TIPS 'N TECHNIQUES

Applesoft Windows

by Michael A. Seeds, Franklin and Marshall College, P.O. Box 3003, Lancaster, PA 17604

DOS 3.3 | ProDOS

The Apple's text display is a window that you can control from within your program. These short programs show you how it's done in both Applesoft and machine language.

Your Apple Monitor contains a window, and looking through that window can solve a few bothersome programming problems. For example, I like to jump around when I am editing a program, and sometimes I need to copy parts of one section into another section. I've often wished I could run two video monitors side by side — one to display the program and one to display my working area. Another problem is displaying short messages without disturbing text already on the screen.

We can solve problems like these using the Apple text window. Normally the window is set to fill the entire video screen, but your can change the boundaries of the text window by POKEing locations 32 to 35. A POKE 32,10, for instance, sets the left margin of the text window to the tenth space from the left. You can try this in immediate mode, if you like. Try these locations:

Location	Function	Limits
32	Left edge	0-39
33	Width of window	1-40
34	Top edge	0-23
35	Bottom edge	0-23

(For more details, see page 31 of the *Apple II Reference Manual* or Appendix F of the; Applesoft BASIC Programmer's Reference Manual.)

When you change the text window, your Apple uses the new area and ignores anything outside it. The HOME command clears just the window and places the cursor at the upper left corner. If you list a program, the text window will scroll as usual, but text outside the window will be left untouched. HTAB will move the cursor relative to the newly defined left edge, but VTAB will allow you to move the cursor above or below the existing text window. Go ahead and try a few POKEs. No matter how much you mess up the window boundaries, you can restore the window to full screen with a TEXT command.

PROGRAM CONTROL OF WINDOWS

Using these same locations, programs can very easily control the window boundaries. The following two programs present two possibilities. The first, BIWIND, gives you two video work areas for program development and testing, while the second, WINDER, demonstrates a technique for creating Macintosh-like "dialog" windows.

BIWIND is a short machine language program that allows you to divide your video screen into a top screen and a bottom screen. You can work in either half without disturbing the other half. With BIWIND installed, you can enter the top half of the screen by moving the cursor to the bottom half and typing CALL 771. When you press <RETURN> the screen will divide in half and you will be working in the top half. You can LIST, EDIT, and RUN programs here without disturbing the text below (unless, of course, your program alters locations 32 to 35 or uses a TEXT command). To enter the bottom half of the screen, move the cursor to the top half and type CALL 794. Press <RETURN> and the bottom window will open for your use. To return to the full screen, just type TEXT.

The program will always divide the screen into two equal areas unless you specify otherwise. To divide the screen at the nth line, just POKE 770, n. The next time you open one of the work areas, the new dividing line will be in effect.

ENTERING THE PROGRAM

To key in BIWIND, enter the machine language code shown in **Listing 1** and save it on disk with the command:

BSAVE BIWIND,A$300,L$40

For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" in the beginning of this issue.

HOW THE PROGRAM WORKS

This machine language program is really two separate routines. The first opens the top of the screen, while the second opens the bottom of the screen. Let's look at the first routine. When we CALL 771, the program saves the present location of the cursor for use when we flip back to the bottom half of the screen, and then it loads in the last location of the cursor in the top half of the screen. The Apple always stores the current cursor location in $25. Next, the program sets the top of the working area to zero and

"The subroutines can be used in any BASIC program to open a small window in a text screen display."

it sets the bottom line to the contents of $302. Finally, it calls the subroutine at $334 to draw a line of equal signs dividing the two screen areas.

The second routine opens the bottom of the screen. This routine is very similar to the first. The major difference is that it sets the top of the screen to the contents of $302 plus 1. This prevents it from overwriting the dividing line of equal signs.

BIWIND is a simple program, and, because of the way it remembers the last cursor position, it can get confused if you try to open the half of the screen in which you are already working. If that happens, try typing HOME. If things get hopelessly confused, just type TEXT and you will be back to a full screen.

I find BIWIND especially useful for editing my programs. For example, I can jump to the top screen to list one segment of the program, and then jump back to the bottom screen to edit a related part of the program.

The program WINDER (**Listing 2**) also uses the text window, but for a different purpose. The subroutines starting at **line 390** can be used in any BASIC program to open a small window in a text screen display. I use this to display messages to the user. The subroutine at **line 560** closes the temporary message window and restores the original data. The first part of the program in **Listing 2** is just a demonstration of the methods.

