SECOND FEATURE

WINDOW MAGIC

DOS 3.3 O 0

Now you can have Mac-like windows on your Apple II Plus, //e, or //c. Windows can be moved on and off the screen, and on top of each other. Enhanced print and input routines stay within the current window boundaries.

ProDOS O 0

by Paul Nick, 31 Rittner Lane, Old Bridge, NJ 08857

The concept of windowing has a special fascination for me. Windowing removes the limitation of a single screen and allows "miniature screens" to direct attention where it is needed. Windowing also allows creation of virtual screens that can be larger than the actual screen display. To bring this capability to the Apple II series, I created Window Magic.

Window Magic lets you add windows easily to any Applesoft program. To create Window Magic, I essentially redesigned the input/output (I/O) routines to the screen and keyboard. Then I wrote an interface to BASIC consisting of ten ampersand commands. You can use Window Magic to define as many as 255 windows of any size. You can overlap them, move them around (even off the display screen), direct I/O to them, and delete them.

USING WINDOW MAGIC

To use Window Magic, you simply BRUN WINDOW.MAGIC (**Listing 1**) at the beginning of an Applesoft program and then use a set of ten ampersand (&) commands to define windows, move them around the screen, and direct input and output from within a specific window. Two demonstration programs (**Listings 2 and 3**) have been provided to show you how these commands are used. In the following descriptions of the functions of the ampersand statements, note that A, B, C, D, and E represent numbers, numeric variables or expressions, while A$ represents a string variable.

When you first BRUN WINDOW .MAGIC, a window the size of the screen is automatically defined. This window forms the backdrop for all other windows and thus may not be deleted or moved. Every window has a number. The backdrop window is window 0. The first window you define is window 1, the second window you define is window 2, and so on. The larger the window's number, the higher the priority of the window. This means that if two windows overlap, the lower priority window will be partially (or even completely) hidden under the higher priority window. Note, also, that whenever you define a window, it is turned on for subsequent input/output as if you had performed an & ON command.

THE AMPERSAND COMMANDS

& DEF A,B,C,D,E — This statement is used to define a window. A and B are the horizontal and vertical positions, respectively, of the upper-left corner of the window. The allowable ranges for A and B are 0-255. It is perfectly acceptable to define a window off the screen. You can move it onto the screen later in your program. C and D are the width and height, respectively, of the window. The ranges of C and D are 3-96. Note that one can define a window larger than the screen, but this consumes extra memory.

E is the screen code for the character that will make up the window's border. The screen codes vary with Apple models, and depend on whether the 80-column card is enabled. For example, to define a window with a border of inverse spaces, use:

```
& DEF A,B,C,D,32
```

if you do not want a border at all, let E=0. The range of E is 0-255.

& ON A — This command turns on window A for input/output. The backdrop window is window 0 and any windows that you define will be numbered starting with one. Note that this program keeps track of the cursor position for each window, so that when you turn on a window, the cursor is where it was when it was turned off.

& DEL — This command deletes the window that is currently on. The number of windows higher than this will then be decreased. For instance, if you have three windows and delete window 2, then window 3 becomes window 2. The current window is still number 2. If you try to delete a window that hasn't been defined, you will get a SYNTAX ERROR.

& HOME — This statement clears the window that is currently on and homes the cursor.

& VTAB A — This command tabs vertically to a position relative to the top-left corner of the window that is currently on. An ILLEGAL QUANTITY error occurs if you attempt to VTAB beyond the limits of the current window.

& HTAB A — This command tabs horizontally to a relative position within the

window in a manner similar to that of the & VTAB statement. *Note:* Using a border decreases the effective size of a window by two in each dimension since the first and last row, and the first and last column in the window, are occupied by the border character. Both & VTAB and & HTAB ignore the border rows and columns.

& PRINT — This command functions like a regular PRINT statement except that the following are not supported: TAB(*x*), SPC(*x*), and commas (used to partition the screen into tab columns). Instead, Window Magic offers four other printing commands (embedded in the PRINT statement) that are implemented the same way. The "at" sign (@) executes a carriage return, allowing multiple lines to be printed with one PRINT statement. The symbols #, $, and % set character display to normal, inverse, and flashing, respectively. At the end of an & PRINT statement, they prevent a carriage return just as a semicolon does. For example, the statement:

```
& PRINT %"H"#"ELLO"@$
"THERE";
```

prints H in flashing video and the rest of HELLO in normal video. It then performs a carriage return and prints THERE in inverse. It does not perform a carriage return after printing THERE because of the semicolon. Note that on the //c and enhanced //e you may get Mousetext instead of certain inverse characters. Also the 80-column firmware does not support flashing characters.

Window Magic also supports the usual INVERSE, NORMAL, and FLASH statements. Note that control characters are always printed in inverse.

& INPUT A$,A,B — Window input is a vast improvement over regular input. It allows entry of commas and lower-case letters, employs several editing features, allows you to set a maximum field length, displays control characters as inverse, and allows a special number entry mode. A is the user-defined input mode, and B is the maximum desired length of the string. The input modes are explained in detail later.

& GOTO A,B — This statement moves the currently active window to horizontal and vertical coordinates A,B within the range 0-255. A window will wrap around as its position approaches 255. For instance, if a window is at position 0,255, all except the top row will appear on the screen because of this wraparound effect.

& GOTO can be used to produce some very interesting effects. The relative cursor position of each window is preserved, so it is possible to print to a window while it is moving (or give the appearance thereof) by printing and moving in a loop. You can define a window off-screen, print a large amount of information on it and then move it on-screen so that all the information appears instantly.

& RESTORE — If for some reason the screen is destroyed, this statement restores it by reprinting all windows to the screen. You can use it to switch back and forth between using Window Magic and the Apple's normal screen output routines.

FEATURES OF & INPUT

Moving the Cursor

All four arrows on the Apple //e and //c work with this program. On the Apple II Plus, <CTRL>K and <CTRL>J move the cursor up and down. However, the up and down arrows will not go beyond the limit of the typed string in either direction. <CTRL>B moves the cursor to the beginning of the string, and <CTRL>N moves it to the end of the string.

Special Editing Features

Whenever & INPUT is invoked, the string is moved from memory into the input buffer and onto the screen for editing (a provision to prevent this will be explained later). <CTRL>D or DELETE on the Apple //e or //c deletes the character under the cursor. <CTRL>I or the <TAB> key on the //e and //c inserts a space at the cursor. If you are in a numeric mode, <CTRL>I inserts a zero at the cursor.

Terminating Input

Pressing <RETURN> accepts all input regardless of where the cursor is at the time. <CTRL>Q accepts input only up to the cursor's location. Characters from the cursor to the end of the field are discarded. Normal Applesoft input statements always perform a carriage return when <RETURN> is pressed. With the new input routine, using a semicolon instead of a comma after the string name prevents this:

```
& INPUT A$;5,50
```

It is then possible to input on the last line of a window without causing the window to scroll.

Control Characters

Any control character you type that is not one of the commands mentioned above is entered into the string and displayed in inverse video. You can also enter control characters that are commands by pressing <CTRL>O first. In this way, it is possible to enter backspaces, carriage returns, and so on, directly into the string. Any control characters that are printed with an & PRINT statement are also printed in inverse.

Lower-case and Special Characters

Apple II Plus owners with lower-case adapters can use <ESC> to shift in and out of lower-case. & INPUT starts out in upper-case lock mode. Pressing <ESC> once enters lower-case mode. Once in lower-case mode, pressing <ESC> once shifts to

TABLE 1: Modes for the & INPUT Command

Modes	Characters Allowed
0, 9	All characters are allowed. Control characters are shown in inverse. Mode 9 and all higher modes do not read the string in from memory.
1, 10	Lower-case characters are rejected.
2, 11	Control characters are rejected.
3, 12	Lower-case and control characters are rejected.
4, 13	Digits only are allowed.
5, 14	Digits and leading + or − are allowed.
6, 15	Digits, plus leading + or −, and the decimal point are allowed.
7	Same as Applesoft GET. The third parameter specifies the ASCII value of the character to wait for; if zero, any character is allowed.
8	Input from text files. Commas, colons, and semicolons are allowed. See note in text regarding cursor positioning.

upper-case for the following character only. Pressing <ESC> twice in sequence locks in upper-case. Apple //e and //c owners should stay in upper-case lock and use the normal shifting procedure. Also, several characters that are unavailable on the II Plus keyboard can be entered with this program. <CTRL>Z yields a left bracket, <CTRL>X yields a backslash, and <CTRL>C yields an underline character.

User-Defined Length

You can set a limit on the length of the string using this feature. But if the length you set is too long to fit in the window, you will get an ILLEGAL QUANTITY error. If you set the length at zero, the program automatically defaults to the largest size that can fit in the window up to a maximum of 255 characters.

Modes

Sixteen user modes allow you to determine the kind of input you will enter (**Table 1**). In cases where two modes fit the same description, the second mode will not read the string from memory into the input buffer.

The user length parameter doesn't specify the length in mode 7. If the value is zero, mode 7 will get any character. If it is not zero, it waits for a keypress of the ASCII value of the parameter. For example, to GET the letter A, use:

```
& INPUT A$,7,ASC("A")
```

To get any character, use:

```
& INPUT A$,7,0
```

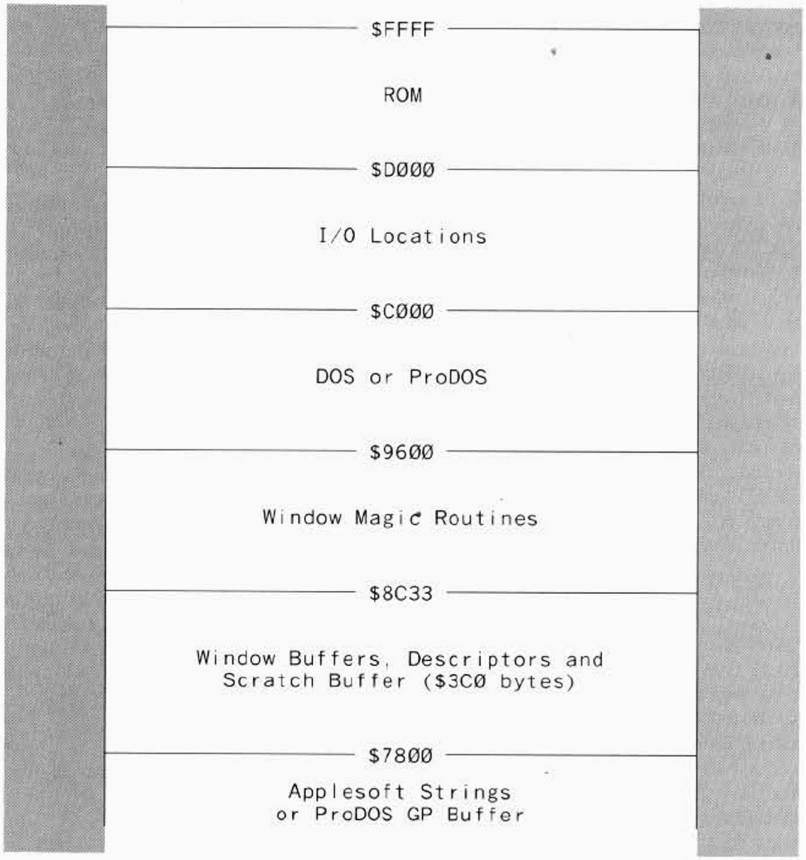
You can use the Apple's normal INPUT statement for input from the disk, but mode 8 offers the advantage of letting you input commas from text files (without evoking an EXTRA IGNORED error). Note that this is the only mode that will work with the disk. Note also that the user length parameter still works here. In most cases, it should be 255. To write to text files or execute disk commands from within a program, use a normal PRINT statement.

Note: Reading from a text file under DOS 3.3, whether using input mode 8 or the regular Applesoft INPUT statement, causes the line from the last cursor position to the edge of the screen to be cleared. Under ProDOS the screen is cleared from the cursor to the end. Before reading from a text file, then, you should position the cursor to a "safe" place on the screen using the normal cursor positioning statements. For example, VTAB 19: HTAB 39 often does the trick. Never position the cursor on line 24 or the screen will scroll one line. Under ProDOS you may have to issue an & RESTORE command to restore information at the bottom of the screen.

Input for Numbers

The new input works only with strings, but you can go from strings to numbers and back using the STR$(X) and VAL(X$) statements. For instance, to convert the number X to the string X$, use X$=STR$(X), and to convert back, use X=VAL(X$).

