Behavior Research Methods, Instruments, & Computers
1987, 19 (5), 457-459

COMPUTER TECHNOLOGY

Accurate display timing using an enhanced BASIC for the Tandy Color Computer

ARTHUR J. FLEXSER
Florida International University, Miami, Florida

The general suitability of the Tandy Color Computer as an inexpensive laboratory microcomputer is discussed. A program is described that adds several useful timing-related functions to the native BASIC in the Tandy Color Computer, thus allowing BASIC to be used for programming experiments involving tachistoscopic displays and reaction times.

A number of authors have noted the general unsuitability of the BASIC language for microcomputer laboratory applications requiring precise display timing (e.g., Green & Shwartz, 1978; Mapou, 1982). Mapou, for example, wrote

> Attempts to do reaction timing in BASIC (Perera, 1978, 1979) indicate that timing cannot be performed with the required accuracy, even with an external clock. This is due to the fact that BASIC is an interpreted language: Each individual line is read by the computer, translated into machine language, and executed. The process is time-consuming and limits the accuracy of any timing method. Additionally, the start of the reaction time interval must also occur at the same time as stimulus onset, and this coordination is impossible in BASIC. (p. 534)

If these difficulties could be overcome, there would be a number of advantages to using BASIC as a program ming language for control of laboratory experiments. The language is widely known and relatively easy to learn and use. The fact that it is interpreted rather than compiled speeds program development by eliminating the necessity for a compilation stage. Furthermore, the language is supplied with many microcomputers, often in ROM. For these reasons, BASIC is often the language of choice in applications for which speed is not critical.

What is not so widely known about BASIC is that the structure of the BASIC interpreter is generally quite modular, permitting additional commands and functions, written as assembly-language routines, to be readily added to the language. Microcomputer BASICs usually have special provisions written into the interpreter to facilitate making such additions to the command set. Periodicals devoted to specific microcomputers occasionally run articles with examples of how the capabilities of BASIC may be extended in such a fashion. This extendability aspect of BASIC offers the opportunity for overcoming the timing problems noted above, by the addition of specialized timing routines to the language.

As Mapou (1982) pointed out, BASIC programs are interpreted line by line. (If several statements are included on a single line, as is allowed in many BASICs, each such statement is also individually interpreted.) He was not quite correct in asserting that the lines are translated into machine language. It would be more accurate to say that when the interpreter encounters a BASIC command or function keyword, it transfers control to a specific machine-language routine within itself corresponding to that particular command or function. This routine does any necessary evaluation of arguments and performs the other operations involved in executing the specified command, and then returns control to the top level of the interpreter. Therefore, if a sequence of steps requires accurate timing, one way of meeting this requirement is by extending the BASIC language in such a way as to allow a single BASIC statement to represent the entire sequence of operations for which timing is critical. It is also necessary that the timing-related commands be written in such a way that the entire BASIC statement is interpreted, that is, all arguments are evaluated, prior to the actual execution of the command. This ensures that variations in interpretation time will not affect display or reaction timing.

THE TANDY COLOR COMPUTER

On the basis of cost and capabilities, the Tandy (Radio Shack) Color Computer, popularly known as the CoCo, represents an attractive machine for laboratory use. Reed (1982) presented a detailed evaluation of this microcomputer, but some of the information is now somewhat dated. A strong point of the CoCo is its processor, the 6809. Although this processor uses an 8-bit bus, its internal registers are 16-bit and its assembly language is versatile and logically structured. The version of BASIC included

Requests for reprints should be sent to Arthur J. Flexser, Department of Psychology, Florida International University, University Park Campus, Miami, FL 33199. Copies of LBASIC are available on Color Computer diskette or cassette, with the assembler source code, for a fee of $10 to cover expenses.

in the CoCo ROMs is relatively strong and highly similar to that in the IBM PC. Disk input/output is well integrated with the BASIC language, and both sequential and direct-access file types are supported.

Two aspects of the design of the CoCo make it particularly suitable for use as a tachistoscope. First, the video display may be placed so as to have its origin at any 512-byte boundary of RAM, making it possible to set up multiple displays in RAM and to switch the display window rapidly from one area of RAM to another.[1] Second, both the vertical and horizontal scan synchronization pulses generate interrupts, making it possible, using the 6809's SYNC instruction, to achieve precise synchronization of the display to machine-language events. The horizontal synchronization pulse, which occurs approximately every $\frac{1}{16}$ of a millisecond, makes an excellent timing unit, and allows the CoCo to be used for collecting reaction times to submillisecond accuracy without the need of an external clock (Reed, 1982).[2]

