INSTRUMENTATION & TECHNIQUES

Tachistoscopic software for the Hercules display controller

GARY P. FINLEY

University of Alberta, Edmonton, Alberta, Canada

The functions necessary for a high-quality microcomputer emulation of a tachistoscope are summarized. A software package containing an extensive set of tachistoscope-emulation functions for IBM PC-type computers is described. This software gives computers equipped with the Hercules monochrome/graphics display controller the ability to produce fast tachistoscopic presentations of text material.

Under the control of appropriate software, microcomputer display screens can produce rapidly changing presentations of text material that duplicate many of the functions of the traditional electric tachistoscope. For best performance, a tachistoscope-emulation program should store a number of text samples and be able to select any of these for display very quickly. A well-designed program should also make provisions to minimize the visible effects of the scan pattern used to draw the screen image. Unlike the static pictures of an electric tachistoscope, the scanned image on a computer monitor imposes constraints on the timing of dynamic visual stimuli.

A few fundamental software functions are needed to satisfy these requirements. Rapid selection among a series of stored text samples can be made very effectively with the "virtual-screen" technique. In this approach, the video-controller board's display memory is partitioned into a number of discrete areas, each of which can store a text sample. These samples may be as large as the full screen capacity of 25 rows of 80-column text. The text stored in a virtual screen remains invisible until that section of display memory is specifically selected for screen presentation with the software tachistoscope. An adjustment of a video-controller register determines which portion of the display memory will be mapped onto the screen. This can be done very quickly; a software tachistoscope using virtual screens can switch between two text screens in less than a millisecond, even if both of them contain the maximum possible 2,000 characters.

To synchronize such display changes with the monitor's raster scan, the software tachistoscope should include a function that delays program execution until the next vertical-retrace interval. This interval is the period of time between scans, during which the monitor's video signal is disabled while the electron beam is returned to the start-

ing position. The tachistoscope software can detect the vertical-retrace condition by reading a video-controller register that contains a data bit which changes state during this interval.

In addition to the basic operations of screen switching and video synchronization, a well-designed software tachistoscope should include several other functions. The program should provide a convenient mechanism for loading text samples into the virtual screens, and it should allow the text to be located at any screen position. For precise control of the duration of tachistoscopic text presentations, the software should include a screen-blanking function. Finally, it is often desirable to eliminate the blinking text cursor maintained by the video controller, so a function should be included for this purpose.

A number of papers on the subject of tachistoscopic software for IBM PC-type computers have appeared during the last two years. Graves and Bradley (1987) described a program that uses the extra display memory on the Color Graphics Adapter (CGA) video controller as virtual screens that can be accessed with BASIC's SCREEN statement. These authors later provided (1988) an assembly-language subroutine to improve the screen-switching speed of their program. Segalowitz (1987) described an assembly-language subroutine for BASIC that allowed screen changes to be synchronized with the monitor's scan. Bühner, Sparrer, and Weitkunat (1987) used a section of the computer's main memory as a virtual screen and described software that could exchange the contents of this memory with the display memory of the CGA and Hercules display controllers. They also included assembly-language functions for video synchronization and screen blanking. Heathcote (1988) described Pascal software that provided synchronization and blanking operations for several display controllers.

Each of these papers contains parts of a completely satisfactory solution of the software-tachistoscope problem. Graves and Bradley's (1987, 1988) use of display-controller memory as virtual screens gives their tachistoscope ex-

Correspondence may be addressed to Gary P. Finley, Psychology Department, The University of Alberta, Edmonton, Alberta T6G 2E9, Canada.

cellent screen-switching speed. However, their CGA video controller has the disadvantage that the character font is based on an 8×8 matrix of pixels, resulting in low-resolution character outlines. The width of these characters can be doubled with the CGA's 40-column mode, but this does not improve the resolution. The newer Hercules and EGA/VGA controllers use faster video hardware to display text that is based on a 9×14 pixel matrix, and the resulting characters contain much more detail. Bühner, Sparrer, and Weitkunt (1987) have presented a very complete package, which includes the important functions of screen blanking and video synchronization. Unfortunately, they use computer memory instead of display-controller memory for their virtual screen storage. This means that their screen changes involve copying text from one kind of memory to another, a process that is much slower than the pointer manipulations used to select virtual screens stored in display memory. Heathcote's (1988) Pascal software has the advantage of eliminating the inconvenient assembly-language subroutines used by the other authors, and his procedures provide video synchronization and screen blanking for a variety of display controllers. However, he relies on standard Pascal routines to display text, which deprives his program of the speed advantages of the virtual-screen technique. Without virtual screens, his software tachistoscope has only one "field."