FIGURE 1: Memory Allocation



ENTERING THE PROGRAM

To key in the program, use an assembler to enter the source code from **Listing 1** or use the Monitor to enter the hex code directly. Save the program using the command:

```
BSAVE WINDOW.MAGIC,A$8BE8,
L$A18
```

Two short demonstration programs are included. Enter the Applesoft program shown in **Listing 2** and save it with the command:

```
SAVE WINDOW.DEMO1
```

To key in the second demonstration program, enter the Applesoft program shown in **Listing 3** and save it with the command:

```
SAVE WINDOW.DEMO2
```

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

MEMORY ALLOCATION

Window Magic is set up to start its buffers at $7800 as shown in **Figure 1**. The memory allocations set in the published listings should be adequate for most applications, and your program need only include the following line at the beginning of the program:

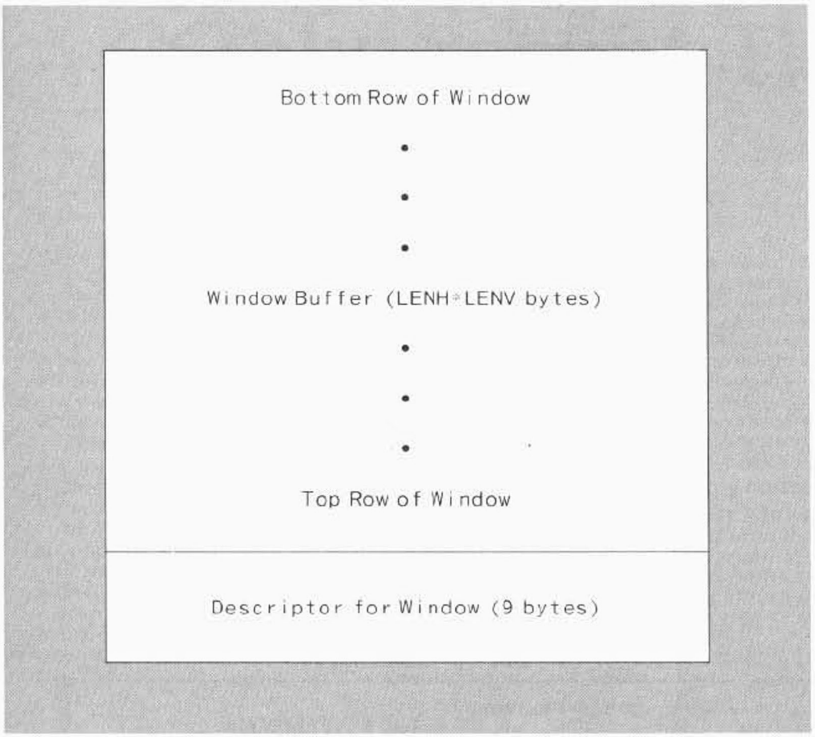
```
10 PRINT CHR$(4)"BRUN WINDOW
   .MAGIC"
```

This is how WINDOW.DEMO1 (**Listing 1**) initializes the system. Under ProDOS you must set HIMEM $400 below the value set by WINDOW.MAGIC. This is accomplished in WINDOW.DEMO1 with the following line:

```
90 IF PEEK(48896)=76 THEN POKE 115
   ,0: POKE 116,PEEK(116)-4
```

If you get an OUT OF MEMORY error when defining a window, you may need to allocate more memory for window storage. This can be done in either of two ways. First, you can modify the WINDOW.MAGIC program permanently by changing the value of MEMBOT (**line 1287** of **Listing 1**) and reassembling or resaving the file. Alternatively, your program can execute a series of Applesoft instructions that BLOAD WINDOW.MAGIC, change the values at $95FE and $95FF (decimal 38398 and 38399) and then start the program with a CALL statement. The following sequence is used in WINDOW.DEMO2 (**Listing 3**) to set HIMEM to $6000:

FIGURE 2: Arrangement of Window Data




```
80 PRINT CHR$(4)"BLOAD WINDOW
   .MAGIC": POKE 38398,0: POKE
   38399,96: CALL 35816
```

ProDOS still requires that HIMEM be lowered another $400. When changing the HIMEM value under ProDOS it is important that you change in increments of $400 and stay on $400 (1K) boundaries.

Caution: Since the startup code for WINDOW.MAGIC may be overwritten by window data, the program should always be re-started by BRUNning it from disk or BLOADing it and CALLing it. You should also re-start the program from disk when you change either the character set or the number of columns displayed.

TABLE 2: The Descriptor

Bytes	Function
7, 8	Cursor position relative to the top-left corner of the window (CH,CV)
6	ASCII character of the border
4, 5	Horizontal and vertical lengths of the window (LENH,LENV)
2, 3	Position of the window relative to the screen (POSH,POSV)
0, 1	Offset to the next window (LO,HI)

Memory for the windows is allocated upwards from HIMEM toward the Window Magic routines, so that the last defined window is highest in memory. **Figure 2** shows the format of an individual window. The text of a window is stored upside down in the buffer.

The nine-byte descriptor contains the parameters that define the window (see **Table 2**). The assembler symbols used in the listing are shown in parentheses. The scratch buffer is used to quickly dump all windows to the screen.

HOW IT WORKS

When Window Magic is first BRUN, a routine called SETUP (**line 76** in **Listing 1**) is executed. This resets HIMEM below the area of memory that holds the windows, and it sets the ampersand vector to point to the main program, specifically to START (**line 109**). It then JMPs to DEF1 (**line 164**), which defines the backdrop window and exits to BASIC. You should not define strings before BRUNning Window Magic because they will be overwritten and destroyed. Also, increasing MAXFILES under DOS 3.3 could destroy the memory area occupied by Window Magic.

After Window Magic is installed in memory, an ampersand command will transfer control to Window Magic by JMPing to START (**line 109**), which interprets the ampersand command and executes the appropriate routine. A SYNTAX ERROR or ILLEGAL QUANTITY error can occur if what follows the ampersand is not correct.

The most important routines are SETCHAR (**line 1057**), FASTDUMP (**line 1153**), and GETMEMCU (**line 1134**). When screen coordinates are entered, SETCHAR determines which window has priority over these coordinates, takes the appropriate character from the window, and puts it on the screen. FASTDUMP rapidly dumps the windows, in order from lowest priority to highest priority, into a scratch buffer and then dumps the scratch buffer onto the screen. This is used for large changes on the screen such as clearing, deleting, and moving windows. GETMEMCU determines the location of the cursor in the currently active window. This routine is used in all I/O to the window.

In order to avoid having to make patches for different machine configurations, sections of code were added to identify the configuration. This allows the same program to run on all machines. By setting up flag registers, configuration-dependent decisions can be made as conveniently as possible. The machine type is divided into three categories: II Plus, original //e, and //c or enhanced //e. This information is stored in the internal register IIC (**lines 92 and 97**). In addition, if the machine is found to be a //e or a //c, the 80-column card must be checked (**line 102**). If the 80-column card is enabled and displaying 80 columns, the width of a row is changed for calculations (**lines 103 and 104**) and the input cursor calculations are affected. If the 80-column card is enabled, but displaying only 40 columns, the cursor calculations are still affected. Finally, the enhanced //e and //c need further cursor adjustments when the 80-column firmware is in effect. The input cursor is handled in **lines 785-808**.

For BASIC programs with an unusual and professional look, use windows. You'll also find that Window Magic offers data entry routines that are superior to the Apple's.

Nibble MeterMan, Window Magic and Spacecade are available on diskette for an introductory price of $19.95 plus $1.50 shipping/handling ($2.50 outside the U.S.) from Nibble, 45 Winthrop St., Concord, MA 01742. Introductory price expires 12/31/85.

