COMPUTER TECHNOLOGY

Machine language millisecond timers for the Z-80 microprocessor

JOSEPH G. DLHOPOLSKY
St. John's University, Staten Island, New York

Two Z-80 machine language millisecond timers for the Model III TRS-80 are presented and described: MSEC DELAY TIMER and MSEC LATENCY TIMER. The latency timer may be used with keyboard input (key release or key press) and may also be designed to respond to a signal at one of the computer's I/O ports. The assembly language programs may be revised for other Z-80-based computers. Calibration procedures are described, and a sample two-field tachistoscope program is presented.

Experiments run on microcomputers may use an external millisecond clock for timing stimulus exposures, interstimulus intervals, and response latencies. Such a hardware clock may be read through an input port or expansion board. Grice (1981) described such a modification for the Model I TRS-80. Flowers (1982), Flowers and Leger (1982), and Rayfield (1982) reported similar techniques for the Apple II. Other computers—DEC's PDP-11 (Ternes, Ehrman, & O'Brien, 1982) and the TERA 8510 (Post & Fox, 1982)—also require external timing hardware.

An external millisecond timer may answer timing requirements, but a software timer would mitigate the cost and interfacing requirements of additional equipment. Although Derr, Wright, and Coriell (1982) argued that software timers create more problems than they solve in complex programs, a software timer may be preferred in settings in which complex stimulus-timing sequences are not the rule or in which programmers feel confident in exercising care in calibrating the timer. Although interstimulus intervals and stimulus exposures may be timed by FOR/NEXT loops in BASIC programs (Perera, 1979), the resolution is at best around 1.35 msec in the Model III TRS-80 (Dlhopolsky, 1983). Moreover, it is often necessary to calibrate a timer for each new use because of idiosyncratic features of BASIC: the length of the current program, the number of variables and arrays being used in the program, the nature of the looping variable used in the timer, and the location of this variable in random access memory (RAM).

Programs written in BASIC, or any other high-level language, also suffer from their inability to synchronize

the appearance of stimulus data in video memory with its appearance on the screen (Dlhopolsky, 1982; Lincoln & Lane, 1980; Merikle, Cheesman, & Bray, 1982; Reed, 1979). Furthermore, when response latency is also recorded, the timing loop must include a command monitoring the subject's response, be it a keyboard manipulation or an external input to one of the computer's I/O ports. This results in a further reduction of resolution for BASIC timers.

Because of the problems with BASIC timers, there is a clear need for machine language millisecond timers. The computer executes machine language routines with great speed: The execution time of a machine language command may be on the order of 2 to 12 microsec. Therefore, millisecond precision is within reach. Such a timer may be calibrated from published information (Radio Shack, 1981; Zaks, 1980a, 1980b). Once developed and calibrated, the timer need not be revised and recalibrated for each new programming use, because its accuracy depends only on the computer's system clock frequency, not on the vagaries of BASIC.

Figure 1 contains the assembly language source code for MSEC DELAY TIMER and MSEC LATENCY TIMER. MSEC DELAY TIMER is a 17-byte machine language routine beginning at hexadecimal (hex) address FFC5 [65477 decimal (dec)]. It may be used as a subroutine that is called whenever an interval in the range of 1-65,536 msec must be timed. This supplements the DELAY subroutine supplied in the Model III TRS-80 read only memory (ROM), which times up to about 1 sec, with a resolution of 14.8 microsec. MSEC LATENCY TIMER is a 32-byte routine that begins at FFD6 hex (65494 dec). It may serve as a subroutine for timing intervals in the same range while monitoring for the subject's response. The version in Figure 1 monitors for any of the keyboard keys to be pressed. Modifications may be made to this routine if the researcher

The author's mailing address is: Department of Psychology,
St. John's University, Staten Island, New York 10301.

```

00100 ;MSEC DELAY TIMER (Revision 8304.11)
00110 ;By Joseph G. Dlhopsky, Ph.D.
00120 ;Calibrated to 2.02752 MHz clock .0237% over error
00130
FFC5      00140      ORG      0FFC5H
FFC5 F3    00150      DI              ;Disable interrupts
FFC6 2AFEFF 00160      LD      HL,(0FFFEH) ;Delay in HL
FFC9 0698   00170 TIMEA  LD      B,152    ;152 = loop constant
FFCB 10FE   00180 LOOPA  DJNZ    LOOPA    ;1 msec = 152 loops
FFCD F600   00190      OR      0        ;7 T-state calibration
FFCF 00     00200      NOP              ;4 T-state calibration
FFD0 2B     00210      DEC      HL        ;1 msec off clock
FFD1 7C     00220      LD      A,H        ;Check end of interval
FFD2 B5     00230      OR      L          ;HL at 0?
FFD3 20F4   00240      JR      NZ,TIMEA   ;No? Time next msec
FFD5 FB     00250      EI              ;Yes? Enable interrupts
FFD6 C9     00260      RET              ;Return from timer
00270
00280
00290
00300 ;MSEC LATENCY TIMER (Revision 8304.11)
00310 ;By Joseph G. Dlhopsky, Ph.D.
00320 ;Calibrated for 2.02752 MHz clock .0237% over error
00330
FFD7      00340      ORG      0FFD7H
0049      00350 KBWAIT  EQU      0049H    ;Key I.D. routine
FFD7 F3    00360      DI              ;Disable interrupts
FFD8 2AFCFF 00370      LD      HL,(0FFFCB) ;Abort latency in HL
FFDB 0697   00380 TIMEB  LD      B,151    ;151 = loop constant
FFDD 10FE   00390 LOOPB  DJNZ    LOOPB    ;Time one msec
FFDF 2B     00400      DEC      HL        ;Take 1 msec off clock
FFE0 3A7F3B 00410      LD      A,(3B7FH)  ;Keyboard address
FFE3 B7     00420      OR      A          ;Check for key press
FFE4 2009   00430      JR      NZ,KEY     ;Identify key
FFE6 7C     00440      LD      A,H        ;Check abort interval
FFE7 B5     00450      OR      L          ;HL = 0?
FFE8 20F1   00460      JR      NZ,TIMEB   ;No? Time next msec
FFEA 22FAFF 00470 EXIT   LD      (0FFFAH),HL ;Store HL time in RAM
FFED FB     00480      EI              ;Enable interrupts
FFEE C9     00490      RET              ;Return from timer
00500
FFEF CD4900 00510 KEY    CALL    KBWAIT   ;Get key pressed
FFF2 32F9FF 00520      LD      (0FFF9H),A ;Store ASCII code in RAM
FFF5 18F3   00530      JR      EXIT      ;Goto EXIT
FFC5      00540      END      0FFC5H
00000 Total Errors

EXIT      FFEA
KEY       FFEF
LOOPB     FFDD
TIMEB     FFDB
KBWAIT    0049
LOOPA     FFCB
TIMEA     FFC9