The cost of the Color Computer is quite modest, relative to its capabilities. At the time of this writing, in early 1987, the 64K Color Computer 2 (CoCo 2) with Extended Color Basic was available for under $100. This machine is currently being phased out by Tandy in favor of the newly introduced Color Computer 3 (CoCo 3). If purchasing a CoCo 2, one should specify a machine with a model number ending in "B," because the earlier models have a video display generator that does not provide for the display of lowercase letters. A 128K CoCo 3, which features 80×24 and 40×24 text display modes in addition to the original CoCo's 32×16 display, is available from discounters for about $170. The difference in cost between the two models is effectively reduced by the fact that the CoCo 2, but not the CoCo 3, requires a separate interface (available from third parties for about $25) for use with a composite monitor. The CoCo 3 includes both composite and analog RGB video outputs, as well as the RF output included in the earlier models.

The CoCo uses standard, IBM-compatible, 5¼-in. disk drives, which can be obtained for about $100 with case and power supply, and requires, for use with disk, a disk controller that sells for about $100. Monochrome composite monitors sell for about $70. Thus, a complete CoCo 3 system suitable for laboratory use can be assembled for approximately $450 at today's prices. This fact makes the CoCo especially attractive for setting up a low-cost student laboratory with multiple stations. The $450 price can be lowered by about $200 if the purchaser is willing to use a cassette recorder for program and data storage. This might be an attractive short-term alternative in cases where funding would permit later upgrading to disk storage.

LBASIC

LBASIC (for "laboratory BASIC") is an enhanced version of the CoCo's BASIC that has been modified by the addition of various timing-related and other commands and functions. System requirements for LBASIC are a CoCo 1 or 2 with 64K of RAM and Extended Color BASIC. A disk system is not required. LBASIC-3, which supports the CoCo 3's 80-column text screen, requires a 128K CoCo 3.

LBASIC is booted up by running a machine-language program that copies the BASIC ROMs to RAM and patches them. The patches are transferred to high RAM, thus leaving free all memory space ordinarily available to BASIC.

Fundamental to the operation of LBASIC is its dissociation of two memory areas ordinarily inextricably linked in BASIC: the display area than contains the visible text screen, and the output area that BASIC's screen printing commands affect. These two areas of RAM will be referred to respectively as the display field and the output field. LBASIC sets up four fields in memory, denoted as Fields 0 to 3, any one of which may at a given time be selected as the display field, the output field, or both.

LBASIC's DISPLAY and SELECT commands determine which of the four fields are the current display and output field, respectively. Each takes an argument from 0 to 3 that is the number of the desired field. Field 0 is both the output field and the display field at startup. One might, for example, in emulating a three-channel tachistoscope, have one's program first do a SELECT 1 and print a fixation point in the middle of Field 1, using BASIC's normal printing commands. This fixation point would not be visible on screen at this point, since Field 0, rather than Field 1, is the current display field. Then, one might give a SELECT 3 command and print a series of symbols in the middle of the Field 3 screen to serve as a mask. Then, a SELECT 2 command would enable the stimulus to be placed in the center of Field 2. If one broke out of the program at this point (after restoring the output field to Field 0 with a SELECT 0 command), one could view the three fields thus prepared by successively typing DISPLAY 1, DISPLAY 2, and DISPLAY 3.

For use in timing the display, there is a more general and more powerful form of the DISPLAY command. This form has the syntax "DISPLAY f_1 FOR t_1 THEN DISPLAY f_2 FOR t_2 THEN DISPLAY f_3 FOR t_3 ... THEN DISPLAY f_n." Here, the f_is represent field numbers in the range 0 to 3 and the t_is represent display durations, in units of $\frac{1}{60}$ of a second (16.67 msec). (One-sixtieth of a second is the hardware-dictated minimum unit for display durations, because it is the interval required for a single "frame" of the display, which is controlled by the vertical synchronization pulse.) Thus, for example, "DISPLAY 1 FOR 30 THEN DISPLAY 2 FOR 2 THEN DISPLAY 3 FOR 60 THEN DISPLAY 0" would show the fixation point for ½ second, then the stimulus for $\frac{1}{30}$ sec, then the mask for 1 sec before reverting to Field 0, the "standard" display. This form of the DISPLAY command has the property discussed earlier of being intepreted in its entirety before it is executed, with execution being controlled by a routine that uses the vertical synchronization interrupt for precise timing.