LISTING 1: WINDOW.MAGIC

```
                        1  ;*********************
                        2  ;    WINDOW.MAGIC     *
                        3  ;    BY PAUL NICK     *
                        4  ; COPYRIGHT (C) 1985  *
                        5  ; BY MICROSPARC, INC. *
                        6  ; CONCORD, MA 01742   *
                        7  ;*********************
                        8  ;
                        9  ; MERLIN ASSEMBLER
                       10  ;
                       11          ORG  $8BE8      ;=35816
                       12  ;
                       13  ; ZERO PAGE LOCATIONS
                       14  ;
                       15  PTR     =  $04          ;PTR TO WINDOW DESCRIPTOR
                       16  PTR1    =  $06          ;PTR TO POSITION WITHIN WINDOW.
                       17  PTR2    =  $08          ;PTR TO APPLESOFT STRING
                       18  CH      =  $17          ;CURSOR POSITION RELATIVE
                       19  CV      =  $18          ;TO TOP OF SCREEN
                       20  BASL    =  $28          ;SCREEN POINTER.
                       21  INVFLG  =  $32          ;VIDEO FLAG.
                       22  A1L     =  $3C          ;WORKING VARIABLES
                       23  A1H     =  $3D          ;ALSO PARAMETERS
                       24  A2L     =  $3E          ;FOR MONITOR
                       25  A2H     =  $3F          ;MEMORY MOVE.
                       26  A4L     =  $42
                       27  A4H     =  $43
                       28  INDEX   =  $5E
                       29  FRETOP  =  $6F
                       30  HIMEM   =  $73
                       31  FORPNT  =  $85
                       32  FACLO   =  $A1
                       33  CHRGET  =  $B1          ;GETS BYTE OF PROGRAM TEXT
                       34  CHRGOT  =  $B7
                       35  TXTPTR  =  $B8          ;PTR TO PROGRAM TEXT.
                       36  ZP      =  $FB
                       37  ;
                       38  IN      =  $200         ;INPUT BUFFER.
                       39  AMPER   =  $3F5         ;AMPERSAND VECTOR.
                       40  KBD     =  $C000        ;KEYBOARD
                       41  STROBE  =  $C010        ;STROBE
                       42  RDALTCHR =  $C01E       ;BIT $80=ALT. SET IN USE
                       43  COL80REG =  $C01F       ;BIT $80=80 COL. ON
                       44  PAGE1   =  $C054        ;DISABLE PAGE 2
                       45  PAGE2   =  $C055        ;ENABLE PAGE 2
                       46  ;
                       47  ; APPLESOFT ROUTINES
                       48  ;
                       49  MEMERR  =  $D410        ;OUT OF MEMORY ERROR.
                       50  GDBUFS  =  $D539        ;USED IN COLLECTING INPUT.
                       51  FRMNUM  =  $DD67        ;EVALUATE NUMBER FORMULA
                       52  CHKSTR  =  $DD6C        ;CHECKS FOR STRING.
                       53  FRMEVL  =  $DD7B        ;EVALUATE ANY FORMULA
                       54  CHKCOM  =  $DEBE        ;CHECKS FOR COMMA.
                       55  SYNERR  =  $DEC9        ;SYNTAX ERROR.
                       56  PTRGET  =  $DFE3        ;GETS PTR TO VARIABLE.
                       57  RANGERR =  $E199        ;ILLEGAL QUANTITY ERROR
                       58  STRLIT  =  $E3E7        ;PREPARE STR FOR PRINTING
                       59  FREFAC  =  $E600        ;FREE STRING POINTER
                       60  CONINT  =  $E6FB        ;CONVERT REAL TO INTEGER
                       61  FOUT    =  $ED34        ;NUMBER TO STRING ROUTINE
                       62  SETNORM =  $F273        ;NORMAL VIDEO
                       63  INVERSE =  $F277        ;INVERSE VIDEO
                       64  FLASH   =  $F280        ;FLASHING VIDEO
                       65  ;
                       66  ; MONITOR ROUTINES & LOCATIONS
                       67  ;
                       68  IDBYTE  =  $FBB3
                       69  IDBYTE2 =  $FBC0
                       70  BASCALC =  $FBC1        ;GETS SCREEN ADDRESS.
                       71  GETLN   =  $FD6F        ;GETS LINE OF INPUT.
                       72  MOVE    =  $FE2C        ;MEMORY MOVE ROUTINE.
                       73  ;
                       74  ; SETUP ROUTINE (MAY BE OVERWRITTEN BY WINDOWS)
                       75  ;
8BE8: A9 4C            76  SETUP   LDA  #$4C        ;SET UP
8BEA: 8D F5 03         77          STA  AMPER      ;AMPERSAND
8BED: AD FC 95         78          LDA  MEMTOP     ;VECTOR.
8BF0: 8D F6 03         79          STA  AMPER+1    ;RESET
8BF3: AD FD 95         80          LDA  MEMTOP+1   ;HIMEM
8BF6: 8D F7 03         81          STA  AMPER+2    ;AND
8BF9: AD FE 95         82          LDA  MEMBOT     ;FRETOP.
8BFC: 85 73            83          STA  HIMEM      ;DEFINE
8BFE: 85 6F            84          STA  FRETOP     ;BACKGROUND
8C00: AD FF 95         85          LDA  MEMBOT+1   ;WINDOW
8C03: 85 74            86          STA  HIMEM+1    ;BEFORE
8C05: 85 70            87          STA  FRETOP+1   ;EXITING
8C07: A2 80            88          LDX  #$80
8C09: AD B3 FB         89          LDA  IDBYTE     ;06=IIC/IIE
8C0C: C9 06            90          CMP  #$06
8C0E: D0 20            91          BNE  JMPOUT
8C10: 8E EB 95         92          STX  IIC
8C13: AD C0 FB         93          LDA  IDBYTE2    ;E0=ENH. IIE, 00=IIC
8C16: 29 0F            94          AND  #$0F       ;EA=ORIG. IIE
8C18: D0 05            95          BNE  CHECK80
8C1A: A9 C0            96          LDA  #$C0
8C1C: 8D EB 95         97          STA  IIC
8C1F: A9 28            98  CHECK80 LDA  #$28
8C21: 2C 1F C0         99          BIT  COL80REG
8C24: 10 04           100          BPL  STORCOL
8C26: 0A              101          ASL
8C27: 8E EA 95        102          STX  COL80
8C2A: 8D EC 95        103  STORCOL STA  NOCOLS
8C2D: 8D EF 95        104          STA  LENH
8C30: 4C AA 8C        105  JMPOUT  JMP  DEF1        ;TO BASIC
                      106  ;
                      107  ; START OF MAIN PROGRAM
                      108  ;
8C33: A2 0A           109  START   LDX  #$0A        ;FIND CORRECT
8C35: DD 40 8C        110  WDFIND  CMP  WORDVALS,X  ;APPLESOFT
8C38: F0 10           111          BEQ  WDFOUND    ;TOKEN OR
8C3A: CA              112          DEX             ;PRODUCE
8C3B: 10 F8           113          BPL  WDFIND     ;SYNTAX
8C3D: 4C C9 DE        114  GSYNERR JMP  SYNERR     ;ERROR.
8C40: B8 B4 85        115  WORDVALS HEX  B8B48597A2 ;NEW STATEMENT
8C43: 97 A2
8C45: 96 BA 84        116          HEX  96BA84A8AE ;TOKENS
8C48: AB AE
8C4A: 20 B1 00        117  WDFOUND JSR  CHRGET     ;EAT UP A BYTE
8C4D: 8A              118          TXA             ;GET ADDRESS
8C4E: 0A              119          ASL             ;OF ROUTINE
8C4F: AA              120          TAX             ;PUSH ON STACK
8C50: BD 5A 8C        121          LDA  ADDLIST+1,X ;AND RTS
8C53: 48              122          PHA             ;TO ROUTINE
8C54: BD 59 8C        123          LDA  ADDLIST,X
8C57: 48              124          PHA
8C58: 60              125          RTS
                      126  ;
8C59: 6C 8C           127  ADDLIST DA   DEF-1       ;ADDRESS
8C5B: 01 8D           128          DA   ON-1        ;LIST OF
8C5D: 29 8D           129          DA   DEL-1       ;ROUTINES
8C5F: 66 8D           130          DA   HOME-1
8C61: F9 8D           131          DA   VTAB-1
8C63: 1E 8E           132          DA   HTAB-1
8C65: 60 8E           133          DA   PRINT-1
8C67: AA 8E           134          DA   INPUT-1
8C69: 96 92           135          DA   GOTO-1
8C6B: D4 92           136          DA   RESTORE-1
                      137  ;
8C6D: 20 E8 94        138  DEF      JSR  SAVEPOS    ;SAVE CURSOR POSITION
8C70: A2 00           139          LDX  #$00       ;CALL GETNUMPR
8C72: 86 3D           140          STX  A1H        ;TO GET PARAMETERS.
8C74: 86 3C           141          STX  A1L
8C76: CA              142          DEX
8C77: 86 3E           143          STX  A2L
8C79: 20 B9 94        144          JSR  GETNUMPR
8C7C: 8D ED 95        145          STA  POSH
8C7F: 20 B9 94        146          JSR  GETNUMPR
8C82: 8D EE 95        147          STA  POSV
8C85: A2 03           148          LDX  #$03
8C87: 86 3D           149          STX  A1H
8C89: A2 60           150          LDX  #$60
8C8B: 86 3E           151          STX  A2L
8C8D: 20 B9 94        152          JSR  GETNUMPR
8C90: 8D EF 95        153          STA  LENH
8C93: 20 B9 94        154          JSR  GETNUMPR
8C96: 8D F0 95        155          STA  LENV
8C99: A9 01           156          LDA  #1
8C9B: 85 3C           157          STA  A1L
8C9D: A2 00           158          LDX  #$00
8C9F: 86 3D           159          STX  A1H
8CA1: CA              160          DEX
8CA2: 86 3E           161          STX  A2L
8CA4: 20 B9 94        162          JSR  GETNUMPR
8CA7: 8D F1 95        163          STA  BORDER
8CAA: AD F4 95        164  DEF1     LDA  NUMWIND    ;MAKE NEW WINDOW
8CAD: 8D F5 95        165          STA  CURWIND    ;THE CURRENT WINDOW.
8CB0: 20 D5 93        166          JSR  GWINADD    ;GET PTR TO NEW WINDOW
8CB3: 38              167          SEC             ;FIND HOW
8CB4: AD FC 95        168          LDA  MEMTOP     ;MUCH MEMORY
8CB7: E5 04           169          SBC  PTR        ;IS LEFT.
8CB9: 85 3C           170          STA  A1L
8CBB: AD FD 95        171          LDA  MEMTOP+1
8CBE: E5 05           172          SBC  PTR+1
8CC0: 85 3D           173          STA  A1H
8CC2: A9 09           174          LDA  #$09       ;FIND HOW
8CC4: 85 3E           175          STA  A2L        ;MUCH MEMORY
8CC6: A9 00           176          LDA  #$00       ;NEW WINDOW NEEDS.
8CC8: 85 3F           177          STA  A2H
8CCA: AC F0 95        178          LDY  LENV
8CCD: 18              179  DADDLEN  CLC
8CCE: A5 3E           180          LDA  A2L
8CD0: 6D EF 95        181          ADC  LENH
8CD3: 85 3E           182          STA  A2L
8CD5: 90 02           183          BCC  DADDLEN1
8CD7: E6 3F           184          INC  A2H
8CD9: 88              185  DADDLEN1 DEY
8CDA: D0 F1           186          BNE  DADDLEN
8CDC: A5 3F           187          LDA  A2H        ;SEE IF THERE
8CDE: C5 3D           188          CMP  A1H        ;IS ENOUGH
8CE0: 90 0B           189          BCC  DMEMYY     ;MEMORY LEFT.
8CE2: F0 03           190          BEQ  DMEMY
8CE4: 4C 10 D4        191  MEMOUT   JMP  MEMERR
8CE7: A5 3E           192  DMEMY    LDA  A2L
8CE9: C5 3C           193          CMP  A1L
8CEB: B0 F7           194          BCS  MEMOUT
8CED: A5 3E           195  DMEMYY   LDA  A2L        ;SAVE THE MEMORY
8CEF: 91 04           196          STA  (PTR),Y    ;NEEDED BY THE
8CF1: C8              197          INY             ;WINDOW AS THE OFFSET
8CF2: A5 3F           198          LDA  A2H        ;TO THE NEXT WINDOW
8CF4: 91 04           199          STA  (PTR),Y    ;IN THE DESCRIPTOR.
8CF6: 20 15 93        200          JSR  SAVEDESC   ;SAVE REST OF DESCRIPTOR.
8CF9: 20 00 93        201          JSR  POSICALC   ;CALCULATE POS1H & POS1V.
8CFC: EE F4 95        202          INC  NUMWIND    ;INCREMENT NUMBER OF WINDOWS
8CFF: 4C 67 8D        203          JMP  HOME       ;HOME NEW WINDOW AND EXIT.
                      204  ;
8D02: 20 E8 94        205  ON       JSR  SAVEPOS    ;SAVE CURSOR POS.
8D05: A2 01           206          LDX  #$01       ;GET WINDOW
8D07: 86 3C           207          STX  A1L        ;TO BE OPENED
8D09: CA              208          DEX
8D0A: 86 3D           209          STX  A1H
8D0C: AE F4 95        210          LDX  NUMWIND
8D0F: CA              211          DEX
8D10: 86 3E           212          STX  A2L
8D12: 20 B9 94        213          JSR  GETNUMPR
8D15: 8D F5 95        214  ON1      STA  CURWIND    ;SAVE IT AS CURRENT WINDOW
8D18: 20 D5 93        215          JSR  GWINADD    ;GET WINDOW PTR.
8D1B: 20 F4 92        216          JSR  LOADDESC   ;GET DESCRIPTOR.
8D1E: A0 07           217          LDY  #$07       ;GET CURSOR POSITION
8D20: B1 04           218          LDA  (PTR),Y
8D22: 85 17           219          STA  CH
```

```
8D24: C8          220          INY
8D25: B1 04       221          LDA  (PTR),Y
8D27: 85 18       222          STA  CV
8D29: 60          223          RTS                  ;EXIT
                  224
8D2A: AD F5 95    225  DEL     LDA  CURWIND         ;DON'T DELETE
8D2D: D0 03       226          BNE  DEL0            ;WINDOW 0
8D2F: 4C 3D 8C    227          JMP  GSYNERR
8D32: A5 04       228  DEL0    LDA  PTR             ;SET UP
8D34: 85 42       229          STA  A4L             ;PARAMETERS
8D36: A5 05       230          LDA  PTR+1           ;FOR MEMORY
8D38: 85 43       231          STA  A4H             ;MOVE ROUTINE
8D3A: AE F5 95    232          LDX  CURWIND         ;TO ERASE THE
8D3D: E8          233          INX                  ;CURRENT WINDOW
8D3E: 20 D8 93    234          JSR  GWINADD1
8D41: A5 04       235          LDA  PTR
8D43: 85 3C       236          STA  A1L
8D45: A5 05       237          LDA  PTR+1
8D47: 85 3D       238          STA  A1H
8D49: AE F4 95    239          LDX  NUMWIND
8D4C: 20 D8 93    240          JSR  GWINADD1
8D4F: A5 04       241          LDA  PTR
8D51: 85 3E       242          STA  A2L
8D53: A5 05       243          LDA  PTR+1
8D55: 85 3F       244          STA  A2H
8D57: A0 00       245          LDY  #$00
8D59: 20 2C FE    246          JSR  MOVE            ;DO IT
8D5C: CE F4 95    247          DEC  NUMWIND         ;DEC  NUMBER OF WINDOWS
8D5F: 20 F4 94    248          JSR  FASTDUMP        ;REDUMP SCREEN
8D62: A9 00       249          LDA  #0              ;TURN ON
8D64: 4C 15 8D    250          JMP  ON1             ;WINDOW 0
                  251
8D67: AE F0 95    252  HOME    LDX  LENV            ;FILL THE
8D6A: 20 14 94    253          JSR  GETMEMAD        ;SCREEN WITH
8D6D: A5 06       254          LDA  PTR1            ;SPACES
8D6F: 85 3C       255          STA  A1L
8D71: A5 07       256          LDA  PTR1+1
8D73: 85 3D       257          STA  A1H
8D75: A2 00       258          LDX  #0
8D77: 20 14 94    259          JSR  GETMEMAD
8D7A: A4 06       260          LDY  PTR1
8D7C: A9 00       261          LDA  #0
8D7E: 85 06       262          STA  PTR1
8D80: A9 A0       263          LDA  #$A0
8D82: 91 06       264  HLOOP   STA  (PTR1),Y
8D84: C8          265          INY
8D85: D0 02       266          BNE  HLOOP1
8D87: E6 07       267          INC  PTR1+1
8D89: C4 3C       268  HLOOP1  CPY  A1L
8D8B: D0 F5       269          BNE  HLOOP
8D8D: A6 07       270          LDX  PTR1+1
8D8F: E4 3D       271          CPX  A1H
8D91: D0 EF       272          BNE  HLOOP
8D93: AD F1 95    273          LDA  BORDER          ;IF THERE
8D96: F0 4F       274          BEQ  HOMEOUT         ;IS  BORDER
8D98: A2 00       275          LDX  #$00            ;THEN MAKE IT
8D9A: 20 14 94    276          JSR  GETMEMAD
8D9D: AC EF 95    277          LDY  LENH
8DA0: 88          278          DEY
8DA1: 88          279          DEY
8DA2: AD F1 95    280          LDA  BORDER
8DA5: 91 06       281  HBORD1  STA  (PTR1),Y
8DA7: 88          282          DEY
8DA8: 10 FB       283          BPL  HBORD1
8DAA: 18          284          CLC
8DAB: A5 06       285          LDA  PTR1
8DAD: CE EF 95    286          DEC  LENH
8DB0: 6D EF 95    287          ADC  LENH
8DB3: EE EF 95    288          INC  LENH
8DB6: 85 06       289          STA  PTR1
8DB8: 90 02       290          BCC  HBORD15
8DBA: E6 07       291          INC  PTR1+1
8DBC: AE F0 95    292  HBORD15 LDX  LENV
8DBF: CA          293          DEX
8DC0: A0 00       294          LDY  #$00
8DC2: AD F1 95    295  HBORD2  LDA  BORDER
8DC5: 91 06       296          STA  (PTR1),Y
8DC7: C8          297          INY
8DC8: 91 06       298          STA  (PTR1),Y
8DCA: 88          299          DEY
8DCB: CA          300          DEX
8DCC: F0 0E       301          BEQ  HBORD3
8DCE: 18          302          CLC
8DCF: A5 06       303          LDA  PTR1
8DD1: 6D EF 95    304          ADC  LENH
8DD4: 85 06       305          STA  PTR1
8DD6: 90 EA       306          BCC  HBORD2
8DD8: E6 07       307          INC  PTR1+1
8DDA: D0 E6       308          BNE  HBORD2
8DDC: AC EF 95    309  HBORD3  LDY  LENH
8DDF: AD F1 95    310          LDA  BORDER
8DE2: 91 06       311  HBORD4  STA  (PTR1),Y
8DE4: 88          312          DEY
8DE5: 10 FB       313          BPL  HBORD4
8DE7: 20 F4 94    314  HOMEOUT  JSR  FASTDUMP       ;REDUMP SCREEN
8DEA: 20 4E 93    315  HOMECURS JSR  SHRINK         ;HOME CURSOR
8DED: AD ED 95    316          LDA  POSH
8DF0: 85 17       317          STA  CH
8DF2: AD EE 95    318          LDA  POSV
8DF5: 85 18       319          STA  CV
8DF7: 4C 60 93    320          JMP  EXPAND          ;EXIT
                  321
8DFA: A2 01       322  VTAB    LDX  #$01            ;GET PARAMETER
8DFC: 86 3C       323          STX  A1L
8DFE: 86 3D       324          STX  A1H
8E00: AE F0 95    325          LDX  LENV
8E03: AD F1 95    326          LDA  BORDER
8E06: F0 02       327          BEQ  VTAB1
8E08: CA          328          DEX
8E09: CA          329          DEX
8E0A: 86 3F       330  VTAB1   STX  A2L
8E0C: 20 B9 94    331          JSR  GETNUMPR
8E0F: 48          332          PHA                  ;ADD TO POSV
8E10: 20 4E 93    333          JSR  SHRINK          ;TO PRODUCE
8E13: 68          334          PLA                  ;CURSOR
```

```
8E14: 18          335          CLC                  ;INDEX RELATIVE
8E15: 6D EE 95    336          ADC  POSV            ;TO TOP OF SCREEN
8E18: 85 18       337          STA  CV
8E1A: C6 18       338          DEC  CV
8E1C: 4C 60 93    339          JMP  EXPAND
                  340
8E1F: A2 01       341  HTAB    LDX  #$01            ;SEE ABOVE
8E21: 86 3C       342          STX  A1L
8E23: 86 3D       343          STX  A1H
8E25: AE EF 95    344          LDX  LENH
8E28: AD F1 95    345          LDA  BORDER
8E2B: F0 02       346          BEQ  HTAB1
8E2D: CA          347          DEX
8E2E: CA          348          DEX
8E2F: 86 3E       349  HTAB1   STX  A2L
8E31: 20 B9 94    350          JSR  GETNUMPR
8E34: 48          351          PHA
8E35: 20 4E 93    352          JSR  SHRINK
8E38: 68          353          PLA
8E39: 18          354          CLC
8E3A: 6D ED 95    355          ADC  POSH
8E3D: 85 17       356          STA  CH
8E3F: C6 17       357          DEC  CH
8E41: 4C 60 93    358          JMP  EXPAND
                  359
8E44: 20 00 E6    360  STRPRT  JSR  FREFAC         ;PRINT STRING
8E47: 8D F8 95    361          STA  SEMI            ;IN LOOP
8E4A: A0 00       362          LDY  #$00
8E4C: EE F8 95    363          INC  SEMI
8E4F: CE F8 95    364  STRPRTL DEC  SEMI
8E52: F0 0D       365          BEQ  PRINT
8E54: 98          366          TYA
8E55: 48          367          PHA
8E56: B1 5E       368          LDA  (INDEX),Y
8E58: 20 DB 92    369          JSR  COUT
8E5B: 68          370          PLA
8E5C: A8          371          TAY
8E5D: C8          372          INY
8E5E: 4C 4F 8E    373          JMP  STRPRTL
8E61: 20 B7 00    374  PRINT   JSR  CHRGOT          ;GET LAST CHAR
8E64: F0 3B       375  PRINT1  BEQ  PRTCR           ;END OF LINE? EXIT AFTER CR
8E66: F0 3C       376  PRTEVAL BEQ  PRTEXIT         ;END OF LINE? DON'T DO CR
8E68: C9 3B       377          CMP  #$3B            ;";"
8E6A: F0 39       378          BEQ  PRTNXCHR
8E6C: C9 40       379          CMP  #$40            ;"@"
8E6E: D0 06       380          BNE  PRTEVAL1
8E70: 20 48 93    381          JSR  CR
8E73: 4C A5 8E    382          JMP  PRTNXCHR
8E76: C9 23       383  PRTEVAL1 CMP  #$23            ;"#"
8E78: D0 05       384          BNE  PRTEVAL2
8E7A: 20 73 F2    385          JSR  SETNORM
8E7D: F0 26       386          BEQ  PRTNXCHR        ;ALWAYS TAKEN
8E7F: C9 24       387  PRTEVAL2 CMP  #$24            ;"$"
8E81: D0 05       388          BNE  PRTEVAL3
8E83: 20 77 F2    389          JSR  INVERSE
8E86: F0 1D       390          BEQ  PRTNXCHR        ;ALWAYS TAKEN
8E88: C9 25       391  PRTEVAL3 CMP  #$25            ;"%"
8E8A: D0 05       392          BNE  PRTEVAL4
8E8C: 20 80 F2    393          JSR  FLASH           ;ALWAYS TAKEN
8E8F: D0 14       394          BNE  PRTNXCHR
8E91: 20 7B DD    395  PRTEVAL4 JSR  FRMEVL          ;EVALUATE FORMULA (STR OR NUM)
8E94: 24 11       396          BIT  $11             ;STRING OR NUMBER?
8E96: 30 AC       397          BMI  STRPRT          ;BRANCH IF STRING
8E98: 20 34 ED    398          JSR  FOUT            ;TURN NUM IN FAC TO STRING
8E9B: 20 E7 E3    399          JSR  STRLIT          ;PREPARE IT FOR PRINTING
8E9E: 4C 44 8E    400          JMP  STRPRT          ;PRINT IT
8EA1: 20 48 93    401  PRTCR   JSR  CR              ;CARRIAGE RETURN
8EA4: 60          402  PRTEXIT RTS
8EA5: 20 B1 00    403  PRTNXCHR JSR  CHRGET         ;GETS NEXT CHAR
8EA8: 4C 66 8E    404          JMP  PRTEVAL
                  405
8EAB: 20 73 F2    406  INPUT   JSR  SETNORM        ;SET NORMAL VIDEO
8EAE: 20 E3 DF    407          JSR  PTRGET          ;GET POINTER
8EB1: 20 6C DD    408          JSR  CHKSTR          ;OF STRING
8EB4: 85 85       409          STA  FORPNT          ;VARIABLE
8EB6: 84 86       410          STY  FORPNT+1
8EB8: A9 01       411          LDA  #$01
8EBA: 8D F8 95    412          STA  SEMI
8EBD: 20 B7 00    413          JSR  CHRGOT          ;CHECK FOR
8EC0: C9 3B       414          CMP  #$3B            ;SEMICOLON
8EC2: F0 0A       415          BEQ  GETSTRP         ;OR
8EC4: CE F8 95    416          DEC  SEMI            ;FOR A
8EC7: C9 2C       417          CMP  #$2C            ;COMMA
8EC9: F0 03       418          BEQ  GETSTRP
8ECB: 4C C9 DE    419          JMP  SYNERR
8ECE: 20 B1 00    420  GETSTRP JSR  CHRGET
8ED1: A0 00       421          LDY  #$00            ;GET LENGTH
8ED3: B1 85       422          LDA  (FORPNT),Y      ;AND POINTER
8ED5: 8D F7 95    423          STA  LEN             ;TO ACTUAL
8ED8: C8          424          INY                  ;STRING
8ED9: B1 85       425          LDA  (FORPNT),Y
8EDB: 85 08       426          STA  PTR2
8EDD: C8          427          INY
8EDE: B1 85       428          LDA  (FORPNT),Y
8EE0: 85 09       429          STA  PTR2+1
8EE2: A2 00       430          LDX  #$00            ;GET INPUT
8EE4: 86 3C       431          STX  A1L             ;PARAMETERS
8EE6: 86 3D       432          STX  A1H
8EE8: A9 0F       433          LDA  #$0F
8EEA: 85 3E       434          STA  A2L
8EEC: 20 B9 94    435          JSR  GETNUMPR
8EEF: C9 09       436          CMP  #$09
8EF1: 08          437          PHP
8EF2: 90 04       438          BCC  I1
8EF4: E9 09       439          SBC  #$09
8EF6: E6 3F       440          INC  A2H
8EF8: 8D FA 95    441  I1      STA  MODE
8EFB: C9 07       442          CMP  #$07
8EFD: 08          443          PHP
8EFE: A9 FF       444          LDA  #$FF
8F00: B0 28       445          BCS  MAXOUT
8F02: 20 4E 93    446          JSR  SHRINK
```

```
8F05: AC EF 95  447          LDY  LENH      ;OF STRING THAT
8F08: AE F0 95  448          LDX  LENV      ;THAT CAN FIT
8F0B: AD F1 95  449          LDA  BORDER    ;INTO WINDOW
8F0E: F0 04     450          BEQ  MAXLENCK
8F10: CA        451          DEX
8F11: CA        452          DEX
8F12: 88        453          DEY
8F13: 88        454          DEY
8F14: 84 42     455  MAXLENCK STY  A4L
8F16: AD F2 95  456          LDA  POS1H
8F19: E5 17     457          SBC  CH
8F1B: 48        458          PHA
8F1C: 20 60 93  459          JSR  EXPAND
8F1F: 68        460          PLA
8F20: 18        461          CLC
8F21: CA        462  MAXLOOP  DEX
8F22: F0 06     463          BEQ  MAXOUT
8F24: 65 42     464          ADC  A4L
8F26: 90 F9     465          BCC  MAXLOOP
8F28: A9 FF     466          LDA  #$FF
8F2A: 85 3E     467  MAXOUT   STA  A2L
8F2C: A2 01     468          LDX  #$01
8F2E: 86 3C     469          STX  A1L
8F30: E8        470          INX
8F31: 8E FB 95  471          STX  CASE
8F34: 20 B9 94  472          JSR  GETNUMPR
8F37: 28        473          PLP
8F38: 08        474          PHP
8F39: B0 06     475          BCS  MAXST0
8F3B: C9 00     476          CMP  #$00
8F3D: D0 02     477          BNE  MAXST0
8F3F: A5 3E     478          LDA  A2L
8F41: 8D F9 95  479  MAXST0   STA  MAXLEN
8F44: CD F7 95  480          CMP  LEN
8F47: B0 03     481          BCS  I0
8F49: 8D F7 95  482          STA  LEN
8F4C: 28        483  I0      PLP
8F4D: F0 14     484          BEQ  GET
8F4F: 90 2D     485          BCC  READVAR
8F51: 68        486          PLA
8F52: 20 6F FD  487          JSR  GETLN     ;CODE FOR MODE 8
8F55: EC F9 95  488          CPX  MAXLEN    ;THE DOS MODE
8F58: 90 03     489          BCC  DOSOUT
8F5A: AE F9 95  490          LDX  MAXLEN
8F5D: 8E F6 95  491  DOSOUT   STX  CURS
8F60: 4C 7D 91  492          JMP  LEAVE
8F63: 68        493  GET     PLA
8F64: 20 0D 92  494  GET1    JSR  GETCHAR   ;THIS IS MODE 7
8F67: 8D 00 02  495          STA  IN        ;THE GET A CHAR MODE
8F6A: AD F9 95  496          LDA  MAXLEN
8F6D: F0 07     497          BEQ  GETOUT
8F6F: 09 80     498          ORA  #$80
8F71: CD 00 02  499          CMP  IN
8F74: D0 EE     500          BNE  GET1
8F76: A9 01     501  GETOUT   LDA  #$01
8F78: 8D F6 95  502          STA  CURS
8F7B: 4C 7D 91  503          JMP  LEAVE
8F7E: A0 00     504  READVAR  LDY  #0
8F80: 8C F6 95  505          STY  CURS
8F83: 28        506          PLP
8F84: B0 1C     507          BCS  GBEGIN
8F86: CC F7 95  508  MOVEVAR  CPY  LEN       ;MOVE THE STRING
8F89: F0 17     509          BEQ  GBEGIN     ;INTO THE INPUT
8F8B: B1 08     510          LDA  (PTR2),Y   ;BUFFER
8F8D: 09 80     511          ORA  #$80       ;AND ONTO
8F8F: 20 B3 91  512          JSR  CHK        ;THE SCREEN
8F92: AC F6 95  513          LDY  CURS
8F95: B0 0B     514          BCS  GBEGIN
8F97: 99 00 02  515          STA  IN,Y
8F9A: 20 DB 92  516          JSR  COUT
8F9D: AC F6 95  517          LDY  CURS
8FA0: D0 E4     518          BNE  MOVEVAR
8FA2: 8C F7 95  519  GBEGIN   STY  LEN
8FA5: 4C 84 90  520          JMP  BEGIN
8FA8: AD FB 95  521  RDKEY    LDA  CASE      ;SHIFT OUT
8FAB: C9 01     522          CMP  #$01
8FAD: D0 03     523          BNE  RD0
8FAF: CE FB 95  524          DEC  CASE
8FB2: 20 0D 92  525  RD0      JSR  GETCHAR   ;GET A CHAR
8FB5: 20 59 92  526          JSR  GETKEY2
8FB8: C9 9B     527          CMP  #$9B      ;ESC?
8FBA: D0 11     528          BNE  NEXTA
8FBC: EE FB 95  529  CASETOG  INC  CASE      ;SHIFT HANDLER
8FBF: AD FB 95  530          LDA  CASE
8FC2: C9 03     531          CMP  #$03
8FC4: D0 EC     532          BNE  RD0
8FC6: A9 00     533          LDA  #$00
8FC8: 8D FB 95  534          STA  CASE
8FCB: F0 E5     535          BEQ  RD0
8FCD: C9 8D     536  NEXTA    CMP  #$8D      ;RETURN?
8FCF: D0 1F     537          BNE  NEXTB
8FD1: 20 8C 91  538  RETURN   JSR  BEGINSUB  ;GET RID OF TRAILING
8FD4: 20 9E 91  539          JSR  REDUCE     ;SPACES. MOVE
8FD7: AE F6 95  540  RETURN1  LDX  CURS       ;CURSOR TO
8FDA: EC F7 95  541          CPX  LEN        ;END OF STRING
8FDD: B0 06     542          BCS  RETURN2    ;AND EXIT
8FDF: 20 22 93  543          JSR  ADVANCE    ;THROUGH INPUT
8FE2: 4C D7 8F  544          JMP  RETURN1    ;COLLECTOR.
8FE5: AD F8 95  545  RETURN2  LDA  SEMI
8FE8: D0 03     546          BNE  RETURN3
8FEA: 20 48 93  547          JSR  CR
8FED: 4C 7D 91  548  RETURN3  JMP  LEAVE
8FF0: C9 88     549  NEXTB    CMP  #$88      ;BACKSPACE?
8FF2: D0 0B     550          BNE  NEXTC
8FF4: AD F6 95  551  BACKSPAC LDA  CURS       ;MOVE CURSOR BACK.
8FF7: F0 03     552          BEQ  GRDKEY
8FF9: 20 7D 92  553          JSR  BACK
8FFC: 4C A8 8F  554  GRDKEY   JMP  RDKEY
8FFF: C9 91     555  NEXTC    CMP  #$91      ;^Q?
9001: D0 2C     556          BNE  NEXTD
9003: 20 9E 91  557  QUIT     JSR  REDUCE     ;REMOVE TRAILING SPACES
9006: AD F7 95  558          LDA  LEN        ;IF CURSOR

9009: 85 3D     559          STA  A1H        ;OUT TOO FAR.
900B: 85 3C     560          STA  A1L
900D: CD F6 95  561          CMP  CURS       ;BACK IT UP
9010: B0 03     562          BCS  QUIT2
9012: 20 90 91  563          JSR  BACKLOOP
9015: AD F6 95  564  QUIT2    LDA  CURS       ;BLANK OUT REST
9018: 85 3C     565          STA  A1L        ;OF STRING
901A: 4C 22 90  566          JMP  QUIT4
901D: A9 A0     567  QUIT3    LDA  #$A0
901F: 20 DB 92  568          JSR  COUT
9022: AD F6 95  569  QUIT4    LDA  CURS
9025: C5 3D     570          CMP  A1H
9027: D0 F4     571          BNE  QUIT3
9029: 20 90 91  572          JSR  BACKLOOP   ;BACK UP AGAIN
902C: 4C E5 8F  573          JMP  RETURN2    ;AND EXIT
902F: C9 95     574  NEXTD    CMP  #$95       ;FORWARD SPACE?
9031: D0 12     575          BNE  NEXTE
9033: AE F6 95  576  FORWARD  LDX  CURS       ;ADVANCE CURSOR
9036: EC F7 95  577          CPX  LEN        ;IF CURSOR AT END OF STRING
9039: 90 04     578          BCC  FOR1       ;THEN LOAD A
903B: A9 A0     579          LDA  #$A0       ;SPACE AND CONTINUE
903D: D0 06     580          BNE  NEXTE
903F: 20 22 93  581  FOR1     JSR  ADVANCE
9042: 4C A8 8F  582          JMP  RDKEY
9045: C9 8A     583  NEXTE    CMP  #$8A       ;^J?
9047: D0 1B     584          BNE  NEXTF
9049: A5 17     585  DOWN     LDA  CH         ;ADVANCE CURSOR
904B: 85 3C     586          STA  A1L        ;UNTIL IT IS
904D: 4C 59 90  587          JMP  DOWN2      ;DIRECTLY
9050: 20 22 93  588  DOWN1    JSR  ADVANCE    ;BELOW
9053: A5 17     589          LDA  CH         ;OR AT END
9055: C5 3C     590          CMP  A1L        ;OF STRING
9057: F0 08     591          BEQ  DOWNOUT
9059: AD F6 95  592  DOWN2    LDA  CURS
905C: CD F7 95  593          CMP  LEN
905F: D0 EF     594          BNE  DOWN1
9061: 4C A8 8F  595  DOWNOUT  JMP  RDKEY
9064: C9 8B     596  NEXTF    CMP  #$8B       ;^K?
9066: D0 18     597          BNE  NEXTG
9068: A5 17     598  UP       LDA  CH         ;BACK UP CURSOR
906A: 85 3C     599          STA  A1L        ;UNTIL IT IS
906C: 4C 78 90  600          JMP  UP2        ;DIRECTLY ABOVE
906F: 20 7D 92  601  UP1      JSR  BACK       ;OR AT BEGINNING
9072: A5 17     602          LDA  CH         ;OF INPUT
9074: C5 3C     603          CMP  A1L
9076: F0 05     604          BEQ  UPOUT
9078: AD F6 95  605  UP2      LDA  CURS
907B: D0 F2     606          BNE  UP1
907D: 4C A8 8F  607  UPOUT    JMP  RDKEY
9080: C9 82     608  NEXTG    CMP  #$82       ;^B?
9082: D0 06     609          BNE  NEXTH
9084: 20 8C 91  610  BEGIN    JSR  BEGINSUB   ;GO TO BEGINNING
9087: 4C A8 8F  611          JMP  RDKEY      ;OF STRING
908A: C9 8E     612  NEXTH    CMP  #$8E       ;^N?
908C: D0 17     613          BNE  NEXT1
908E: 20 8C 91  614  END      JSR  BEGINSUB   ;GOTO BEGINNING
9091: 20 9E 91  615          JSR  REDUCE     ;REDUCE TRAILING SPACES
9094: 4C 9A 90  616          JMP  END2       ;AND GOTO
9097: 20 22 93  617  END1     JSR  ADVANCE    ;END OF STRING.
909A: AD F6 95  618  END2     LDA  CURS
909D: CD F7 95  619          CMP  LEN
90A0: D0 F5     620          BNE  END1
90A2: 4C A8 8F  621          JMP  RDKEY
90A5: C9 84     622  NEXT1    CMP  #$84       ;^D?
90A7: F0 04     623          BEQ  DELETE
90A9: C9 FF     624          CMP  #$FF       ;DELETE?
90AB: D0 3F     625          BNE  NEXTJ
90AD: AD F6 95  626  DELETE   LDA  CURS
90B0: CD F7 95  627          CMP  LEN
90B3: F0 34     628          BEQ  DELOUT
90B5: 85 3C     629          STA  A1L        ;SAVE CURSOR POS
90B7: EE F6 95  630  DELOOP   INC  CURS       ;MOVE EACH
90BA: AE F6 95  631          LDX  CURS       ;CHAR DOWN ONE
90BD: EC F7 95  632          CPX  LEN        ;IN BUFFER
90C0: B0 19     633          BCS  DOUT       ;PRINTING THEM
90C2: BD 00 02  634          LDA  IN,X       ;AT THE SAME
90C5: 48        635          PHA             ;TIME
90C6: 20 DB 92  636          JSR  COUT
90C9: CE F6 95  637          DEC  CURS
90CC: CE F6 95  638          DEC  CURS
90CF: AE F6 95  639          LDX  CURS
90D2: 68        640          PLA
90D3: 9D 00 02  641          STA  IN,X
90D6: EE F6 95  642          INC  CURS
90D9: D0 DC     643          BNE  DELOOP
90DB: A9 A0     644  DOUT     LDA  #$A0       ;BLANK OUT
90DD: 20 DB 92  645          JSR  COUT       ;LAST CHAR
90E0: CE F6 95  646          DEC  CURS
90E3: 20 90 91  647          JSR  BACKLOOP   ;BACK UP
90E6: CE F7 95  648          DEC  LEN
90E9: 4C A8 8F  649  DELOUT   JMP  RDKEY      ;EXIT
90EC: C9 89     650  NEXTJ    CMP  #$89
90EE: D0 65     651          BNE  NEXTK
90F0: AC FA 95  652  INSERT   LDY  MODE       ;DON'T INSERT
90F3: C0 05     653          CPY  #$05       ;BEFORE "+" OR "-"
90F5: 90 0E     654          BCC  INS1       ;DURING MUMBER
90F7: AE F6 95  655          LDX  CURS       ;ENTRY
90FA: BD 00 02  656          LDA  IN,X
90FD: C9 AB     657          CMP  #"+"
90FF: F0 51     658          BEQ  INSOUT
9101: C9 AD     659          CMP  #"-"
9103: F0 4D     660          BEQ  INSOUT
9105: AE F7 95  661  INS1     LDX  LEN
9108: EC F6 95  662          CPX  CURS
910B: F0 45     663          BEQ  INSOUT
910D: CA        664          DEX             ;FLUSH LAST CHAR
910E: BD 00 02  665          LDA  IN,X       ;IF IT'S
9111: C9 A0     666          CMP  #$A0       ;A SPACE.
9113: D0 01     667          BNE  INS2
9115: CA        668          DEX
9116: E8        669  INS2     INX
9117: EC F9 95  670          CPX  MAXLEN
```

