

The PC tachistoscope has 32 pages

BRETT MYORS

Macquarie University, Sydney, New South Wales, Australia

Page switching is the basis of the PC tachistoscope, and usual accounts of page switching on the PC refer to a maximum of 8 available pages. This article shows how to achieve up to 16 pages in 40-column text mode and up to 32 pages in low-resolution graphics mode using conventional page switching combined with the bit-plane layering technique described by Bokhorst (1995). A high level C program is presented to show how this is done. When 32 pages are in use, the images must be monochrome.

Segalowitz (1987) was one of the first authors to describe how the PC could be used for tachistoscopic presentation of stimuli by preloading images into various pages of video memory and rapidly switching between them. In this way, images did not need to be drawn in real time and could therefore be displayed at the maximum possible rate simply by instructing the video hardware to switch to a new image stored at a different location in memory. The maximum display rate is governed by the screen refresh rate of the video monitor—which is typically around 60–70 Hz, although it can vary anywhere between 40 and 90 Hz (Hausmann, 1992). EGA and VGA video standards provide well-documented support for 8 pages in text modes and up to 8 pages in graphics modes, and many authors have described how to access these (see Hausmann, 1992, for a review). However, the number of available pages can be substantially increased relatively easily, thus increasing the capacity and usefulness of the PC tachistoscope. The purpose of this article is to show how the maximum number of available pages can be increased to 16 pages in text mode and 32 in graphics mode.

THE MODE 01h TACHISTOSCOPE

Mode 01h is a 40-column \times 25-line text mode available on the original IBM PC and supported in all subsequent models; thus, it is universally available on PCs with at least CGA capabilities. It is well known that mode 01h supports 8 separate text screens, but because of its low memory requirements, this limit can be easily doubled. In text modes, a screen of display memory comprises many 2-byte character/attribute pairs, and only 1,000 of these are required per screen in mode 01h. Table 1 shows the starting addresses of the pages available in mode 01h. Characters and attributes can be written directly to video memory at appropriate offsets from these locations and subsequently displayed simply by changing the start address registers of the CRT Controller. Modifying the start address

registers changes the origin of the next screen update and hence changes the display. The lines indicated by two asterisks in the `set_page()` routine in Listing 1 show the C code for making these changes. Accessing the CRT Controller directly allows the number of available screens for display to be increased from 8 to 16.

THE MODE 0dh TACHISTOSCOPE

Even more pages are available with the mode 0dh tachistoscope. Mode 0dh is a graphics mode with 320×200 pixel resolution, introduced with the EGA and upwardly compatible to the VGA. Mode 0dh is a planar mode with display memory organized into four bit-planes used to access any one of 16 possible colors. Images are therefore spread across the four planes, resulting in a somewhat complex method of addressing a single pixel (see Farraro, 1994, or Wilton, 1994, for details). Because of its comparatively low resolution, mode 0dh also supports 8 conventional screen pages; however, if the bit-plane layering technique described by Bokhorst (1995) is used in conjunction with the usual page switching described above, the number of available pages is effectively quadrupled. Thus there are 8 conventional display pages, each of which comprises four separate images stored in each bit-plane. Any of these images can be individually accessed and displayed. Although the resolution of mode 0dh is low, it is quite effective for simple graphics and large text.

Listing 1 shows a C program that demonstrates the 32-page mode 0dh tachistoscope. The program uses Microsoft C graphics routines to change video modes and fills each page with an ASCII character, starting with a capital A on page 0 and ending with a single quote on page 31. Character strings are written by directly accessing bitmaps for the 8×8 character set stored in ROM. Fortunately these bitmaps are compatible with planar memory modes and make text drawing relatively easy. The user is then able to switch between pages with the up or down arrow keys.

When the user decides to exit, the program automatically performs 1,000,000 page switches and prints out the elapsed time. Times are measured by the standard `clock()` function, which uses the on-board time-of-day clock. As

Correspondence concerning this article should be addressed to B. Myors, School of Behavioural Sciences, Macquarie University, Sydney, NSW 2109, Australia (e-mail: bmyors@bunyip.bhs.mq.edu.au).