For measuring reaction times, LBASIC adds the TIME function to BASIC. Its syntax is Y=TIME(f), where Y could be replaced by any numeric variable name and f represents a field number. This statement causes Field f to be displayed, as the DISPLAY f command does, but in addition starts a software timer that is synchronized with the onset of the new display. This timer is halted by any keypress or by the closing of either of the joystick switches. The TIME function contains a check to ensure that all keys and joystick switches are in the released position before the new field is displayed and the timer started; execution of the command is automatically delayed until the keyboard and joystick switches are sensed to be in the neutral condition. The timer value, expressed as an integral number of milliseconds, is returned as the value of the function and stored in the variable Y. The timer counts horizontal synchronization pulses, which are approximately $1/16$ of a millisecond apart. This count is multiplied by an appropriate constant to convert to milliseconds and is then rounded to the nearest integer. The very small increase in variability resulting from this rounding process is justified by the convenience of having an integral result. The keypress or joystick switch closing does not automatically remove the designated display field from view. If the user wishes the display to revert to (say) Field 0 when a response key is pressed, the statement containing the TIME function is immediately followed by a DISPLAY 0 statement.

After the TIME function has been used, the function KEY can be employed to identify which key was pressed to halt the timer. The statement K=KEY, where K could be replaced by any numeric variable name, will set the variable K to a value corresponding to the ASCII code of the key that was pressed. KEY retains its value until the next occasion that the TIME function is used. The left and right joystick switches are signified by the KEY values 253 and 254, respectively, and either the left or right shift key produces a KEY value of 255.

LBASIC also contains a few other convenient features. The command WAIT n causes a halt in processing for $n/60$ of a second. This command is useful as a convenient way of timing the presentation rate of a study list, provided that computation time is either negligible compared with the interstimulus interval, or can be compensated for to a sufficient degree of accuracy by decreasing the WAIT argument. Note that the CoCo's built-in Extended BASIC contains a TIMER function that counts 60ths of a second by means of the vertical synchronization interrupt and that does not halt processing; for some applications, use of TIMER may be preferable to WAIT. However, a quirk in the design of the CoCo causes BASIC's TIMER function to run slightly slow while the keyboard is being polled for a keypress, on account of occasional and erratic omissions of interrupt servicing during keyboard polling. LBASIC's timing commands are not subject to this quirk.

Another useful LBASIC command is NOBREAK, which disables the break key's usual function of halting a running program. BREAK reenables the break key. In addition to the obvious advantage of preventing a subject from interrupting the experiment if he/she should accidentally hit the break key, NOBREAK has a second advantage in that it enables the break key to be used as a response key. The ability to use the break key as a response is of particular value since, in a two-choice situation, the "1" key at the upper left and the break key at the upper right of the keyboard may be the best placed keys to assign to the two responses. Furthermore, when NOBREAK is in effect, the keyboard is not polled between statements, which removes the inaccuracy in the TIMER function referred to in the preceding paragraph. A program can still be halted when NOBREAK is in effect by using the reset button, located in the back of the computer, safely out of the accidental reach of subjects.

A final point is that even though using LBASIC directly requires a knowledge of the BASIC programming language, LBASIC has the potential for allowing nonprogrammers to use its capabilities. Because LBASIC includes all the power of the ordinary BASIC language, it would be possible to write general-purpose utility programs under LBASIC for running experiments in various paradigms. Nonprogrammers would be able to make use of such utilities by specifying the parameters applicable to a particular experiment in the form of a data file that would be interpreted by the utility.

REFERENCES

GREEN, B. F., & SHWARTZ, S. P. (1978). Comparative evaluation of computer-based tachistoscopes. *Behavior Research Methods & Instrumentation*, **10**, 789-795.

MAPOU, R. L. (1982). Tachistoscopic timing on the TRS-80. *Behavior Research Methods & Instrumentation*, **14**, 534-538.

PERERA, T. B. (1978). A versatile microcomputer-based multiple-field tachistoscope. *Behavior Research Methods & Instrumentation*, **10**, 546-547.

PERERA, T. B. (1979). A comparison of BASIC language timing loops for the TRS-80 microcomputer. *Behavior Research Methods & Instrumentation*, **11**, 592.

REED, A. V. (1982). The Radio Shack color computer in the experimental psychology laboratory: An evaluation. *Behavior Research Methods & Instrumentation*, **14**, 109-112.

NOTES

1. On the newer Color Computer 3, the display window for the 40- and 80-column text screens may start on any 8-byte boundary.

2. This accuracy is relative to other measurements taken on the same Color Computer. The clock speeds of different uncalibrated Color Computers will tend to vary from one another by amounts that are typically on the order of $1/10$ of a percent. The clock speed can be calibrated by means of an adjustable capacitor located inside the machine. The nominally correct interval between horizontal synchronization pulses is $1/15.72$ msec, not $1/15.75$ msec as given in Reed (1982), since a video frame on the Color Computer consists of 262.0 scan lines, rather than 262.5 as assumed by Reed.

(Manuscript received April 16, 1987;
revision accepted for publication August 21, 1987.)