LISTING 1: WINDOW.MAGIC *(continued)*

```
911A: F0 36     671          BEQ  INSOUT
911C: E8        672          INX
911D: 8E F7 95  673          STX  LEN
9120: CA        674 IMOVELOP DEX                ;MOVE CHARS
9121: BD 00 02  675          LDA  IN,X          ;UPWARD
9124: E8        676          INX                ;IN BUFFER
9125: 9D 00 02  677          STA  IN,X
9128: CA        678          DEX
9129: EC F6 95  679          CPX  CURS
912C: D0 F2     680          BNE  IMOVELOP
912E: AD F6 95  681          LDA  CURS
9131: 48        682          PHA
9132: AA        683          TAX
9133: A9 A0     684          LDA  #$A0          ;STORE SPACE OR
9135: C0 04     685          CPY  #$04          ;ZERO FOR # ENTRY
9137: 90 02     686          BCC  INS3          ;AT CURSOR
9139: A9 B0     687          LDA  #"0"
913B: 9D 00 02  688 INS3     STA  IN,X
913E: BD 00 02  689 IPRLOOP  LDA  IN,X          ;PRINT OUT BUFFER
9141: 20 DB 92  690          JSR  COUT
9144: AE F6 95  691          LDX  CURS
9147: EC F7 95  692          CPX  LEN
914A: D0 F2     693          BNE  IPRLOOP
914C: 68        694          PLA
914D: 85 3C     695          STA  A1L
914F: 20 90 91  696          JSR  BACKLOOP      ;BACK UP CURSOR
9152: 4C A8 8F  697 INSOUT   JMP  RDKEY
9155: C9 8F     698 NEXTK    CMP  #$8F          ;^O?
9157: D0 03     699          BNE  NEXTL
9159: 20 6D 92  700 INBED    JSR  GETCHAR       ;GET ANY CHAR
915C: AE F6 95  701 NEXTL    LDX  CURS          ;DON'T USE CHAR
915F: EC F9 95  702          CPX  MAXLEN        ;IF CURS=MAXLEN
9162: 90 03     703          BCC  IN0
9164: 4C A8 8F  704 GRDKEY1  JMP  RDKEY
9167: 20 B3 91  705 IN0      JSR  CHK           ;CHECK IF CHAR IS OK
916A: B0 F8     706          BCS  GRDKEY1
916C: 9D 00 02  707          STA  IN,X          ;PUT IT IN BUFFER.
916F: EC F7 95  708          CPX  LEN           ;INC. LEN IF
9172: 90 03     709          BCC  IN1           ;CURSOR POSITION
9174: EE F7 95  710          INC  LEN           ;IS>LEN
9177: 20 DB 92  711 IN1      JSR  COUT          ;PRINT IT
917A: 4C A8 8F  712          JMP  RDKEY         ;GO BACK FOR MORE.
917D: AE F6 95  713 LEAVE    LDX  CURS          ;THIS COLLECTS
9180: 20 39 D5  714          JSR  GDBUFS        ;INPUT AND
9183: C8        715          INY                ;RETURNS TO
9184: A2 8D     716          LDX  #$8D          ;APPLESOFT
9186: 20 E9 E3  717          JSR  $E3E9
9189: 4C 9A DA  718          JMP  $DA9A
918C: A9 00     719 BEGINSUB LDA  #$00          ;BACK UP CURSOR
918E: 85 3C     720          STA  A1L           ;TO BEGINNING
9190: AD F6 95  721 BACKLOOP LDA  CURS          ;BACK UP CURSOR
9193: C5 3C     722          CMP  A1L           ;TO A1L
9195: F0 06     723          BEQ  BACKLOUT
9197: 20 7D 92  724          JSR  BACK
919A: 4C 90 91  725          JMP  BACKLOOP
919D: 60        726 BACKLOUT RTS
919E: AC F7 95  727 REDUCE   LDY  LEN           ;FLUSH
91A1: C0 00     728 REDLOOP  CPY  #$00          ;TRAILING
91A3: F0 00     729          BEQ  REDOUT        ;SPACES.
91A5: 88        730          DEY
91A6: B9 00 02  731          LDA  IN,Y
91A9: C9 A0     732          CMP  #$A0
91AB: D0 05     733          BNE  REDOUT
91AD: CE F7 95  734          DEC  LEN
91B0: D0 EF     735          BNE  REDLOOP
91B2: 60        736 REDOUT   RTS
91B3: AC FA 95  737 CHK      LDY  MODE          ;CHECK IF CHARACTER
91B6: F0 17     738          BEQ  CHK2          ;IS OK WITH
91B8: C9 A0     739          CMP  #$A0          ;USER MODE
91BA: B0 09     740          BCS  CHK1          ;IF NOT, IT CONVERTS
91BC: C0 01     741          CPY  #$01          ;IT OR RETURNS
91BE: F0 0F     742          BEQ  CHK2          ;WITH CARRY SET
91C0: 18        743          CLC                ;INDICATING
91C1: 69 40     744          ADC  #$40          ;IT'S IMPOSSIBLE
91C3: D0 0A     745          BNE  CHK2
91C5: C9 E0     746 CHK1     CMP  #$E0
91C7: 90 06     747          BCC  CHK2
91C9: C0 02     748          CPY  #$02
91CB: F0 02     749          BEQ  CHK2
91CD: 29 DF     750          AND  #$DF
91CF: C0 04     751 CHK2     CPY  #$04
91D1: B0 01     752          BCS  CHK3
91D3: 60        753          RTS
91D4: C9 B0     754 CHK3     CMP  #$B0
91D6: 90 05     755          BCC  CHK4
91D8: C9 BA     756          CMP  #$BA
91DA: B0 01     757          BCS  CHK4
91DC: 60        758          RTS
91DD: C0 05     759 CHK4     CPY  #$05
91DF: 90 10     760          BCC  CHKNO
91E1: C9 AD     761          CMP  #"-"
91E3: F0 21     762          BEQ  CHK6
91E5: C9 AB     763          CMP  #"+"
91E7: F0 1D     764          BEQ  CHK6
91E9: C0 05     765          CPY  #$05
91EB: F0 04     766          BEQ  CHKNO
91ED: C9 AE     767          CMP  #"."
91EF: F0 02     768          BEQ  CHK5
91F1: 38        769 CHKNO    SEC
91F2: 60        770          RTS
91F3: AC F7 95  771 CHK5     LDY  LEN
91F6: 88        772 CHKL     DEY
91F7: B9 00 02  773          LDA  IN,Y
91FA: C9 AE     774          CMP  #"."
91FC: F0 F3     775          BEQ  CHKNO
91FE: C0 00     776          CPY  #$00
9200: D0 F4     777          BNE  CHKL
9202: A9 AE     778          LDA  #"."
9204: 18        779          CLC
9205: 60        780          RTS
9206: AE F6 95  781 CHK6     LDX  CURS

9209: D0 E6     782          BNE  CHKNO
920B: 18        783          CLC
920C: 60        784          RTS
920D: 20 D6 94  785 GETCHAR  JSR  GETMEMCU      ;MAKE A CURSOR.
9210: B1 06     786          LDA  (PTR1),Y
9212: 85 3C     787          STA  A1L
9214: 2C EB 95  788          BIT  IIC           ;CHECK FOR //E OR //C
9217: F0 1A     789          BEQ  CURS40        ;II PLUS--80 COL. NOT SUPPORTED
9219: 2C 1E C0  790          BIT  RDALTCHR      ;CHECK FOR 80-COL FIRMWARE
921C: 10 15     791          BPL  CURS40
921E: 29 7F     792          AND  #$7F          ;INVERSE MOST CHARS
9220: 2C EB 95  793          BIT  IIC           ;CHECK FOR //C OR ENH. //E
9223: 50 18     794          BVC  CURSOUT       ;OLD //E
9225: B8        795          CLV
9226: C9 40     796          CMP  #$40          ;CHARS $40-$5F
9228: 90 13     797          BCC  CURSOUT       ;(MOSTLY CAPS)
922A: C9 60     798          CMP  #$60          ;MUST BE CONVERTED
922C: B0 0F     799          BCS  CURSOUT       ;TO $00-$1F
922E: 29 1F     800          AND  #$1F          ;$40-$5F IS MOUSETEXT!
9230: 4C 3D 92  801          JMP  CURSOUT
9233: C9 E0     802 CURS40   CMP  #$E0          ;W/O 80-COL FIRMWARE
9235: 90 02     803          BCC  CURS40A       ;FLASHING CURSOR
9237: 29 DF     804          AND  #$DF          ;CAN BE USED
9239: 29 3F     805 CURS40A  AND  #$3F          ;LOWERCASE MUST BE
923B: 09 40     806          ORA  #$40          ;UPPERCASE FLASHING
923D: 91 06     807 CURSOUT  STA  (PTR1),Y
923F: 20 35 94  808          JSR  SETCHAR
9242: AD 00 C0  809 GETKEY   LDA  KBD           ;GET A KEYSTROKE.
9245: 10 FB     810          BPL  GETKEY
9247: 8D 10 C0  811          STA  STROBE
924A: 85 3D     812          STA  A1H
924C: 20 D6 94  813          JSR  GETMEMCU
924F: A5 3C     814          LDA  A1L
9251: 91 06     815          STA  (PTR1),Y      ;REMOVE CURSOR
9253: 20 35 94  816          JSR  SETCHAR
9256: A5 3D     817          LDA  A1H
9258: 60        818          RTS                ;EXIT
9259: A2 03     819 GETKEY2  LDX  #$03          ;REDEFINE KEYS
925B: CA        820 DEFKEYL  DEX                ;KEYBOARD FOR
925C: 30 08     821          BMI  DEFKEYOT      ;FOR BACKSLASHES,ETC
925E: DD 77 92  822          CMP  KEYLIST,X
9261: D0 F8     823          BNE  DEFKEYL
9263: BD 7A 92  824          LDA  REPLACEL,X
9266: AC FB 95  825 DEFKEYOT LDY  CASE          ;SHIFT CHAR
9269: D0 0B     826          BNE  GKEYOUT       ;TO LOWER CASE
926B: C9 E0     827          CMP  #$E0          ;IF NECESSARY
926D: B0 07     828          BCS  GKEYOUT
926F: C9 C0     829          CMP  #$C0
9271: 90 03     830          BCC  GKEYOUT
9273: 18        831          CLC
9274: 69 20     832          ADC  #$20
9276: 60        833 GKEYOUT  RTS
9277: 9A 98 83  834 KEYLIST  HEX  9A9883
927A: DB DC DF  835 REPLACEL HEX  DBDCDF
927D: 20 4E 93  836 BACK     JSR  SHRINK        ;BACK UP CURSOR
9280: A5 17     837          LDA  CH            ;AND DEC
9282: CD ED 95  838          CMP  POSH          ;CURS
9285: D0 07     839          BNE  BACK1
9287: AD F2 95  840          LDA  POS1H
928A: 85 17     841          STA  CH
928C: C6 18     842          DEC  CV
928E: C6 17     843 BACK1    DEC  CH
9290: CE F6 95  844          DEC  CURS
9293: 20 60 93  845          JSR  EXPAND
9296: 60        846          RTS
                847 *
9297: AD F5 95  848 GOTO     LDA  CURWIND       ;DON'T MOVE
929A: D0 03     849          BNE  GOTO0         ;WINDOW 0.
929C: 4C 3D 8C  850          JMP  GSYNERR
929F: A2 00     851 GOTO0    LDX  #$00          ;GET PARAMTERS
92A1: 86 3D     852          STX  A1H
92A3: 86 3C     853          STX  A1L
92A5: CA        854          DEX
92A6: 86 3E     855          STX  A2L
92A8: 20 B9 94  856          JSR  GETNUMPR
92AB: 48        857          PHA                ;ADJUST
92AC: 18        858          CLC                ;CURSOR POSITION
92AD: 65 17     859          ADC  CH
92AF: 38        860          SEC
92B0: ED ED 95  861          SBC  POSH
92B3: 85 17     862          STA  CH
92B5: 68        863          PLA
92B6: 8D ED 95  864          STA  POSH
92B9: E6 3C     865          INC  A1L
92BB: 20 B9 94  866          JSR  GETNUMPR      ;GET 2ND
92BE: 48        867          PHA                ;PARAMETER
92BF: 18        868          CLC                ;ADJUST CURSOR
92C0: 65 18     869          ADC  CV            ;POSITION.
92C2: 38        870          SEC
92C3: ED EE 95  871          SBC  POSV
92C6: 85 18     872          STA  CV
92C8: 68        873          PLA
92C9: 8D EE 95  874          STA  POSV
92CC: 20 15 93  875          JSR  SAVEDESC      ;SAVE CHANGES.
92CF: 20 00 93  876          JSR  POSICALC
92D2: 4C F4 94  877          JMP  FASTDUMP      ;DUMP SCREEN.
                878 *
92D5: 20 F4 94  879 RESTORE  JSR  FASTDUMP      ;DUMP SCREEN
92D8: 4C EA 8D  880          JMP  HOMECURS      ;AND HOME CURSOR.
                881 *
92DB: 09 80     882 COUT     ORA  #$80          ;PRINT
92DD: C9 A0     883          CMP  #$A0          ;CHAR IN A
92DF: B0 02     884          BCS  COUT1         ;CONTROL
92E1: 29 3F     885          AND  #$3F          ;CHARS IN
92E3: 25 32     886 COUT1    AND  INVFLG        ;INVERSE.
92E5: 05 F3     887          ORA  $F3
92E7: 48        888          PHA                ;ADVANCE
92E8: 20 D6 94  889          JSR  GETMEMCU      ;CURSOR
92EB: 68        890          PLA
92EC: 91 06     891          STA  (PTR1),Y
92EE: 20 35 94  892          JSR  SETCHAR
```