```

Figure 1. Source code for MSEC DELAY TIMER and MSEC LATENCY TIMER. Column 1 contains the memory addresses for the routine. Column 2 contains the hexadecimal machine code. Column 3 contains the program line number. The next three columns list the assembly language commands. The last column contains comments.

prefers a key-release response or wishes to have an external response signal sent to one of the computer's external input ports.

The routines may be directly used on the Modell III TRS-80 microcomputer with 48K RAM and a system clock frequency of 2.027252 MHz. In their present form, they are calibrated such that 1 millisc equals 1000.2367 microsec. Since the machine language used

here is generic to the Z-80 microprocessor, the routines may be revised and recalibrated to operate on any Z-80-based computer. Revisions, in cases in which they are likely to be necessary, are described below.

The timers may be assembled with any editor-assembler program and stored on tape or disk. Alternatively, the object code may be POKed into RAM as part of a BASIC routine, using the short program in Figure 2.

```

100 FOR J=&HFFC5 TO &HFFF6
110 READ J0
120 POKE J,J0
130 NEXT
140 DATA 243,42,254,255,6,152,16,254,246
150 DATA 0,0,43,124,181,32,244,251,201
160 DATA 243,42,252,255,6,151,16,254
170 DATA 43,58,127,56,183,32,9,124
180 DATA 181,32,241,34,250,255,251,201
190 DATA 205,73,0,50,249,255,24,243

```

Figure 2. A BASIC routine that loads the object code for MSEC DELAY TIMER and MSEC LATENCY TIMER. This routine may be used in place of the normal editor/assembler process.

Table 1
Data Buffers for MSEC DELAY TIMER and
MSEC LATENCY TIMER

Address	Decimal	Contents
FFF7	65527	LSB video synch adjustment
FFF8	65528	MSB video synch adjustment
FFF9	65529	ASCII code for key pressed
FFFA	65530	LSB of abort less response latency
FFFB	65531	MSB of abort less response latency
FFFC	65532	LSB of abort latency
FFFD	65533	MSB of abort latency
FFFE	65534	LSB of delay interval
FFFF	65535	MSB of delay interval
FF80-FFBF	65408-65471	64-byte target stimulus buffer
FF40-FF7F	65344-65407	64-byte probe stimulus buffer

Note—Addresses FFF7, FFF8, FF80-FFBF, and FF40-FF7F are used by TWO-FIELD TACHISTOSCOPE, which is described later in the article.

The routines reside in high memory above 65477 dec or FFC5 hex. So, if initializing TRS-80 BASIC, the MEMORY SIZE query should be answered with a figure below this number. Otherwise, the routines will be overwritten by string variables. This will result in the computer's hanging up or resetting itself when the overwritten machine language routine is called. Seven bytes in high memory (FFF9-FFFF hex or 65529-65535 dec) are also used as data buffers (Table 1).

The routines are relocatable. Users with 32K systems should change the most significant byte (MSB) in all addresses to BF hex. For example, FFFF and FFF9 should be BFFF and BFF9, respectively. Users with 16K systems should use 7F as the MSB: FFFF and FFF9 become 7FFF and 7FF9.

MILLISECOND DELAY TIMING

MSEC DELAY TIMER times a millisecond by decrementing the Z-80's B register 152 times. But before the timer is activated, the desired interval must be stored in least significant byte, most significant byte (LSB, MSB) format in memory addresses FFFE and FFFF hex (65534 and 65535 dec). Note that the LSB is stored in the lower memory address. For example, if 200 msec

is the desired delay, it has a LSB of C8 hex and an MSB of 00 hex. Therefore, C8 would be stored in FFFE and 00 in FFFF. If the delay codes are programmed in assembly language, this is done automatically by the assembler. But if the routine is called from a BASIC program, the programmer must be certain to POKE the delay codes into the correct memory locations before the timer is called.