To open a window, the main program must set the quantities WL, WT, WW, and WB. These are the four numbers to be POKEd into locations 32 to 35, and they define the location and size of the window. To allow room for a border, WW and WB must be greater than two. Of course, the boundaries of the window must not go beyond the boundaries of the video screen.

Entering the program

To key in WINDER, simply enter the Applesoft program shown in **Listing 2** and save it on disk with the command:

SAVE WINDER

If you decide to use these subroutines in your own programs, notice that the variables they use all begin with a W. Avoid using W variables in the rest of your program so that the window subroutines won't interfere with them. Notice, also, that your main program must dimension WS$(24).

If you follow these few rules, it's simple to open windows into your computer.

LISTING 1: BIWIND

```
                         ;
                         ;
                         ; BIWIND
                         ; BY MIKE SEEDS
                         ; COPYRIGHT (C) 1985
                         ; BY MICROSPARC, INC.
                         ; CONCORD, MA Ø1742
                         ;
                         ; MICROSPARC ASSEMBLER
                         ;
                                 ORG $3ØØ
                         VCURS   EQU $25       ;VERTICAL CURSOR POSITION
                         BOTSCR  EQU $23       ;BOTTOM OF TEXT WINDOW
                         TOPSCR  EQU $22       ;TOP EDGE OF WINDOW
Ø3ØØ  ØB                 TOP     DFC 11        ;TOP CURSOR POSITION
Ø3Ø1  17                 BOT     DFC 23        ;BOTTOM CURSOR POSITION
Ø3Ø2  ØC                 LINE    DFC 12        ;DIVIDING LINE SET AT 12
Ø3Ø3  A5 25                      LDA VCURS     ;*** OPEN TOP ***
Ø3Ø5  8D Ø1 Ø3                   STA BOT       ;SAVE BOTTOM CURSOR POSITION
Ø3Ø8  AD ØØ Ø3                   LDA TOP       ;
Ø3ØB  85 25                      STA VCURS     ;SET TOP CURSOR POSITION
Ø3ØD  A9 ØØ                      LDA #$Ø
Ø3ØF  85 22                      STA TOPSCR    ;SET TOP OF AREA
Ø311  AD Ø2 Ø3                   LDA LINE      ;GET DIVIDING LINE
Ø314  85 23                      STA BOTSCR    ;SET BOTTOM OF AREA
Ø316  2Ø 34 Ø3                   JSR DIVIDE    ;DRAW DIVIDING LINE
Ø319  6Ø                         RTS           ;END OF OPEN TOP ROUTINE

Ø31A  A5 25                      LDA VCURS     ;*** OPEN BOTTOM ***
Ø31C  8D ØØ Ø3                   STA TOP       ;SAVE TOP CURSOR POSITION
Ø31F  AD Ø1 Ø3                   LDA BOT
Ø322  85 25                      STA VCURS     ;SET BOTTOM CURSOR POSITION
Ø324  AD Ø2 Ø3                   LDA LINE      ;GET DIVIDING LINE
Ø327  18                         CLC
Ø328  69 Ø1                      ADC #1        ;ADD ONE
Ø32A  85 22                      STA TOPSCR    ;SET TOP OF AREA
Ø32C  A9 18                      LDA #$18      ;DECIMAL 24
Ø32E  85 23                      STA BOTSCR    ;SET BOTTOM OF AREA
Ø33Ø  2Ø 34 Ø3                   JSR DIVIDE    ;DRAW DIVIDING LINE
Ø333  6Ø                         RTS           ;END OF OPEN BOTTOM ROUTINE
                         ;
Ø334  AD Ø2 Ø3           DIVIDE  LDA LINE      ;GET LINE POSITION
Ø337  2Ø 24 FC                   JSR $FC24     ;VTAB TO DIVIDING LINE
Ø33A  A2 27                      LDX #39       ;PRINT 39 SYMBOLS
Ø33C  A9 BD                      LDA #$BD      ;= SIGN
Ø33E  2Ø FØ FD           OUT     JSR $FDFØ     ;PRINT A SYMBOL
Ø341  CA                         DEX
Ø342  DØ FA                      BNE OUT       ;LAST SYMBOL?
Ø344  6Ø                         RTS           ;DONE

ØØØ  ERRORS

Ø3ØØ  HEX START OF OBJECT
Ø344  HEX END OF OBJECT
ØØ45  HEX LENGTH OF OBJECT
95C3  HEX END OF SYMBOLS
```