LISTING 1: WINDOW.MAGIC *(continued)*

```
92F1: 4C 22 93   893             JMP   ADVANCE
                 894 *
92F4: A0 06      895 LOADDESC    LDY   #$06        ;GETS DESCRIPTOR
92F6: B1 04      896 LOADDES1    LDA   (PTR),Y     ;EXCEPT
92F8: 99 EB 95   897             STA   POSH-2,Y    ;CH & CV.
92FB: 88         898             DEY
92FC: C0 01      899             CPY   #$01
92FE: D0 F6      900             BNE   LOADDES1
9300: 18         901 POSICALC    CLC
9301: AD ED 95   902             LDA   POSH
9304: 6D EF 95   903             ADC   LENH
9307: 8D F2 95   904             STA   POS1H
930A: 18         905             CLC
930B: AD EE 95   906             LDA   POSV
930E: 6D F0 95   907             ADC   LENV
9311: 8D F3 95   908             STA   POS1V
9314: 60         909             RTS
                 910 *
9315: A0 02      911 SAVEDESC    LDY   #$02        ;SAVE
9317: B9 EB 95   912 SAVEDES     LDA   POSH-2,Y    ;DESCRIPTOR
931A: 91 04      913             STA   (PTR),Y     ;EXCEPT
931C: C8         914             INY               ;FOR CH
931D: C0 07      915             CPY   #$07        ;AND CV.
931F: D0 F5      916             BNE   SAVEDES
9321: 60         917             RTS
                 918 *
9322: EE F6 95   919 ADVANCE     INC   CURS        ;ADVANCE
9325: 20 4E 93   920             JSR   SHRINK      ;CURSOR.
9328: E6 17      921             INC   CH
932A: A5 17      922             LDA   CH
932C: CD F2 95   923             CMP   POS1H
932F: D0 14      924             BNE   ADVOUT
9331: AD ED 95   925 ADVANCE1    LDA   POSH
9334: 85 17      926             STA   CH
9336: E6 18      927             INC   CV
9338: A5 18      928             LDA   CV
933A: CD F3 95   929             CMP   POS1V
933D: D0 06      930             BNE   ADVOUT
933F: 20 72 93   931             JSR   SCROLL      ;SCROLL IF NEEDED
9342: C6 18      932             DEC   CV
9344: 60         933             RTS
9345: 4C 60 93   934 ADVOUT      JMP   EXPAND
                 935 *
9348: 20 4E 93   936 CR          JSR   SHRINK      ;CARRIAGE
934B: 4C 31 93   937             JMP   ADVANCE1    ;RETURN.
                 938 *
934E: AD F1 95   939 SHRINK      LDA   BORDER      ;IF THERE
9351: F0 0C      940             BEQ   SHRIKOUT    ;IS A BORDER,
9353: EE ED 95   941             INC   POSH        ;SHRINK
9356: EE EE 95   942             INC   POSV        ;WINDOW BY
9359: CE F2 95   943             DEC   POS1H       ;ONE IN
935C: CE F3 95   944             DEC   POS1V       ;ALL DIRECTIONS
935F: 60         945 SHRIKOUT    RTS
                 946 *
9360: AD F1 95   947 EXPAND      LDA   BORDER      ;OPPOSITE
9363: F0 0C      948             BEQ   EXPANOUT    ;OF SHRINK
9365: CE ED 95   949             DEC   POSH
9368: CE EE 95   950             DEC   POSV
936B: EE F2 95   951             INC   POS1H
936E: EE F3 95   952             INC   POS1V
9371: 60         953 EXPANOUT    RTS
                 954 *
9372: A2 00      955 SCROLL      LDX   #$00        ;SET UP
9374: AD F1 95   956             LDA   BORDER      ;PARAMETERS
9377: F0 01      957             BEQ   SC1         ;FOR MONITOR
9379: E8         958             INX               ;MEMORY
937A: 86 3C      959 SC1         STX   A1L         ;MOVE.
937C: 20 14 94   960             JSR   GETMEMAD
937F: A5 06      961             LDA   PTR1
9381: 85 42      962             STA   A4L
9383: A5 07      963             LDA   PTR1+1
9385: 85 43      964             STA   A4H
9387: A6 3C      965             LDX   A1L
9389: E8         966             INX
938A: 20 14 94   967             JSR   GETMEMAD
938D: A5 06      968             LDA   PTR1
938F: 85 3C      969             STA   A1L
9391: A5 07      970             LDA   PTR1+1
9393: 85 3D      971             STA   A1H
9395: AE F0 95   972             LDX   LENV
9398: 20 14 94   973             JSR   GETMEMAD
939B: 38         974             SEC
939C: A5 06      975             LDA   PTR1
939E: E9 01      976             SBC   #$01
93A0: 85 3E      977             STA   A2L
93A2: A5 07      978             LDA   PTR1+1
93A4: E9 00      979             SBC   #$00
93A6: 85 3F      980             STA   A2H
93A8: A0 00      981             LDY   #$00
93AA: 20 2C FE   982             JSR   MOVE        ;MOVE IT.
93AD: AE F0 95   983             LDX   LENV
93B0: CA         984             DEX
93B1: AD F1 95   985             LDA   BORDER
93B4: F0 01      986             BEQ   SC25
93B6: CA         987             DEX
93B7: 20 14 94   988 SC25        JSR   GETMEMAD
93BA: AC EF 95   989             LDY   LENH
93BD: 84 3C      990             STY   A1L
93BF: A0 00      991             LDY   #$00
93C1: AD F1 95   992             LDA   BORDER
93C4: F0 03      993             BEQ   SC3
93C6: C8         994             INY
93C7: C6 3C      995             DEC   A1L
93C9: A9 A0      996 SC3         LDA   #$A0        ;BLANK OUT
93CB: 91 06      997 SCRLOOP     STA   (PTR1),Y    ;LAST LINE.
93CD: C8         998             INY
93CE: C4 3C      999             CPY   A1L
93D0: D0 F9      1000            BNE   SCRLOOP
93D2: 4C F4 94   1001            JMP   FASTDUMP    ;DUMP SCREEN.
                 1002 *
93D5: AE F5 95   1003 GWINADD    LDX   CURWIND     ;CALCULATE
93D8: 18         1004 GWINADD1   CLC               ;ADDRESS OF WINDOW
93D9: AD FE 95   1005            LDA   MEMBOT      ;BY ADDING
93DC: 69 C0      1006            ADC   #$C0        ;ADD 960
93DE: 85 04      1007            STA   PTR
93E0: AD FF 95   1008            LDA   MEMBOT+1
93E3: 69 03      1009            ADC   #$03
93E5: 85 05      1010            STA   PTR+1
93E7: 2C EA 95   1011            BIT   COL80
93EA: 10 0D      1012            BPL   ONLY40
93EC: 18         1013            CLC               ;FOR 80 COL.
93ED: A5 04      1014            LDA   PTR         ;ADD ANOTHER 960
93EF: 69 C0      1015            ADC   #$C0
93F1: 85 04      1016            STA   PTR
93F3: A5 05      1017            LDA   PTR+1
93F5: 69 03      1018            ADC   #$03
93F7: 85 05      1019            STA   PTR+1
93F9: E0 00      1020 ONLY40     CPX   #$0
93FB: F0 16      1021            BEQ   GWINOUT
93FD: A0 00      1022            LDY   #$00
93FF: 18         1023 GWINLOOP   CLC
9400: B1 04      1024            LDA   (PTR),Y
9402: 65 04      1025            ADC   PTR
9404: 48         1026            PHA
9405: C8         1027            INY
9406: B1 04      1028            LDA   (PTR),Y
9408: 65 05      1029            ADC   PTR+1
940A: 85 05      1030            STA   PTR+1
940C: 68         1031            PLA
940D: 85 04      1032            STA   PTR
940F: 88         1033            DEY
9410: CA         1034            DEX
9411: D0 EC      1035            BNE   GWINLOOP
9413: 60         1036 GWINOUT    RTS
                 1037 *
9414: 18         1038 GETMEMAD   CLC               ;USES PTR
9415: A5 04      1039            LDA   PTR         ;TO FIND
9417: 69 09      1040            ADC   #$09        ;ADDRESS
9419: 85 06      1041            STA   PTR1        ;OF THE START
941B: A5 05      1042            LDA   PTR+1       ;OF A ROW
941D: 69 00      1043            ADC   #$00        ;RESULT PUT
941F: 85 07      1044            STA   PTR1+1      ;IN PTR1.
9421: E0 00      1045            CPX   #$00
9423: F0 0F      1046            BEQ   GETMEOUT
9425: 18         1047 GETMLOOP   CLC
9426: A5 06      1048            LDA   PTR1
9428: 6D EF 95   1049            ADC   LENH
942B: 85 06      1050            STA   PTR1
942D: 90 02      1051            BCC   GETMNEXT
942F: E6 07      1052            INC   PTR1+1
9431: CA         1053 GETMNEXT   DEX
9432: D0 F1      1054            BNE   GETMLOOP
9434: 60         1055 GETMEOUT   RTS
                 1056 *
9435: A5 17      1057 SETCHAR    LDA   CH          ;FIND WHICH
9437: CD EC 95   1058            CMP   NOCOLS      ;WINDOW CONTROLS
943A: B0 0D      1059            BCS   SETCHOUT    ;POSITION
943C: A5 18      1060            LDA   CV          ;CH,CV.
943E: C9 18      1061            CMP   #24         ;GET CHAR
9440: B0 07      1062            BCS   SETCHOUT    ;FROM THAT WINDOW
9442: AD F4 95   1063            LDA   NUMWIND
9445: 85 42      1064            STA   A4L
9447: D0 43      1065            BNE   WINDFNXT    ;ALWAYS
9449: 60         1066 SETCHOUT   RTS
944A: 20 D8 93   1067 WINDFIND   JSR   GWINADD1
944D: 20 F4 92   1068            JSR   LOADDESC
9450: AD ED 95   1069            LDA   POSH
9453: 30 0E      1070            BMI   WNDFIND1
9455: A5 17      1071            LDA   CH
9457: CD ED 95   1072            CMP   POSH
945A: 90 30      1073            BCC   WINDFNXT
945C: CD F2 95   1074            CMP   POS1H
945F: B0 2B      1075            BCS   WINDFNXT
9461: 90 0B      1076            BCC   WNDFIND2
9463: AD F2 95   1077 WNDFIND1   LDA   POS1H
9466: 30 24      1078            BMI   WINDFNXT
9468: C5 17      1079            CMP   CH
946A: 90 20      1080            BCC   WINDFNXT
946C: F0 1E      1081            BEQ   WINDFNXT
946E: AD EE 95   1082 WNDFIND2   LDA   POSV
9471: 30 0E      1083            BMI   WNDFIND3
9473: A5 18      1084            LDA   CV
9475: CD EE 95   1085            CMP   POSV
9478: 90 12      1086            BCC   WINDFNXT
947A: CD F3 95   1087            CMP   POS1V
947D: B0 0D      1088            BCS   WINDFNXT
947F: 90 11      1089            BCC   WINFOUND
9481: AD F3 95   1090 WNDFIND3   LDA   POS1V
9484: 30 06      1091            BMI   WINDFNXT
9486: C5 18      1092            CMP   CV
9488: F0 02      1093            BEQ   WINDFNXT
948A: B0 06      1094            BCS   WINFOUND
948C: C6 42      1095 WINDFNXT   DEC   A4L
948E: A6 42      1096            LDX   A4L
9490: 10 B8      1097            BPL   WINDFIND
9492: A5 18      1098 WINFOUND   LDA   CV          ;PRINT
9494: 20 C1 FB   1099            JSR   BASCALC     ;THAT CHAR.
9497: 20 D6 94   1100            JSR   GETMEMCU
949A: B1 06      1101            LDA   (PTR1),Y
949C: A4 17      1102            LDY   CH
949E: 2C EA 95   1103            BIT   COL80       ;CHECK FOR 80 COL
94A1: 10 0E      1104            BPL   ONWARD
94A3: 48         1105            PHA               ;SAVE A
94A4: 98         1106            TYA
94A5: 4A         1107            LSR               ;DIVIDE BY 2
94A6: A8         1108            TAY
94A7: A9 00      1109            LDA   #$0         ;CARRY=ODD/EVEN
94A9: 2A         1110            ROL
94AA: 49 01      1111            EOR   #$01
94AC: AA         1112            TAX               ;X=1 EVEN,X=0 ODD
94AD: BD 54 C0   1113            LDA   PAGE1,X     ;SWITCH IN PROPER PAGE
94B0: 68         1114            PLA               ;RESTORE A
94B1: 91 28      1115 ONWARD     STA   (BASL),Y
94B3: 20 D5 93   1116            JSR   GWINADD     ;RESTORE CURRENT
```