To facilitate conversion of an integer to LSB, MSB format, a short BASIC program is listed in Figure 3. This routine takes a variable OE in the range of 0 to 65535 and returns the LSB and MSB as I0 and I1 (integer values). In addition to initializing the data buffers, the location of machine language routines must be defined if they are to be called from a BASIC program. In cassette-based TRS-80s, the address of a routine is defined by POKEing the LSB and MSB of the address in addresses 16526 and 16527, respectively. Then a command of the form J=USR(x) calls the routine. In TRSDOS-based disk systems, the location of a machine language subroutine is defined by the DEFUSRx=nn command, where x is any digit and nn is the two-byte memory address. The routine is then called by a command of the form J=USRx(n), where n is the same integer used in the DEFUSR command and x is a dummy expression. Other Z-80-based computers will undoubtedly use different methods.

The operation of MSEC DELAY TIMER is fairly simple. Once the desired interval has been stored in the HL register (line 160), the B register is set to the looping variable of 152 (line 170). The DJNZ (line 180) command causes a successive decrement of the B register until it reaches 0. Then HL, which counts down milliseconds, is decremented (line 210). The LD A,H command (line 220) stores the MSB of HL in the A register for comparison with the LSB (stored in register L). If the H and L register both contain 0, the A register will contain 0 after the OR L command is executed in line 230. The JR NZ, TIMEA in line 240 means "Jump to time the next millisecond if A does not equal zero." If A contains 0, line 260 causes a return to the source from which the subroutine was called (BASIC or the parent machine language program).

A number of features in MSEC DELAY TIMER need explanation. The DI command in line 150 causes the computer to disable all maskable interrupts. An

```

100 CLS:INPUT "Enter an integer between 0 and 65535";OE
110 IF OE<0 OR OE>65535 THEN 100
120 IF OE>32767 THEN II=-1*(65536-OE) ELSE II=OE
130 I0=IIAND255
140 I1=(II-I0)/256
150 DI:II=IIAND255
160 PRINT "LSB =";I0,"MSB =";I1
170 POKE&HFFC,I0:' LSB of abort latency
180 POKE&HFFD,I1:' MSB of abort latency
190 POKE&HFFE,I0:' LSB of delay interval
200 POKE&HFFF,I1:' MSB of delay interval
210 GOTO100

```

Figure 3. A BASIC routine that converts a number in the range of 1-65,535 into its LSB, MSB representation.

interrupt is a high-priority message that causes the microprocessor to stop what it is doing and perform some other task before returning to the interrupted task. In the Model III TRS-80, the real-time clock is maintained by interrupt processing. If the interrupts are allowed to remain active, it will be difficult to calibrate the timer according to the procedure described below. Therefore, it is best to disable interrupts. Subsequently, they must be reenabled at the end of the routine by the EI command (line 250).

Ideally, the DI command should be executed before the timer is called, because an interrupt could occur before the DI instruction is executed and this would disrupt the timing accuracy and video synchronization (if any is used). This is no problem when all trial events are controlled by machine language, because the DI command can be placed at the very beginning of the routine before any crucial timing is carried out. However, if the timer is called from BASIC, it is not always possible to assure that interrupts are disabled. A two-byte machine language subroutine (decimal codes 243 and 201, DI and RET commands) that disables interrupts can be called from BASIC, but some BASIC commands, specifically CLS and PRINT commands (but there may be others), reenables interrupts in the normal course of their execution. All things considered, it is not wise to bounce back and forth from BASIC to machine language in controlling trial events.

Published T-states for Z-80 commands (Zaks, 1980a) were used to calibrate MSEC DELAY TIMER. The T-state is one clock cycle of the computer's system clock. The Model III TRS-80's clock operates at a frequency of 2.02752 MHz, so one T-state is equivalent to $1/2,027,520$ sec, or .4932 microsec. Since the system clock frequency varies from one manufacturer to another, the T-state, rather than the microsecond, is customarily used as the time unit for execution of Z-80 instructions. Knowing the system clock frequency for any other Z-80-based computer, the programmer can calculate the execution time of a routine from the T-states of the constituent instructions.

The T-states for a selection of commands used in MSEC DELAY TIMER and MSEC LATENCY TIMER appear in Table 2 (Zaks, 1980a). To calibrate a timer, the T-states for each command must be identified. Then the T-states are summed, with care being taken to multiply the T-states for loops by the number of times the loop is cycled. Once the T-states are summed, the following formula may be used to determine how long a routine takes:

$$t = T/\text{MHz.} \quad (1)$$

In this equation, t is the execution time measured in microseconds, T is the number of T-states used up by the commands, and MHz is the frequency of the system clock. According to this equation, a routine made up of 100 T-states would take 49.32 microsec to execute

Table 2
T-States for Some of the Z-80 Assembly Language
Commands Used in MSEC DELAY TIMER and
MSEC LATENCY TIMER

Command	T-States	Time
DI	4	1.9728
LD HL,(0FFFEH)	16	7.8914
LD B,152	7	3.4525
DJNZ	13*	6.4118
DJNZ	8**	3.9457
OR 0	7	3.4525
NOP	4	1.9728
DEC HL	6	2.9593
LD A,H	4	1.9728
OR L	4	1.9728
JR NZ,TIMEA	12†	5.9186
JR NZ,TIMEA	7††	3.4525
EI	4	1.9728
RET	10	4.9321
LD A,(387FH)	13	6.4118
LD (0FFFAH),HL	16	7.8914
CALL	17	8.3846
LD (0FFF9H),A	13	6.4118
JR EXIT	12	5.9186

Note—Times are measured in microseconds, for 2.02752 MHz.
*each loop. **at loop end. †if A not 0. ††if A = 0.

on a Model III TRS-80. Alternatively, Equation 2 describes the calculation of the necessary T-states for a desired interval:

$$T = t(\text{MHz}). \quad (2)$$

On the Model III TRS-80, 1 msec (1,000 microsec) requires 2,027.52 T-states. Since T-states cannot be fractional, 1 millisecc must use either 2,028 or 2,027 T-states, corresponding to 1,000.2367 or 999.7435 microsec.