Table 1
Screen Pages in Mode 01h

Page	Start Address
0	b8000000h
1	b8000800h
2	b8001000h
3	b8001800h
4	b8002000h
5	b8002800h
6	b8003000h
7	b8003800h
8	b8004000h
9	b8004800h
10	b8005000h
11	b8005800h
12	b8006000h
13	b8006800h
14	b8007000h
15	b8007800h

Note—The first 8 pages are well documented but the subsequent 8 pages are also usually available.

such, `clock()` is a low-resolution timer with a period of only 54.9 msec—hence the use of 1,000,000 trials to gain adequate accuracy.

The `enable_plane()` function shows how the Map Mask Register of the VGA Sequencer can be set to allow only one plane to be written to at a time. Page switching is performed by the `set_page()` function, which shows how the Color Plane Enable Register of the Attribute Controller can be used to select an individual bit-plane for display. Use of the start address registers of the CRT Controller for conventional page switching has already been described. The function `sync()` shows how to lock onto the vertical retrace via the Input Status Register. Locking onto the vertical retrace ensures that screen changes occur only while the electron guns in the monitor are moving from the bottom right-hand corner of the screen to the top left-hand corner and is essential for tachistoscopic applications and smooth screen transitions. Ideally the code for `sync()` would be incorporated into `set_page()`, to avoid an unnecessary function call, but it was kept separate so that `set_page()` could be accurately timed.

The program shown in Listing 1 provides all the relevant display routines for tachistoscopic presentation of text stimuli, including how to display a text string at an arbitrary location on the screen. Resolution of the 8×8 character set is surprisingly good in this mode. One likely additional requirement, not shown, is the setting of VGA color registers 1, 2, 4, and 8 to the same color. As it stands, the demonstration program leaves the default colors unchanged, so that screens involving plane 1 are shown in blue, screens involving plane 2 are shown in green, screens involving plane 4 are shown in red, and screens involving plane 8 are displayed in dark gray. The color registers can easily be altered with a BIOS call. If graphical stimuli such as simple geometric shapes are required, new functions will need to be written in order to load them appropriately.

One limitation of the current scheme is that the images must all be monochrome because only one bit-plane is used for each image. Multicolor images could be set up by utilizing more than one bit-plane per image, but this would reduce the number of planes available for layering and hence the number of display pages.

DISCUSSION

Bokhorst (1995) provided assembly language routines for bit-plane layering in modes 10h and 12h. Mode 12h does not support any conventional pages at all, and mode 10h only supports two, giving a maximum of 8 pages available when layering is used. In contrast, the techniques described here support considerably more than this.

Results of the timing of the `set_page()` function in Listing 1 were periods of 5.44 μ sec for a single page switch on a Pentium 100 computer with an STB Powergraph 64 video accelerator card. This is an order of magnitude less than the retrace interval of 60 μ sec (Sargent & Shoemaker, 1995) and indicates that this computer is suitable for tachistoscopic operations using the methods described here. Comparable results were obtained on other computers tested.

Areas of substantive interest where the techniques described might prove useful include the rapid sequential visual presentation (RSVP) paradigm (Forster, 1970; Potter, 1993) used to study very short-term conceptual memory, and the inspection time paradigm used in cognitive abilities studies (Nettlebeck, 1987). In particular, the RSVP paradigm involves a large number of images. For example, discovery of the attentional blink (Raymond, Shapiro, & Arnell, 1992) involved up to 24 images, well in excess of the conventional PC tachistoscope.

Of course, if a new image can be written to display memory inside the duration of a single vertical retrace, the number of pages available to the PC tachistoscope is effectively infinite. However, the retrace interval is extremely short. Precise timing with the mode 2 timer (Graves & Bradley, 1991) showed that it takes about 0.2 msec to completely fill the mode 01h screen on the Pentium 100 PC.¹ Thus, only about 30% of the image can be drawn within the retrace interval, and it is unlikely that any other video mode will improve on the performance of this low-resolution 40×25 line text mode. We are left with the conventional method of preloading images into display memory and switching between them.