LISTING 1: WINDOW.MAGIC *(continued)*

```
94B6: 4C F4 92  1117            JMP  LOADDESC     ;WINDOW AND EXIT
                1118 *
94B9: 20 67 DD  1119 GETNUMPR   JSR  FRMNUM       ;GETS NUMBER
94BC: 20 FB E6  1120            JSR  CONINT       ;FROM APPLESOFT
94BF: A5 3C     1121            LDA  A1L          ;SURE A COMMA FOLLOWS
94C1: D0 03     1122            BNE  GETNUMP1
94C3: 20 BE DE  1123            JSR  CHKCOM
94C6: A5 A1     1124 GETNUMP1   LDA  FACLO        ;MAKE SURE IT'S < A2L
94C8: C5 3D     1125            CMP  A1H          ;AND > A1H
94CA: 90 07     1126            BCC  GRANGERR
94CC: C5 3E     1127            CMP  A2L
94CE: F0 02     1128            BEQ  GETNUOUT
94D0: B0 01     1129            BCS  GRANGERR
94D2: 60        1130 GETNUOUT   RTS               ;EXIT
                1131 *
94D3: 4C 99 E1  1132 GRANGERR   JMP  RANGERR      ;ILLEGAL QUANT ERROR
                1133 *
94D6: 38        1134 GETMEMCU   SEC               ;GETS LOCATION
94D7: A5 18     1135            LDA  CV           ;IN WINDOW
94D9: ED EE 95  1136            SBC  POSV         ;FROM CURSOR
94DC: AA        1137            TAX               ;POSITION
94DD: 20 14 94  1138            JSR  GETMEMAD
94E0: 38        1139            SEC
94E1: A5 17     1140            LDA  CH
94E3: ED ED 95  1141            SBC  POSH
94E6: A8        1142            TAY
94E7: 60        1143            RTS
                1144 *
94E8: A0 07     1145 SAVEPOS    LDY  #$07         ;SAVES CURSOR
94EA: A5 17     1146            LDA  CH           ;POSITION
94EC: 91 04     1147            STA  (PTR),Y      ;INTO
94EE: C8        1148            INY               ;DESCRIPTOR
94EF: A5 18     1149            LDA  CV
94F1: 91 04     1150            STA  (PTR),Y
94F3: 60        1151            RTS
                1152 *
94F4: A2 00     1153 FASTDUMP   LDX  #$00         ;DUMP ALL
94F6: 86 3E     1154            STX  A2L          ;WINDOWS FROM
94F8: 20 D8 93  1155 WINDLOOP   JSR  GWINADD1     ;0 TO NUMWIND
94FB: 20 F4 92  1156            JSR  LOADDESC     ;INTO SCRATCH
94FE: AD ED 95  1157            LDA  POSH
9501: CD EC 95  1158            CMP  NOCOLS
9504: 90 03     1159            BCC  WNDLP1
9506: A9 00     1160            LDA  #$00
9508: 18        1161            CLC
9509: 6D FE 95  1162 WNDLP1     ADC  MEMBOT
950C: 85 3C     1163            STA  A1L
950E: AD FF 95  1164            LDA  MEMBOT+1
9511: 69 00     1165            ADC  #$00
9513: 85 3D     1166            STA  A1H
9515: AE EE 95  1167            LDX  POSV
9518: CA        1168            DEX
9519: 86 3F     1169            STX  A2H
951B: E8        1170            INX
951C: F0 53     1171            BEQ  NEXTROW
951E: E0 18     1172            CPX  #24
9520: 90 0B     1173            BCC  ADDROW
9522: EC F3 95  1174            CPX  POS1V
9525: 90 4A     1175            BCC  NEXTROW
9527: A9 FF     1176            LDA  #$FF
9529: 85 3F     1177            STA  A2H
952B: D0 44     1178            BNE  NEXTROW
952D: 20 DD 95  1179 ADDROW     JSR  ADD28
9530: CA        1180            DEX
9531: D0 FA     1181            BNE  ADDROW
9533: F0 3C     1182            BEQ  NEXTROW
9535: 38        1183 ROWLOOP    SEC
9536: ED EE 95  1184            SBC  POSV
9539: AA        1185            TAX
953A: 20 14 94  1186            JSR  GETMEMAD
953D: A0 00     1187            LDY  #$00
953F: AE ED 95  1188            LDX  POSH
9542: EC EC 95  1189            CPX  NOCOLS
9545: 90 15     1190            BCC  COLULOOP
9547: EC F2 95  1191            CPX  POS1H
954A: 90 22     1192            BCC  COLOUT
954C: 8A        1193            TXA
954D: 49 FF     1194            EOR  #$FF
954F: 18        1195            CLC
9550: 69 01     1196            ADC  #$01
9552: 65 06     1197            ADC  PTR1
9554: 85 06     1198            STA  PTR1
9556: A2 00     1199            LDX  #$00
9558: 90 02     1200            BCC  COLULOOP
955A: E6 07     1201            INC  PTR1+1
955C: EC EC 95  1202 COLULOOP   CPX  NOCOLS
955F: B0 0D     1203            BCS  COLOUT
9561: EC F2 95  1204            CPX  POS1H
9564: B0 08     1205            BCS  COLOUT
9566: B1 06     1206            LDA  (PTR1),Y
9568: 91 3C     1207            STA  (A1L),Y
956A: E8        1208            INX
956B: C8        1209            INY
956C: D0 EE     1210            BNE  COLULOOP
956E: 20 DD 95  1211 COLOUT     JSR  ADD28
9571: E6 3F     1212 NEXTROW    INC  A2H
9573: A5 3F     1213            LDA  A2H
9575: C9 18     1214            CMP  #$18
9577: B0 05     1215            BCS  NEXTWIND
9579: CD F3 95  1216            CMP  POS1V
957C: 90 B7     1217            BCC  ROWLOOP
957E: E6 3E     1218 NEXTWIND   INC  A2L
9580: A6 3E     1219            LDX  A2L
9582: EC F4 95  1220            CPX  NUMWIND
9585: B0 03     1221            BCS  NEXTWND1
9587: 4C F8 94  1222            JMP  WINDLOOP
958A: AD FE 95  1223 NEXTWND1   LDA  MEMBOT       ;THEN DUMP
958D: 85 3C     1224            STA  A1L          ;SCRATCH
958F: AD FF 95  1225            LDA  MEMBOT+1     ;BUFFER
9592: 85 3D     1226            STA  A1H          ;DIRECTLY ONTO
9594: A9 00     1227            LDA  #$0          ;SCREEN
```