The DJNZ command (line 180) requires 13 T-states to complete whenever register B is nonzero. Therefore, 152 executions of DJNZ uses up 1,976 T-states. Following this, 41 T-states are required to determine if the interval has been completed (lines 210-240). The cumulative T-states now number 2,017. The OR A and NOP commands in lines 190 and 200 are dummy commands that carry out no usable function other than to take up an additional 11 T-states, for a total of 2,028.

Other commands in MSEC DELAY TIMER appear before and after the timing section. These add an overhead to the entire routine and remain fixed regardless of the length of the desired interval. All told, these use up a constant 41 T-states, for a total of 20.2217 microsec.

MSEC DELAY TIMER will operate on any Z-80-based computer. However, in order to calibrate the timer for another Z-80-based device, such as the Model I or Model 4 TRS-80, the owner's manual must be consulted for the system clock rate. Then the number of T-states necessary to time 1 millisecc can be determined from Equation 2. MSEC DELAY TIMER may then be revised

by changing the looping constant in the B register (line 170) or by adding or removing dummy commands. The programmer must be careful, however, not to use a dummy command that affects the condition of the HL register or reenables the interrupts.

LATENCY TIMING

MSEC LATENCY TIMER operates on the same principle as that of MSEC DELAY TIMER. The timing section also uses 2,028 T-states, so each millisecond is 1000.2367 microsec. The major difference between the two routines is that the microprocessor must also monitor for the subject's response. In the listed version, the response is a key press. However, a simple change in one command alters the routine to monitor for a key release. A more involved revision is necessary if the response generates a signal that appears at one of the computer's I/O ports. These variations will be addressed below.

The techniques for monitoring for a key press, a key release, and an input signal to an external port are highly dependent on the brand of computer in use. Not all computers have a memory-mapped keyboard like that of the TRS-80. And if the keyboard is memory mapped, the architecture may not be the same. Furthermore, MSEC LATENCY TIMER uses a Radio Shack Level III BASIC ROM routine that returns the ASCII code for a key after it has been pressed. It is unlikely that this routine is located in the same memory address in other computers. Consequently, MSEC LATENCY TIMER will require more substantial revisions for other Z-80-based computers than does MSEC DELAY TIMER.

In measuring responses and response latency, it is wise to provide an abort latency to determine an upper limit for the response interval. Without such a limit, the computer would time an unpredictable interval up to 65,536 msec whenever the subject failed to respond. An abort latency is used in MSEC LATENCY TIMER to initialize the millisecond clock and is determined in a manner similar to the delay interval in MSEC DELAY TIMER. It is passed to the routine in LSB,MSB format in addresses FFFC and FFFD hex (65532 and 65533 dec). The value of the abort latency is decremented once for each millisecond timed. When the resulting figure is zero, control returns to the source from which the timer was called.

Line 350 identifies the address for the ROM routine (KBWAIT) that is used to identify the key pressed, as tested by lines 410-430. Note that the KBWAIT routine is called only after a key is pressed. At this point, the HL register contains the decremented abort latency. Since the KBWAIT routine does not use the HL register, this value remains unchanged and can be stored in RAM at a more convenient time. The KBWAIT routine will not be used in the key-release or port-input versions. Lines 370-460 serve the same timing purpose as the similar lines (lines 160-240) in MSEC DELAY TIMER.

Lines 410 and 420 check to see if a key is being pressed after each millisecond is timed. The RAM address 387F hex (14463 dec) is one of the keyboard memory addresses in the TRS-80 Model I and the Model III. If any key is being depressed at the time the LD A, (387FH) instruction is executed, the byte stored in 387F will be nonzero. This condition is checked by the OR A command. If nonzero is signaled, the Z-80 jumps to the KEY routine (line 510), which calls KBWAIT to get the ASCII character of the key pressed. This code is returned from KBWAIT in register A. Line 520 stores the code in RAM address FFF9 hex (65559 dec). Then control is passed to EXIT (line 470), where the decremented abort latency is stored in LSB,MSB format in RAM addresses FFFA and FFFB hex (65560 and 65561 dec).

Note that MSEC LATENCY TIMER decrements the abort latency to time response latency. If a subject has a 250-msec reaction time and the abort latency was 2 sec, the values returned in the FFFA and FFFB hex addresses will be 214 and 6, respectively. Multiplying the MSB (6) by 256 and adding this value to the LSB (214) gives 1,750 msec. When subtracted from 2,000 msec, the abort latency, this returns the actual response latency. A short BASIC routine that carries out this process is:

```
100 J=PEEK(&HFFFA)+256*PEEK(&HFFFB)
110 J = 2000-J: 2000 is the abort latency
```

Note that the expressions &HFFFA and &HFFFB are TRSDOS codes for hexadecimal values. Not all computers express hexadecimal numbers in this way, so their decimal counterparts may be used (-6 and -5, respectively).

If the subject has not pressed a key, then the Z-80 will check to see if the abort latency has been reached. If so, the HL register will contain a zero and this will be tested in lines 440-460, as in the analogous portion of MSEC DELAY TIMER. When the abort latency has been reached, 0 is stored in both FFFA and FFFB. If the abort latency has not been reached, then the next millisecond is timed.