In the functions described below, I have tried to assemble a complete set of components for a software tachistoscope. The system can store text in four separate virtual screens, any one of which can be selected quickly for display. Character strings can be written to any position (specified in row and column coordinates) of any virtual screen, and each virtual screen can be individually erased. Functions are provided for video synchronization, screen blanking, and eliminating the blinking cursor. All of these functions are available to a session-controlling program written in the C or Pascal languages. No assembly language is required.

Because the selection of virtual screens is done by pointer manipulations instead of data movement, the screen-switching speed is very high, even for screens that are filled with text. The Turbo C version of the screen-selection function executes in 0.34 msec on a standard IBM PC. IBM AT computers or IBM compatibles with fast system clocks can produce even better performance. For example, an IBM AT-compatible computer with a 12-MHz clock can select a virtual screen in under 50 μ sec.

This software tachistoscope package was written as a set of C-language functions (in Turbo C, version 1.5) and also as a set of Pascal procedures (Turbo Pascal 4.0). Both of these language compilers are products of Borland International (4585 Scotts Valley Drive, Scotts Valley, CA 95066). The operations performed by the C functions and Pascal procedures in the package can be described as follows:

1. The `Hercwrite()` function loads text strings into virtual screens. The characters of a given string are copied to a specified row and column position in the area of display memory reserved for a particular virtual screen.

2. `Hercshow()` performs virtual-screen selection. The address registers of the display controller's video interface circuit are set to the memory address of a specified virtual screen, causing the screen to be displayed.

3. `Herclear()` does virtual-screen clearing. Two thousand ASCII "blank" characters are loaded into a specified virtual screen, erasing the screen's contents. Each blank character is accompanied by a "normal video" display-attribute byte.

4. `Hsyncwait()` provides synchronization with the video signal. This function waits for the next vertical retrace of the monitor's video scan. This allows the selection of virtual screens and screen blank/unblank operations to be confined to the vertical-retrace interval, when the monitor's video circuits are inactive.

5. The `Hblank()` and `Hunblank()` functions control the video signal from the display controller to the monitor. `Hblank()` switches this signal off, blanking the screen. `Hunblank()` restores normal monitor operation. These functions provide for strict control of the timing of tachistoscopic text presentations. The display-blanking function also means that it is not necessary to use one of the four virtual screens as a blank field to separate tachistoscopic text presentations.

6. Functions for cursor control: One of the virtual screens coincides with the display-memory area used by DOS and most standard software. This screen contains a blinking cursor maintained by the video controller. The `Hcursoff()` function is provided for erasing this cursor during tachistoscopic sessions. `Hcurson()` restores the cursor when it is required.

System Requirements

The high performance of this software is obtained at the cost of hardware generality. The functions described are specific to the monochrome/graphics display controller made by Hercules Computer Technology (2550 Ninth St., Berkeley, CA 94710). This controller is widely used in IBM PC-, XT-, and AT-type computers, and the software should work in any computer compatible with the PC, XT, AT, or 386, if it has a Hercules card installed. The functions also work with most of the many Hercules-compatible display controllers on the market. However, some of the newer controllers that are claimed to be completely compatible with the Hercules design contain custom large-scale integrated circuits that do not perfectly emulate the functions of the Hercules card's MC6845 video-controller chip. Such cards may not perform virtual screen changes correctly with this software.

Although these functions are hardware specific, the technique could easily be applied to other display controllers for IBM PC-compatible computers. For fast tachistoscopic operation, a display controller must contain enough surplus display memory to hold one virtual screen or more, and it must have a pointer register that can be used to select the portion of this memory that appears on the screen. For the video-synchronization and blanking functions, the controller must provide software access to video timing information, as well as control over

the video-signal output. Both the older CGA-type and the newer EGA and VGA models contain these hardware components.

Example Program

Listing 1 shows a Turbo C program that uses the package to perform a simple tachistoscopic text presentation. The program begins by blanking the display, clearing all four virtual screens, and loading each one with an identifying text string. The program then performs a loop that displays each virtual screen in turn and waits for a keyboard input before continuing the loop. All changes to the display (blanking, unblanking, and virtual-screen selections) are synchronized with the monitor's scan.