LISTING 1: WINDOW.MAGIC *(continued)*

```
9596: 85 3E     1228            STA  A2L
9598: 20 C1 FB  1229 MOVES1     JSR  BASCALC
959B: A5 28     1230            LDA  BASL         ;TRANSFER TO SAFE ZP LOCS
959D: 85 FB     1231            STA  ZP
959F: A5 29     1232            LDA  BASL+1
95A1: 85 FC     1233            STA  ZP+1
95A3: A2 00     1234            LDX  #$0
95A5: A0 00     1235            LDY  #$0
95A7: 2C EA 95  1236 MOVES2     BIT  COL80        ;CHECK FOR 80 COL
95AA: 10 10     1237            BPL  SKIP
95AC: 2C 55 C0  1238            BIT  PAGE2        ;ENABLE AUX MEM
95AF: B1 3C     1239            LDA  (A1L),Y      ;GET CHAR FROM SCRATCH BUFFER
95B1: 81 FB     1240            STA  (ZP,X)       ;PUT ON THE SCREEN
95B3: 2C 54 C0  1241            BIT  PAGE1
95B6: C8        1242            INY
95B7: CC EC 95  1243            CPY  NOCOLS       ;RIGHT EDGE?
95BA: B0 10     1244            BCS  INCROW
95BC: B1 3C     1245 SKIP       LDA  (A1L),Y      ;GET CHAR FROM BUFFER
95BE: 81 FB     1246            STA  (ZP,X)       ;PUT ON THE SCREEN
95C0: E6 FB     1247            INC  ZP           ;NEXT SCREEN LOC
95C2: D0 02     1248            BNE  SKIP2
95C4: E6 FC     1249            INC  ZP+1
95C6: C8        1250 SKIP2      INY               ;NEXT
95C7: CC EC 95  1251            CPY  NOCOLS
95CA: 90 DB     1252            BCC  MOVES2
95CC: 20 DD 95  1253 INCROW     JSR  ADD28
95CF: E6 3E     1254            INC  A2L
95D1: A5 3E     1255            LDA  A2L
95D3: C9 18     1256            CMP  #$18
95D5: 90 C1     1257            BCC  MOVES1
95D7: 20 D5 93  1258            JSR  GWINADD
95DA: 4C F4 92  1259            JMP  LOADDESC
95DD: 18        1260 ADD28      CLC
95DE: A5 3C     1261            LDA  A1L
95E0: 6D EC 95  1262            ADC  NOCOLS
95E3: 85 3C     1263            STA  A1L
95E5: 90 02     1264            BCC  ADD28OUT
95E7: E6 3D     1265            INC  A1H
95E9: 60        1266 ADD28OUT   RTS               ;EXIT
                1267 *
95EA: 00        1268 COL80      HEX  00           ;BIT $80=80 COL
95EB: 00        1269 IIC        HEX  00           ;BIT $80=//C OR //E $40=//C OR ENH. //E
95EC: 28        1270 NOCOLS     HEX  28
95ED: 00        1271 POSH       HEX  00           ;DESCRIPTOR
95EE: 00        1272 POSV       HEX  00           ;OF CURRENT WINDOW
95EF: 28        1273 LENH       HEX  28           ;KEPT HERE
95F0: 18        1274 LENV       HEX  18
95F1: 00        1275 BORDER     HEX  00
95F2: 00        1276 POS1H      HEX  00           ;COORDINATE OF BOTTOM
95F3: 00        1277 POS1V      HEX  00           ;RIGHT CORNER
95F4: 00        1278 NUMWIND    HEX  00           ;# OF WINDOWS
95F5: 00        1279 CURWIND    HEX  00           ;CURRENT WINDOW
95F6: 00        1280 CURS       HEX  00           ;CURSOR INDEX
95F7: 00        1281 LEN        HEX  00           ;LENGTH OF STRING
95F8: 00        1282 SEMI       HEX  00           ;SEMICOLON FLAG
95F9: 00        1283 MAXLEN     HEX  00           ;MAXIMUM STRING LENGTH
95FA: 00        1284 MODE       HEX  00           ;USER INPUT MODE
95FB: 00        1285 CASE       HEX  00           ;CASE FLAG
95FC: 33 8C     1286 MEMTOP     DA   START        ;TOP AND BOTTOM
95FE: 00 78     1287 MEMBOT     DA   $7800        ;OF WINDOW MEMORY

--End assembly--

2584 bytes

Errors: 0
```