If MSEC LATENCY TIMER is to be used to test key release, the command in line 430 should be JR Z,EXIT. The KEY routine (lines 510-530) can be deleted. When the subject releases a continuously depressed key, 387F will go to zero; otherwise, it will be nonzero. Accordingly, the JR Z,EXIT command jumps to the routine EXIT when a key is released. This command uses the same number of T-states as the JR NZ,EXIT command; therefore, the timing need not be recalibrated.

If MSEC LATENCY TIMER is to be used to monitor input to one of the computer's ports, the command in line 410 should be IN A,(n), where n is the port address. The value at port n will be stored in the A register. With the JR,NZ command in line 430, the timer will remain

active as long as the port byte is 0. If the researcher wishes 0 to be the signal for a response and any non-zero byte to be the signal for no response, then the JR,NZ command may be replaced by JR,Z.

The IN A,(n) command used 11 T-states, whereas the LD A,(387FH) command uses 13 T-states. As a result, the timing of one millisecond will be 999.2503 microsec, a .075% underestimation error. This may be corrected by reducing the looping variable in line 380 to 150 and adding two OR A dummy commands (14 T-states) after line 390 to bring the number of T-states back up to 2,027, or 999.7435 microsec. The CALL KBWAIT command (line 510) can be removed from the KEY subroutine. In order to record the response-port state, however, line 520 should be retained and edited to read KEY LD (OFFF9H),A.

The constant-overhead T-states in MSEC LATENCY TIMER depend on the version of the routine that is used and also on whether a response has been made. If the subject fails to make a response in any version—key press, key release, or port input—45 T-states (22.1945 microsec) exist as overhead. In the key-press version, at least 87 T-states (42.9095 microsec) exist as overhead when the subject responds. This does not include the T-states entailed in the KBWAIT routine [but does include the necessary CALL and RET (return) from the routine]. The command listing of the KBWAIT routine is proprietary information, so the T-states are not available. If a key-release response is made, then there are 35 T-states (17.2625 microsec) as overhead. When a response is detected in the port-input version, 40 T-states will exist as overhead (19.7285 microsec). The overhead times will be constant regardless of the subject's response latency. Since they are only a small fraction of a millisecond, they can be considered transparent. The only time they may be a concern is if subsequent stimuli are to be displayed in the trial. In this case, the overhead times can be eliminated during video synchronization waits.

An important point regarding overhead is that additional time will be taken if the routines are called from BASIC. In addition to POKEing interval parameters and routine starting addresses, a USR call/return takes about 4 msec (Model III TRS-80) beyond the actual execution time of the machine language routine (Dlhopsky, 1983). Because of this, it is recommended that all trial events be operated throughout by machine language. Moreover, it is often desirable to synchronize stimulus displays with the CRT's electron beam (Dlhopsky, 1982), which requires a machine language routine. Thus, all things considered, video synchronization, stimulus display, interval timing, and response recording should all be under machine language control.

A TWO-FIELD TACHISTOSCOPE

Figures 4 and 5 list the source code for a BASIC/machine language hybrid program that demonstrates a use of the millisecond timers. The program is designed

and calibrated for a 48K Model III TRS-80, using the TRSDOS 1.3 disk-operating system. For computers with 16K or 32K RAM, the MSB of RAM addresses can be converted as previously described. Non-TRSDOS users may have to make some changes to the BASIC section, particularly in the representation of hexadecimal values. Any editor-assembler can assemble the machine language section from the source code in Figure 5. Care should be taken to be sure that the resulting machine code disk file is compatible with the disk-operating system used to load the BASIC section. The file name used here for the machine language section is TSCOPE/CMD. Alternatively, the short BASIC program in Figure 6 loads the codes for the machine language section into memory directly from BASIC. Regardless of which method is used to load the machine language section, high memory must be reserved to protect the section from being overwritten by string data.

The BASIC section has the user select timing parameters and stimuli, which are then formatted and stored in the data buffers (Table 1). The machine language section is then called to display the target stimulus for the designated interval, which is timed by a version of MSEC DELAY TIMER. Then a probe stimulus is displayed and remains on the screen until a key is pressed or the abort interval is reached. The target and probe stimuli can be up to 64 characters long and are displayed on the video screen in the center of the sixth screen line. When the machine language routine is completed, control returns to the BASIC section that displays the key pressed and the response latency, if any. For simplicity, the probe stimulus remains on the screen until a key is pressed. However, the program may be modified to control stimulus exposure by calling either MSEC DELAY TIMER or MSEC LATENCY TIMER (if the exposure exceeds the fastest subject's reaction time) to time the stimulus exposure. Then the stimulus can be erased or masked, and the abort latency can be timed. This modification will not be addressed here. A detailed description of the program follows.