Availability

The author will provide a printed listing of the Hercules tachistoscope functions on request, or copy the files to an MS-DOS-format floppy disk (5.25 or 3.5 in.) if one

is included with the request. Write to Gary Finley at The University of Alberta Psychology Department, Edmonton, Alberta T6G 2E9, Canada, or send a message on BITNET to USERGFSO@UALTAMTS.

REFERENCES

- BÜHRER, M., SPARRER, B., & WEITKUNAT, R. (1987). Interval timing routines for the IBM PC/XT/AT microcomputer family. *Behavior Research Methods, Instruments, & Computers*, **19**, 327-334.
- GRAVES, R., & BRADLEY, R. (1987). Millisecond interval timer and auditory reaction time programs for the IBM PC. *Behavior Research Methods, Instruments, & Computers*, **19**, 30-35.
- GRAVES, R., & BRADLEY, R. (1988). More on millisecond timing and tachistoscope applications for the IBM PC. *Behavior Research Methods, Instruments, & Computers*, **20**, 408-412.
- HEATHCOTE, A. (1988). Screen control and timing routines for the IBM microcomputer family using a high-level language. *Behavior Research Methods, Instruments, & Computers*, **20**, 289-297.
- SEGALOWITZ, S. J. (1987). IBM PC Tachistoscope: Text stimuli. *Behavior Research Methods, Instruments, & Computers*, **19**, 383-388.

LISTING 1

```
/* Example program using the Hercules t-scope functions. */
#include "htscope.c" /* compile with t-scope functions */
main() /* start of main program */
{
    int I; /* integer loop counter */
    /* Blank the display and clear all four virtual screens. */
    Hsyncwait(); /* wait for the vertical sync */
    Hblank(); /* blank the display */
    Herclear(0); /* clear virtual screen 0 */
    Herclear(1); /* clear virtual screen 1 */
    Herclear(2); /* clear virtual screen 2 */
    Herclear(3); /* clear virtual screen 3 */
    /* Next, load an identifying string into each screen. */
    /* The first call of Hercwrite() below loads the string in */
    /* double quotes into virtual screen 0, starting at row 10, */
    /* column 15. The next three calls load similar strings */
    /* into the same position in screens 1, 2, and 3. */
    Hercwrite(0, 10, 15, "Screen 0: hit any key to switch to screen 1.");
    Hercwrite(1, 10, 15, "Screen 1: hit any key to switch to screen 2.");
    Hercwrite(2, 10, 15, "Screen 2: hit any key to switch to screen 3.");
    Hercwrite(3, 10, 15, "Screen 3: hit any key to return to DOS.");
    Hcuroff(); /* turn off the blinking cursor */
    Hsyncwait(); /* wait for the vertical sync */
    Hunblank(); /* enable the video signal */
    /* Now show each of the 4 screens in turn, waiting for a */
    /* keystroke before continuing to the next. */
    for(I = 0; I < 4; I++)
    {
        Hsyncwait(); /* wait for the vertical sync */
        Hercshow(I); /* display virtual screen I */
    }
```

LISTING 1 (Continued)

```

    while( kbhit() == 0 ); /* wait until a key is hit,      */
    getch();              /* then clear the key buffer  */
    }                    /* loop until 4 screens shown */

/* All 4 screens have been shown. Restore the display to */
/* the configuration used by DOS, and exit the program.   */

Hcuron();              /* restore the blinking cursor */
Herclear(0);           /* clear screen 0 (DOS screen) */
Hercshow(0);           /* switch display to screen 0  */

}                      /* end of example program    */

```

(Manuscript received December 14, 1988;
revision accepted for publication March 1, 1989.)

Announcement

30th Annual Meeting of the Psychonomic Society Atlanta, Georgia November 17-19, 1989

The 30th Annual Meeting of the Psychonomic Society will be held in Atlanta, Georgia, November 17-19, 1989. The meetings will begin Friday morning and continue until Sunday at noon. The headquarters hotel will be the Hyatt Regency.

The Call for Papers was mailed to members and associates in April; the program and hotel reservation cards will be mailed to them in September. A copy of the program will be published in the November issue of the *Bulletin of the Psychonomic Society*.

For further information, please contact the secretary-treasurer of the Society: Michael E. Rashotte, Department of Psychology, Florida State University, Tallahassee, FL 32306-1051 (Telephone: 904-644-2040; BIT-NET: Rashot@FSU).