END OF LISTING 1

```
KEY  PERFECT
RUN ON
WINDOW.MAGIC
=================================
CODE-5.0   ADDR#  - ADDR#   CODE-4.0
--------   ------ - ------  --------
7EF6C3C5   8BE8 - 8C37      2ACD
4D0E667A   8C38 - 8C87      2843
AFFAF665   8C88 - 8CD7      2842
BDA4B8CC   8CD8 - 8D27      2634
A7C8002F   8D28 - 8D77      26C0
6A0A6A01   8D78 - 8DC7      2707
888CD01F   8DC8 - 8E17      265E
FC2ABF25   8E18 - 8E67      23EA
FDF4962D   8E68 - 8EB7      28C5
4C5DDE0B   8EB8 - 8F07      298F
DEF568C8   8F08 - 8F57      2A87
8E9CEACF   8F58 - 8FA7      2C30
B3E2B484   8FA8 - 8FF7      23CC
25A0A0C0   8FF8 - 9047      2467
633F9FC2   9048 - 9097      279C
83C3BC90   9098 - 90E7      27FD
872A378B   90E8 - 9137      293B
2A260333   9138 - 9187      21F2
5156262A   9188 - 91D7      26AA
F6EC05AB   91D8 - 9227      28D3
E95A4C17   9228 - 9277      2B2D
5266FB2D   9278 - 92C7      2433
DDEA728B   92C8 - 9317      2733
64C9AAA3   9318 - 9367      2E27
006771B0   9368 - 93B7      2661
E73084A9   93B8 - 9407      25EB
B161402F   9408 - 9457      2BB4
4E543263   9458 - 94A7      2C70
8051B33F   94A8 - 94F7      24E1
3AE8E467   94F8 - 9547      2426
6D1F2CC5   9548 - 9597      29E9
CB5C4FBF   9598 - 95E7      26DD
5F704D6E   95E8 - 95FF      0A6F
0445C01C = PROGRAM TOTAL =  0A18
```

LISTING 2: WINDOW.DEMO1

```
10  REM ************************
20  REM *     WINDOW.DEMO1       *
30  REM *     BY PAUL NICK       *
40  REM * COPYRIGHT (C) 1985 *
50  REM * BY MICROSPARC, INC *
60  REM * CONCORD, MA  01742 *
70  REM ************************
80  PRINT  CHR$ (4)"BRUN WINDOW.MAGIC": REM
     CALL 35816
90  IF  PEEK (48896) = 76 THEN  POKE 115,0: POKE
     116, PEEK (116) - 4
100  REM  REMOVE 'REM' FROM ABOVE LINE FOR PR
    ODOS
110  &  DEF  PEEK (38380),1,20,3,170
120  &  HOME
130  &  PRINT "   WINDOW MAGIC  ";
140  FOR X =  PEEK (38380) TO 8 STEP  - 1: &
     GOTO X,1: NEXT
150  &  DEF 14,240,3,16,32
160  &  HOME
170  &  PRINT "BY PAUL NICK";
180  FOR Y = 241 TO 255: &  GOTO 14,Y: NEXT
190  FOR Y = 0 TO 4: &  GOTO 14,Y: NEXT
200  &  DEF 234,19,24,5,170
210  &  HOME : &  PRINT "  COPYRIGHT (C) 1985
    ": &  PRINT "  BY MICROSPARC, INC": &  PRINT
    "  CONCORD, MA  01742";
220  FOR X = 235 TO 255: &  GOTO X,19: NEXT
230  FOR X = 0 TO 5: &  GOTO X,19: NEXT
240  &  DEF 18,12,15,6,64: &  HOME : &  PRINT
    : &  PRINT "PRESS SPACE": &  PRINT "TO
    CONTINUE";: &  INPUT A$,7,32
250  &  DEL
260  &  DEF 5,5,25,15,43
270  &  HOME : &  PRINT "INPUT CONTROLS:": &
     PRINT "  ^D--DELETE CHAR": &  PRINT "
    ^I--INSERT CHAR": &  PRINT "  ^Q--ACCEPT
     TO CURSOR": &  PRINT "  ^B--TO BEGINNIN
    G": &  PRINT "  ^N--TO END"
280  &  PRINT : &  PRINT "TYPE ANYTHING:"
290  &  PRINT : &  PRINT ">";: &  INPUT A$,0,
    15
300  &  PRINT : &  PRINT "YOU ENTERED:": &  PRINT
    : &  PRINT A$;
310  FOR I = 1 TO 8: FOR J = 1 TO 400: NEXT :
     &  PRINT : NEXT
320  &  PRINT "PRESS <RETURN> TO QUIT";
330  POKE  - 16368,0: WAIT  - 16384,128
340  FOR I = 4 TO 1 STEP  - 1: FOR J = 1 TO 8
    00: NEXT : &  ON I: &  DEL : NEXT
```

END OF LISTING 2

LISTING 3: WINDOW.DEMO2

```
10  REM ************************
20  REM *     WINDOW.DEMO2       *
30  REM *     BY PAUL NICK       *
40  REM * COPYRIGHT (C) 1985 *
50  REM * BY MICROSPARC, INC *
60  REM * CONCORD, MA  01742 *
70  REM ************************
80  PRINT  CHR$ (4);"BLOAD WINDOW.MAGIC": POKE
     38398,0: POKE 38399,72: CALL 35816
90  IF  PEEK (48896) = 76 THEN  POKE 115,0: POKE
     116, PEEK (116) - 4
100  REM  REMOVE 'REM' FROM ABOVE LINE FOR PR
    ODOS
110  REM      THE ABOVE BLOADS WINDOW MAGIC AN
    D CHANGES MEMBOT TO $6000 BEFORE RUNNING
     IT WITH A CALL.
120  &  DEF 100,0,96,96,32: &  ON 0: &  VTAB
    12: &  HTAB 12: &  PRINT "WAIT ONE MOMEN
    T": &  ON 1
130  FOR X = 1 TO 1980: &  PRINT X" ";: NEXT

140  &  GOTO 0,0: &  ON 0: &  HOME : &  VTAB
    12: &  HTAB 12: &  PRINT "BIG WINDOW DEM
    O": &  ON 1
150  FOR X = 0 TO 255: &  GOTO X,0: IF X = 40
     THEN X = 150
160  NEXT : &  GOTO 0,0
170  FOR X = 0 TO 255: &  GOTO 0,X: IF X = 25
     THEN X = 150
180  NEXT : &  GOTO 0,0
190  FOR X = 1 TO 2000: NEXT : HOME : END
```

END OF LISTING 3