PRODOS QUIT CODE

Sandy has dissected the ProDOS QUIT code and found enough room to install his own, much improved QUIT code — a program selector.

In the past year I have received many letters about the ProDOS QUIT code, which transfers you from one system program to another. Although a few readers wish simply to understand the code more clearly, the vast majority plead for a less cumbersome selector/dispatcher convention. Well, my friends, you've come to the right place. After reviewing and dissecting the QUIT code, I shall provide you with a slick means of gliding smoothly and rapidly between applications. If you have never owned a program selector, you're in for a treat.

MLI QUIT CALL

When the PRODOS file is loaded and relocated, the QUIT code ends up residing at $D100-$D3FF in the second 4K language card (LC) bank. The MLI QUIT vector is located at $BF03 in the System Global Page. An MLI call to QUIT can be made by issuing the BYE command from Applesoft or by the following assembly language sequence:

```
JSR MLI ;call MLI at $BF00
DFB $65 ;QUIT call number
DA QUITPARM ;pointer to QUIT parameter list
.....
QUITPARM DFB 4 ;4 parameters in list
DFB 0,0,0,0 ;all parameters equal zero
```

The classic QUIT call invokes the 40-column screen, which demands that you type the prefix and pathname of the next application. If you cannot remember the exact nomenclature, the system must be rebooted. Like you, I too have come to hate this selection process.

An enhanced QUIT call passes control directly to a designated system program. This call can be made on the IIGS only when the Apple IIGS system disk has been booted. The MLI call is identical, but the parameter list contains an $EE (E for enhanced) QUIT type code and a pointer to a pathname string which follows the ProDOS convention of preceding the positive ASCII characters with a length byte. An example of this enhanced parameter list is:

```
QUITPARM DFB 4 ;4 parameters in list
DFB $EE ;QUIT type
DA PATHNAME ;pointer to next application
DFB 0,0 ;2 zero parameters
PATHNAME DFB $13 ;19 chars in pathname string
ASC '/MYVOL/BASIC.SYSTEM' ;pathname string
```

Versions 1.2 and 1.3* of ProDOS 8 (see Disassembly Lines, Vol. 8/No. 7) support the enhanced QUIT call, but I stress again that the call works only after a system disk boot has installed the ProDOS 16 QUIT code in the LC. It appears that this code remains operational even when a ProDOS 8 application has been selected from the desktop launcher. Do not take this latter statement as gospel, because I have spent little time trying to understand the ProDOS 16 environment; the new versions of ProDOS 8 are at the top of my list right now.

Installation of Quit Code

Listing 1 contains several sections pertaining to the QUIT code. In the loaded PRODOS file, the QUIT code proper is found at $5900-$5BFF for versions 1.2, and 1.3, and at $5700-$59FF for version 1.1.1. The PRODOS segment that relocates the QUIT code is presented in lines **34-54**. After setting the QUIT vector in the System Global Page, and read/write (R/W)-enabling the second LC bank, the X- and Y-Registers are pointed at a table of QUIT code parameters, and the generic ProDOS relocation subroutine copies the QUIT code from the loaded PRODOS file to bank two of the LC. An identification byte is then set, LC bank one is re-enabled,

**Editor's Note: Since this article was completed, Apple has introduced Version 1.4 of ProDOS 8. It's identical to version 1.3 in all the respects discussed here. Version 1.3 was found to crash on the unenhanced IIe and II Plus and should no longer be used. All references to version 1.3 in this article apply to version 1.4 as well.*

errors are trapped, and, if all has gone well, execution of the PRODOS file continues. Because it adds little to the current theme, I have chosen not to disassemble the relocation subroutine.

Execution of MLI QUIT Call

When the MLI recognizes the QUIT call, it passes control to the QUIT vector in the System Global Page (**lines 55-60**) which in turn transfers flow back to the QUIT code handler within the MLI (**lines 61-106**). Here, the second LC bank is R/W-enabled, zero page locations 0-3 are saved on the stack, the three pages of QUIT code are copied from $D100-$D3FF in the LC to $1000-$12FF, the first LC bank is re-enabled, locations 0-3 are restored, the RESET vector is pointed at the relocated code, and control is passed to the QUIT code proper, which has been transferred to low RAM. We are now ready to dissect the working section of the QUIT code, i.e., the selector/dispatcher.

Execution of Quit Code

After enabling motherboard ROM and setting normal 40-column text mode (**lines 114-121**), the selector resets the system bit map to its virgin state, protecting pages 0-1, 4-7 and $BF, and freeing the remainder of RAM (**lines 125-134**). The current prefix is displayed by POKEing the result of the GET_PREFIX call directly into screen memory (**lines 138-157**), and then the cursor is placed at the start of the text (**lines 158-160**).