END OF LISTING 1

LISTING 2: WINDER

```
1Ø   REM  **************************
2Ø   REM  *        WINDER          *
3Ø   REM  *     BY MIKE SEEDS      *
4Ø   REM  *  COPYRIGHT (C) 1985    *
5Ø   REM  *  BY MICROSPARC, INC    *
6Ø   REM  *  CONCORD, MA. Ø1742    *
7Ø   REM  **************************
8Ø   DIM WS$(24)
9Ø   HOME : PRINT : PRINT  TAB( 9)"FOR WHOM TH
      E BELL BONGS"
1ØØ   PRINT : PRINT  TAB( 15)"BY A. MONKEY": PRINT
      : PRINT
11Ø   FOR J = 1 TO 8
12Ø   FOR K = 1 TO 4Ø: PRINT  CHR$ (64 + 26 *
       RND (1));: NEXT K: PRINT
13Ø   NEXT J
14Ø   VTAB 23: PRINT "PRESS ANY KEY TO HALT."
15Ø WL = 12:WT = 1Ø:WW = 1Ø:WB = 5: GOSUB 39Ø
      : REM  OPEN WINDOW
16Ø   PRINT : PRINT "GOT IT?"
17Ø   GOSUB 35Ø: REM  DELAY
18Ø   GOSUB 56Ø: REM  CLOSE WINDOW
19Ø   IF  PEEK (49152) > 128 THEN  TEXT : HOME
      : END
2ØØ WL = 5:WT = 1:WW = 25:WB = 7
21Ø   GOSUB 39Ø: REM  OPEN WINDOW
22Ø   VTAB WT + 2: HTAB 4: PRINT "NOTICE THE T
      EXT IS": HTAB 4: PRINT "RESTORED CORRECT
      LY."
23Ø   GOSUB 35Ø: REM  DELAY
24Ø   GOSUB 56Ø: REM   CLOSE WINDOW
25Ø   IF  PEEK (49152) > 128 THEN  TEXT : HOME
      : END
26Ø WT = 1Ø:WB = 1Ø: GOSUB 39Ø
27Ø   FOR J = 1 TO 25: PRINT "  ";J,J * J: NEXT
      J
28Ø   PRINT : PRINT "SCROLLING IS AUTOMATIC"
29Ø   GOSUB 35Ø: GOSUB 56Ø
3ØØ   IF  PEEK (49152) > 128 THEN  TEXT : HOME
      : END
31Ø   GOTO 15Ø
32Ø   REM  ==========
33Ø   REM    DELAY
34Ø   REM  ==========
35Ø   FOR J = 1 TO 15ØØ: NEXT : RETURN
36Ø   REM  ===================
37Ø   REM   SUBROUTINE WINDOW
38Ø   REM  ===================
39Ø WA = 1Ø24 + 128 * (WT - 1 - 8 *  INT ((WT
      - 1) / 8)) + 4Ø *  INT (WT / 8.5)
4ØØ WS = WA
41Ø   FOR WJ = WT TO WT + WB - 1:WS$(WJ) = ""
42Ø   FOR WK = 1 TO WW:WS$(WJ) = WS$(WJ) +  CHR$
      ( PEEK (WA + WL + WK - 1)): NEXT WK
43Ø   POKE WA + WL,32: POKE WA + WL + WW - 1,3
      2
44Ø WA = WA + 128: IF WA = 2Ø88 THEN WA = 11Ø
      4
45Ø   IF WA = 2Ø48 THEN WA = 1Ø64
46Ø   NEXT WJ
47Ø   FOR WJ = 1 TO WW: POKE WS + WL + WJ - 1,
      32: POKE WA - 128 + 984 * (WA = 1Ø64 OR
      WA = 11Ø4) + WL + WJ - 1,32: NEXT WJ
48Ø   REM  SET TEXT SCREEN
49Ø   POKE 32,WL + 1: POKE 33,WW - 2
5ØØ   POKE 34,WT: POKE 35,WT + WB - 2
51Ø   HOME
52Ø   RETURN
53Ø   REM  ================
54Ø   REM  SUBROUTINE CLOSE
55Ø   REM  ================
56Ø   POKE 32,Ø: POKE 33,4Ø
57Ø   POKE 34,Ø: POKE 35,24
58Ø   FOR WJ = WT TO WT + WB - 1: VTAB WJ: HTAB
      WL + 1: PRINT WS$(WJ): NEXT WJ
59Ø   RETURN
```

END OF LISTING 2