In the BASIC program, line 100 initializes the program by calling line 9300. The subroutine starting there defines variable ranges as strings (A-H), integers (I-N), and double precision (X-Z) (O-W are single precision by default), loads the machine language routine and identifies its starting address, and POKEs the MSB of the video synchronization timing adjustment. The LSB of this adjustment depends on the target-stimulus exposure that is selected and will be described below; the MSB remains unchanged, regardless of the adjustment. Lines 110-170 ask the user to define the abort latency and target-stimulus exposure. The entered values are converted to their LSB,MSB representation and stored in the data buffers. Note that the target-stimulus exposure must be a multiple of 16.7 msec because of the CRT scan frequency, so any value entered is adjusted to the nearest valid integer. Since MSEC DELAY TIMER times in milliseconds, and since the stimulus exposure may be off by as much as ± 333 msec because of screen scanning,

```

10 ' TWO-FIELD TACHISTOSCOPE DEMONSTRATION
    48-KB Model III TRS-80, TRSDOS 1.3
    By Joseph G. Dlhopsky, Ph.D.

20 ' Revised 8308.05

100 CLS: CLEAR500: GOSUB9300
110 INPUT "Enter abort latency (msec):"; PA
113 IF PA<1 THEN PA=1: 'Checks for 0 input
120 OE=PA: GOSUB9050: 'Gets LSB, MSB components
130 POKE&HFFFC, I0: POKE&HFFFD, I1: 'POKEs into abort latency buffer
140 INPUT "Enter target stimulus exposure (msec):"; OE
150 OA=OE/50*3: GOSUB9000: OA=OA*50/3: PO=OA: GOSUB9000: OE=OA: 'Rounds to nearest integer multiple of 16.7 msec
152 IF OE<17 THEN OE=17: PO=16.7: 'Checks for 0 input
155 PRINT TAB(5) " ( Rounded to"; OE; ")"
157 IF PO=OE<0 THEN POKE&HFFF7, 211: 'Adjusting vid sync delay for 333 usec error
160 IF PO=OE<0 THEN POKE&HFFF7, 166: 'Same as line 157
165 GOSUB9050: 'Gets LSB, MSB components
170 POKE&HFFFE, I0: POKE&HFFFF, I1: 'POKEs into target stimulus exposure buffer
180 INPUT "Enter target stimulus:"; BA
190 B0=STRING$((64-LEN(BA))/2, 128)+BA: BA=B0+STRING$(64-LEN(B0), 128): 'Centers target stimulus in blanks
200 FOR J=1 TO LEN(BA): POKE&HFF3F+J, ASC(MID$(BA, J, 1)): NEXT: 'POKEs target stimulus in screen buffer
210 INPUT "Enter probe stimulus:"; BB
220 B0=STRING$((64-LEN(BB))/2, 128)+BB: BB=B0+STRING$(64-LEN(B0), 128): 'Centers probe stimulus in blanks
230 FOR J=1 TO LEN(BB): POKE&HFF7F+J, ASC(MID$(BB, J, 1)): NEXT: 'POKEs probe stimulus in screen buffer
240 PRINT: PRINT "Press any key to begin.": GOSUB9100
250 IF PEEK(&H387F)>0 THEN 250: 'Waits for key release
260 CLS: FOR J=1 TO 1481: NEXT: '2 second pause
270 J=USR(0): 'Calls machine language routine
280 BC=CHR$(PEEK(&HFFF9)): 'Gets key pressed
290 PRINT "Key pressed: "; IF BC="" THEN PRINT "None"; ELSE PRINT BC;
295 PRINT " (ASCII Code: "; PEEK(&HFFF9); ")"
300 P0=PEEK(&HFFFA)+256*PEEK(&HFFFB): 'Converts response latency from LSB, MSB representation
310 PRINT "Response latency: "; IF P0=0 THEN PRINT " No response" ELSE PRINT PA-P0; " msec"
320 PRINT: PRINT: PRINT "Select option: "; PRINT
330 PRINT TAB(5) "(1) New abort latency."
340 PRINT TAB(5) "(2) New target exposure."
350 PRINT TAB(5) "(3) New target stimulus."
360 PRINT TAB(5) "(4) New probe stimulus."
365 PRINT TAB(5) "(5) Repeat the last trial."
366 PRINT TAB(5) "(6) Stop."
370 GOSUB9100: J0=VAL(A): IF J0<1 OR J0>6 THEN 370
375 BC="": POKE&HFFF7, 188: 'Initialize for next trial
380 CLS: ON J0 TO 110, 140, 180, 210, 240: CLS: END
9000 IFOA=INT(OA)<.49999 THEN OA=INT(OA) ELSE OA=INT(OA+1): 'Rounding routine
9010 RETURN
9050 IFOE>32767 THEN I1=-1*(65536-OE) ELSE I1=OE: 'LSB, MSB conversion
9055 I0=I1 AND 255: I1=(I1-I0)/256: I1=I1 AND 255
9060 RETURN
9100 POKE16420, 0: POKE16409, 1: A=INKEY$: 'Waits for user key press
9110 A=INKEY$: IFA="" THEN 9110 ELSE RETURN
9300 DEF STRA-H: DEF INTI-N: DEF DBLX-Z: 'Program initialization
9305 DEF USR0=&HFE0: CMD "L", "TSCOPE/CMD": 'Loads machine language routine
9310 POKE&HFFF8, 1: POKE&HFFF8, 188: 'MSB of video synchronization delay
9390 RETURN

