX,1
        SAR     BX,1
        SAR     BX,1
        ADD     AX,BX           ;save total



        MOV     SI,[BP]+6       ;addr TIM
        MOV     [SI],AX

        POP     DX
        POP     CX
        POP     BX
        POP     AX
```

LISTING 2 (Continued)

```
        POP     BP
        RET 2

CLOCKRD ENDP

CSEG    ENDS
        END
```

LISTING 3
Reaction-Time Loop to be Used if No Real-Time Millisecond Clock is Available

```
650 'RT timing loop
651 LOOP% = 0
652 A$=INKEY$: IF A$="" THEN 655 ELSE 670
655 LOOP% = LOOP% + 1: IF LOOP%>TIMEOUT THEN 670
660 GOTO 652
670 TIM = INT(1.750946 + .8174851*LOOP%): RETURN
```

Note—Accuracy is within 1 msec with times 100 to 2,000 msec.

LISTING 4
Timing Loop for Waiting a Specified Time if No Real-Time Clock is Available

```
820  wait for WATE milliseconds
830  DELAY% = (WATE - .1089954) *11.39136
840  FOR LOOP% = 1 TO DELAY% : FOR WASTE% = 1 to 2 : NEXT : NEXT
850  RETURN
```

Note—Accuracy is better than within 1% with times 100 to 2,000 msec.

(Manuscript received September 12, 1986;
revision accepted for publication January 30, 1987.)