As a matter of caution, it is recommended that tachistoscopic and other precision timing operations be carried out in the native DOS environment only, since only old-fashioned DOS guarantees complete, uninterrupted control of the PC hardware at the lowest level. Windows, Windows '95, or any other multitasking environment should be avoided.

This article shows how to implement conventional page switching in conjunction with bit-plane layering on the IBM PC in the high-level C programming language. As

such, it is hoped that these techniques will extend the research possibilities available to PC users.

Availability

Copies of the program can be obtained by e-mail to the author at bmyors@bunyip.bhs.mq.edu.au.

REFERENCES

- BOKHORST, F. D. (1995). Bit-plane layering for high-resolution EGA and VGA graphics on the IBM PC/XT/AT. *Behavior Research Methods, Instruments, & Computers*, **27**, 496-501.
- FARRARO, R. (1994). *Programmer's Guide to the EGA, VGA, and Super VGA Cards* (3rd ed.). Reading, MA: Addison-Wesley.
- FORSTER, K. I. (1970). Visual perception of rapidly presented word sequences of varying complexity. *Perception & Psychophysics*, **8**, 215-221.
- GRAVES, R. E., & BRADLEY, R. (1991). Millisecond timing on the IBM PC/AT/XT and PS/2: A review of the options and corrections for the Graves and Bradley algorithm. *Behavior Research Methods, Instruments, & Computers*, **23**, 377-379.
- HAUSSMANN, R. E. (1992). Tachistoscopic presentation and millisecond timing on the IBM PC/XT/AT and PS/2: A Turbo Pascal unit to provide general-purpose routines for CGA, Hercules, EGA, and VGA monitors. *Behavior Research Methods, Instruments, & Computers*, **24**, 303-310.
- NETTLEBECK, T. (1987). Inspection time and intelligence. In P. A. Vernon (Ed.), *Speed of information-processing and intelligence* (pp. 295-346). Norwood, NJ: Ablex.
- POTTER, M. C. (1993). Very short-term conceptual memory. *Memory & Cognition*, **21**, 156-161.
- RAYMOND, J. E., SHAPIRO, K. L., & ARNELL, K. M. (1992). Temporary suppression of visual processing in an RSVP task: An attentional blink? *Journal of Experimental Psychology: Human Perception & Performance*, **18**, 849-860.
- SARGENT, M., & SHOEMAKER, R. L. (1995). *The personal computer from the inside out* (3rd ed.). Reading, MA: Addison-Wesley.
- SEGALOWITZ, S. J. (1987). IBM PC tachistoscope: Text stimuli. *Behavior Research Methods, Instruments, & Computers*, **19**, 383-388.
- WILTON, R. (1994). *Programmer's guide to PC video systems* (2nd ed.). Redmond, WA: Microsoft Press.

NOTE

1. In this case *mode 2* refers to programming counter 0 of the 8253 Programmable Interval Timer to mode 2, not to a video mode. The mode 2 timer has a period of 838.1 nsec.

LISTING

Demonstration of the 32-Page Mode 0dh Tachistoscope

```
// The PC tachistoscope has 32 pages.
#include <stdio.h>
#include <conio.h>
#include <dos.h>
#include <time.h>
#include <graph.h>

#define DISPLAY_WIDTH          320
#define PAGE_SIZE              8192
#define ROM_8x8_FONT          0xf000fa6e

#define ESCAPE                 27
#define UP                     72
#define DOWN                   80

// VGA control
#define INPUT_STATUS_REGISTER  0x3da
#define VSYNC_MASK            0x08
#define ATTRIBUTE_CONTROLLER   0x3c0
#define COLOUR_PLANE_ENABLE   0x12
#define RESET_ATTRIBUTE        0x20
#define CRT_CONTROLLER         0x3d4
#define CRT_ADDRESS_HI        0x0c
#define CRT_ADDRESS_LO        0x0d
#define SEQUENCER              0x3c4
#define MAP_MASK               0x02

// PROTOTYPES
void sync (void);
void set_page (int page);
void enable_plane (int bitplane);
void print_char (int row, int col, char thechar, char _far *vb);
void print_string (int row, int col, char *str, char _far *vb);

char _far *video = (char _far *) 0xa0000000; // Graphics mode video buffer

void main (void)
```