```

Figure 4. The BASIC section of TWO-FIELD TACHISTOSCOPE.

```

00130 ;TWO-FIELD TACHISTOSCOPE
00140 ;By Joseph G. Dlhopsky, Ph.D.
00150 ;Revised 8308.03
00160 ;
FEC6 00170 ORG 0FEC0H
0060 00180 DELAY EQU 0060H ;usec delay ROM
0049 00190 KWAIT EQU 0049H ;Key I.D. routine
00200 ;*****
00210 ;$ Revector real time clock interrupt for $
00220 ;$ video synchronization $
00230 ;*****
FEC0 F3 00240 DI ;Disable interrupts
FEC1 21CBFE 00250 LD HL,START ;T-scope routine address
FEC4 224740 00260 LD (4047H),HL
FEC7 FB 00270 EI
FEC8 76 00280 HALT ;Wait for RTC int
FEC9 FB 00290 EI
FECA C9 00300 RET ;Return to BASIC
00310 ;*****
00320 ;$ Transfer target buffer to screen & time exposure $
00330 ;$ upon vertical sync signal from RTC interrupt $
00340 ;*****
FECB F3 00350 START DI ;Disable interrupts
FECC 2140FF 00360 LD HL,0FF40H ;Start of target buffer
FECF 11403D 00370 LD DE,3D40H ;Screen RAM
FED2 014000 00380 LD BC,64 ;Bytes to transfer
FED5 EDB0 00390 LDIR ;BO!
FED7 CD00FF 00400 CALL TIMERA ;Time exposure
00410 ;*****
00420 ;$ Transfer probe buffer to screen RAM, wait for CRT $
00430 ;$ beam to reach line 6, call latency timer, $
00440 ;$ and monitor key press $
00450 ;*****
FEDA 2100FF 00460 LD HL,0FFB0H ;Start of probe buffer
FEDD 11403D 00470 LD DE,3D40H ;Screen RAM
FEE0 014000 00480 LD BC,64 ;Bytes to transfer
FEE3 EDB0 00490 LDIR ;BO!
FEE5 ED4BF7FF 00500 LD BC,(0FFF7H) ;Wait for vid sync
FEE9 CD4000 00510 CALL DELAY
FEEC CD18FF 00520 CALL TIMERB ;Begin timing latency
FEEF 22FAFF 00530 EXIT LD (0FFFAH),HL ;RT in RAM
FEF2 32F9FF 00540 LD (0FFF9H),A ;ASCII code of response
FEF5 014000 00550 LD BC,64
FEF8 21403D 00560 LD HL,3D40H
FEFB 3680 00570 ERASE LD (HL),128
FEFD 23 00580 INC HL
FEFE 10FB 00590 DJNZ ERASE
FF00 212935 00600 LD HL,3529H ;Normal RTC int vector
FF03 224740 00610 LD (4047H),HL
FF06 ED4D 00620 RETI ;Return from interrupt
00630 ;*****
00640 ;$ MILLISECOND DELAY TIMER $
00650 ;*****
FF08 2AFEFF 00660 TIMERA LD HL,(0FFFEH) ;Delay in HL
FF0B 0698 00670 MSEC A LD B,152 ;152 = loop constant
FF0D 10FE 00680 LOOPA DJNZ LOOPA ;1 msec = 152 loops
FF0F F600 00690 OR 0 ;7 T-state calibration
FF11 00 00700 NOP ;4 T-state calibration
FF12 28 00710 DEC HL ;1 msec off clock
FF13 7C 00720 LD A,H ;Check end of interval
FF14 B5 00730 OR L ;HL at 0?
FF15 20F4 00740 JR NZ,MSECA ;No? Time next msec
FF17 C9 00750 RET ;Return from timer
00760 ;*****
00770 ;$ MILLISECOND LATENCY TIMER (key press version) $
00780 ;*****
FF18 2AFCFF 00790 TIMERB LD HL,(0FFFDH) ;Abort latency in HL
FF1B 0697 00800 MSEC B LD B,151 ;151 = loop constant
FF1D 10FE 00810 LOOPB DJNZ LOOPB ;Time one msec
FF1F 28 00820 DEC HL ;Take 1 msec off clock
FF20 3A7F38 00830 LD A,(387FH) ;Keyboard address
FF23 B7 00840 OR A ;Check for key press
FF24 C24900 00850 JP NZ,KWAIT ;Identify key
FF27 7C 00860 LD A,H ;Check abort interval
FF28 B5 00870 OR L ;HL = 0?
FF29 20F0 00880 JR NZ,MSECB ;No? Time next msec
FF2B C9 00890 RET ;Return from timer
FEC6 00900 END 0FEC0H
00000 Total Errors

LOOPB FF1D
MSECB FF1B
LOOPA FF0D
MSECA FF0B
ERASE FEFB
EXIT FEEF
TIMERB FF18
TIMERA FF08
START FECB
KWAIT 0049
DELAY 0060

```

Figure 5. The machine language section of TWO-FIELD TACHISTOSCOPE.


```

100 FOR J=&HFEC0 TO &HFF2B
110 READ J0
120 POKE J,J0
130 NEXT
140 DATA 243,33,203,254,34,71,64,251
150 DATA 118,251,201,243,33,64,255,17
160 DATA 64,61,1,64,0,237,176,205
170 DATA 8,255,33,128,255,17,64,61
180 DATA 1,64,0,237,176,237,75,247
190 DATA 255,205,96,0,205,24,255,34
200 DATA 250,255,50,249,255,1,64,0
210 DATA 33,64,61,54,128,35,16,251
220 DATA 33,41,53,34,71,64,237,77
230 DATA 42,254,255,6,152,16,254,246
240 DATA 0,0,43,124,181,32,244,201
250 DATA 42,252,255,6,151,16,254,43
260 DATA 58,127,56,183,194,73,0,124
270 DATA 181,32,240,201

```

Figure 6. A BASIC routine that loads into memory the object code of the machine language section of TWO-FIELD TACHISTOSCOPE. This routine may be used in place of the normal editor/assembler process.

a video synchronization adjustment sometimes must be made. The LSB,MSB values for this adjustment are POKed into memory locations FFF7 and FFF8. FFF8 was initialized in line 9310; FFF7 receives its value in lines 157-160.

The machine language routine is keyed to begin when a vertical synch pulse is detected. This pulse occurs 8.0 msec before the electron beam approaches the stimulus display on the sixth screen line. During this interval, all the timer and display T-state-overhead and video-synchronization adjustments are made. As a result, the latency timer begins within microseconds of the actual stimulus appearance. A more detailed description of this video synchronization technique appears in Dlhopsky (1982).