LISTING (Continued)

```

{
    char buf[42], _far *vb;
    int i, page, ch;
    long l;
    clock_t tm;
    double dtm;

    _setvideomode(_MRES16COLOR);

    // Set up tachistoscopic display
    buf[40] = '\0';
    for (page = 0; page < 32; ++page) {
        enable_plane(1 << (page % 4));
        vb = video + (page >> 2) * PAGE_SIZE;

        for (i = 0; i < 40; ++i)
            buf[i] = 'A' + page;
        for (i = 0; i < 25; ++i)
            print_string(i, 0, buf, vb);

        sprintf(buf, "PC Tachistoscope page %2d.", page);
        print_string(8, 6, buf, vb);
        print_string(10, 7, "Press up or down arrow.", vb);
        print_string(12, 10, "Press ESC to quit.", vb);
    }

    // User control
    sync();
    set_page(page = 0);
    while ((ch = getch()) != ESCAPE) {
        switch (ch) {
            case UP : page = (page + 31) % 32; break;
            case DOWN : page = (page + 1) % 32; break;
            default : continue;
        }
        sync();
        set_page(page);
    }

    // Rapid switching
    tm = clock();
    for (l = 0; l < 1000000; ++l)
        set_page(l % 32);
    tm = clock() - tm;
    dtm = (((double) tm) / 1000000.0) / ((double) CLOCKS_PER_SEC);

    _setvideomode(_DEFAULTMODE);
    printf("One million page switches occurred in %.1f seconds.\n"
        "One page switch occurred in %lf milliseconds (%.1lfHz).\n"
        "One vertical retrace takes 0.06 milliseconds.\n",
        ((float) tm) / ((float) CLOCKS_PER_SEC),
        dtm * 1000.0, 1.0 / dtm);
}

void sync (void)
{
    // Wait for vertical retrace
    while ( inp(INPUT_STATUS_REGISTER) & VSYNC_MASK);
    while (!(inp(INPUT_STATUS_REGISTER) & VSYNC_MASK));
}

```

LISTING (Continued)

```

void set_page (int page)
{
    int start_address;

    start_address = video + (page >> 2) * PAGE_SIZE;

    _disable();

    // Set bit plane
    outp(ATTRIBUTE_CONTROLLER, COLOUR_PLANE_ENABLE);
    outp(ATTRIBUTE_CONTROLLER, 1 << (page % 4));
    outp(ATTRIBUTE_CONTROLLER, RESET_ATTRIBUTE);

    // Set video page
    outpw(CRT_CONTROLLER, (start_address << 8) | CRT_ADDRESS_LO); // **
    outpw(CRT_CONTROLLER, start_address | CRT_ADDRESS_HI); // **

    _enable();
}

void enable_plane (int bitplane)
{
    outpw(SEQUENCER, (bitplane << 8) | MAP_MASK);
}

void print_char (int row, int col, char thechar, char _far *vb)
{
    int x;
    char _far *ch;

    ch = ROM_8x8_FONT + (thear << 3); // Get character set
    x = row * DISPLAY_WIDTH + col; // Set character position

    // Write character
    for (row = 0; row < 8; ++row) {
        vb[x] = *ch++;
        x += DISPLAY_WIDTH >> 3;
    }
}

void print_string (int row, int col, char *str, char _far *vb)
{
    while (*str)
        print_char(row, col++, *str++, vb);
}

```

(Manuscript received January 15, 1997;
revision accepted for publication April 8, 1997.)