Lines 180-230 of the BASIC section have the user enter the target and probe stimuli, which are then centered in the data buffers. Lines 240-250 ask the user to press any key to begin the trial. When the key is released, a 2-sec delay occurs (line 260), and the machine language routine is called (line 270). After the trial has ended, the response data are displayed on the screen in a straightforward manner.

A description of the machine language section follows. Lines 180-190 identify the addresses of two Radio Shack Level III BASIC ROM routines that are used by the program. KBWAIT is used to identify the key pressed. DELAY is used to delay the start of MSEC LATENCY TIMER until the CRT's electron beam is about to scan the stimulus display line. Lines 240-280 revector the real-time clock interrupt, which operates off the video vertical synch pulse (Dlhopsky, 1982). The next interrupt to occur starts the execution of the main portion of the routine at line 350. At this point,

the location of the video electron beam is known and can be synchronized with the latency timer.

In line 350, maskable interrupts are disabled for the remainder of the routine. Then the target stimulus buffer is transferred to the video RAM for the sixth screen line and MSEC DELAY TIMER is called. After the programmed interval, the probe buffer is transferred to video RAM (lines 460-490), but the stimulus will not immediately appear because the electron beam is pointing elsewhere at this instant. Lines 500-510 arrange for the proper interval to pass (adjusted for all T-state-overhead and video-synchronization requirements) so that MSEC LATENCY TIMER is called when the electron beam is aimed at the proper location for the stimulus to appear on the screen. When a response is made or the abort latency is reached, data are stored in the buffers, the real-time clock interrupt is returned to normal, control returns from the interrupt-processing routine to line 290, and line 300 returns control to the BASIC section.

The BASIC section of TWO-FIELD TACHISTOSCOPE may be made part of a larger BASIC program by any programmer familiar with BASIC. TRSDOS conventions may have to be changed if the program is not used with this operating system. The machine language section requires no modification as long as it is run on a 48K Model III TRS-80 or on a Model 4 operating in the Model III mode. The video synchronization technique will not work on a Model I TRS-80 (see Grice, 1981, for a solution). Other Z-80-based computers may or may not have a software solution to the video synchronization problem.

It is apparent that machine language control over trial events provides acceptable precision for a wide variety of experiments. There is no compelling reason for experimental psychologists to resist assembly language programming; it is not all that difficult, requiring mostly attention to detail. By way of example, both the BASIC and machine language sections of TWO-FIELD TACHISTOSCOPE were designed, coded, calibrated, and tested in less than 1 week. A more complex program would take longer, but would not involve breaking new conceptual ground.

REFERENCES

- DERR, M. A., WRIGHT, C. E., & CORIELL, A. S. A satellite system for controlling real-time experiments. *Behavior Research Methods & Instrumentation*, 1982, 14, 264-271.
- DLHOPOLSKY, J. G. Software synchronizing of video displays and Z-80 processing in the Model III TRS-80. *Behavior Research Methods & Instrumentation*, 1982, 14, 539-544.
- DLHOPOLSKY, J. G. Limitations of high-level microcomputer languages in software designed for psychological experimentation. *Behavior Research Methods & Instrumentation*, 1983, 15, 459-464.
- FLOWERS, J. H. Some simple Apple II software for the collection and analysis of observational data. *Behavior Research Methods & Instrumentation*, 1982, 14, 240-248.
- FLOWERS, J. H., & LEGER, D. W. Personal computers and be-

- havioral observation: An introduction. *Behavior Research Methods & Instrumentation*, 1982, 14, 227-230.
- GRICE, G. R. Accurate reaction time research with the TRS-80 microcomputer. *Behavior Research Methods & Instrumentation*, 1981, 13, 674-676.
- LINCOLN, C. E., & LANE, D. M. Reaction time measurement errors resulting from the use of CRT displays. *Behavior Research Methods & Instrumentation*, 1980, 12, 55-57.
- MERIKLE, P. M., CHEESMAN, J., & BRAY, J. PET flasher: A machine language subroutine for timing visual displays and response latencies. *Behavior Research Methods & Instrumentation*, 1982, 14, 26-28.
- PERERA, T. B. A comparison of BASIC language timing loops for the TRS-80 microcomputer. *Behavior Research Methods & Instrumentation*, 1979, 11, 592.
- POST, T. A., & FOX, J. L. Programming experiments on a TERA 8510/A microcomputer. *Behavior Research Methods & Instrumentation*, 1982, 14, 276-280.
- RADIO SHACK. *Series I editor assembler manual*. Fort Worth, Tex: Author, 1981.
- RAYFIELD, F. Experimental control and data acquisition with BASIC in the Apple computer. *Behavior Research Methods & Instrumentation*, 1982, 14, 409-411.
- REED, A. V. Microcomputer display timing: Problems and solutions. *Behavior Research Methods & Instrumentation*, 1979, 11, 572-576.
- TERNES, J. W., EHRLMAN, R., & O'BRIEN, C. P. A software system for real-time control of psychological experiments. *Behavior Research Methods & Instrumentation*, 1982, 14, 160-164.
- ZAKS, R. *Programming the Z-80* (2nd ed.). Berkeley, Calif: Sybex, 1980. (a)
- ZAKS, R. *Programming the 6502* (3rd ed.). Berkeley, Calif: Sybex, 1980. (b)

(Manuscript received March 29, 1983;
revision accepted for publication August 12, 1983.)