A new prefix is solicited by the input routine in **lines 164-205**. Immediately pressing Return causes the current prefix to be selected. Striking any other key clears the default prefix. Only valid pathname characters are accepted. The Right-Arrow is disabled, and the Left-Arrow and Delete keys produce a destructive backspace. Escape clears the line. If more than 38 characters are entered, the prefix line is cleared and entry must be repeated. Ugh! When a prefix name is finally completed, **lines 209-218** set the prefix. An error restarts the selection process.

The good part is that $300 bytes of space were used where only $200 bytes would have been adequate, thus allowing us an extra $100 bytes.

Next, an application name is requested (**lines 222-228**). Unbelievably, an entirely new input routine is used (**lines 229-272, 334-343**). There is no point in describing the code, because it's nearly identical to the prior input sequence.

When the application name is tucked into the secondary text buffer (TXBUF2), the dispatcher portion of the QUIT code takes over (**lines 276-330**). If the application is not a system (SYS) file, it's rejected and new input is solicited. On confirming a SYS file, all files are closed, the target file is opened, and the file reference number is placed in other pertinent parameter lists. Obtained by the GET_EOF call, the file length is stuffed into the READ parameter list. A check for a file larger than 64K (**lines 310-313**) is unnecessary, because the maximum number of bytes that can be read is limited to the number of bytes between the start of the READ buffer ($2000) and the nearest protected page of memory ($BF00). Thus, a file size of $9F00 bytes is maximum. If no error is reported, the target SYS file is read into memory and closed. **Line 330** turns over control to the loaded application program.

Errors are handled by **lines 356-377**, where all execution miscues are divided into two categories: FILE/PATH NOT FOUND and I/O ERROR.

In reflecting upon the QUIT code, we discover that, as with all creative efforts, pluses and minuses emerge. The bad part is that $300 bytes of space were used where only $200 bytes would have been adequate if the code were tighter. The good part is that $300 bytes of space were used where only $200 bytes would have been adequate, thus allowing us an extra $100 bytes to write our own selector/dispatcher code. It seems to me that the author inadvertently did us a favor, as you shall see in the next section.

A BETTER WAY

Two logical formats for a program selector come to mind: 1) A selector that polls all on-line devices and allows you to home in on a SYS program in one of the directories or subdirectories, and 2) a selector that contains a preset table of your most frequently used applications. I shall present you with the latter type. If the first variety appeals to you, get the July 1986 issue of *Apple Assembly Line* and look at the excellent QUIT code written by Bob Sander-Cederlof. Revisions to the S-C selector are found in the August, October and December 1986 editions.

SELECTOR INSTALLATION

Listing 2 contains the code for our selector. To make the QUIT code functional, follow these instructions:

1. Format a disk using FILER or a similar utility. Name the disk /SELECTOR. Copy PRODOS and BASIC.SYSTEM onto the disk, and note which version of ProDOS is being employed. Alternatively, you may rename a current disk /SELECTOR, provided that the disk contains the two files named above. Be certain that PRODOS is unlocked.
2. From Applesoft, type PREFIX/SELECTOR.
3. Enter the code in **Listing 2** manually (by entering the Monitor with CALL −151), or assemble it. Using an assembler enables you to customize the program more readily. For help with entering *Nibble* listings, see the Typing Tips section. Save the program by typing:
   BSAVE SELECTOR.CODE,A$1000,L$300
4. Type:
   BLOAD PRODOS, TSYS,A$2000
5. If you are using version 1.2 or 1.3 of ProDOS, type:
   BLOAD SELECTOR.CODE,A$5900
   For version 1.1.1 of ProDOS, use $5700 for the A-parameter.
6. Type:
   BSAVE PRODOS,TSYS,A$2000
7. Your ProDOS file now contains the new QUIT code. Reboot the /SELECTOR disk or type -PRODOS followed by a Return to install the selector.

You can automate steps 4-7 by using SELECTOR.INST (**Listing 3**). Type in the program and save it. Be sure you have it saved securely, since it is destroyed when the −PRODOS command is executed in **line 160**. Then run it with SELECTOR.CODE (**Listing 2**) on the same disk. You can replace the PRODOS file using FILER or the system utilities on a number of disks, but note that only the PRODOS file in the current directory is updated when the Control-A or Control-D function is used.

Selector Function

By typing BYE from Applesoft or the monitor, you may access the Selector. Do it. The display is in 80 columns. On my first try with this program, the top line showed which keys could be used to select arrows, dispatch (Return), add (Control-A) and delete (Control-D) applications. Because of limited space, I decided to reclaim these bytes for the application table, so you will have to remember the commands. Only one name is visible: /SELECTOR/BASIC.SYSTEM. The highlighted file is selected. Press Return and see how rapidly you are dispatched to BASIC.SYSTEM.

Now, use the BYE command to reenter our selector. Type Control-D and note that a single listed file cannot be deleted. Type Control-A, observe the PATH request near the bottom of the screen, and enter the full pathname of a frequently used application. Only SYS files can be executed. For example, I would type /SELECTOR/MERLIN.SYSTEM to execute the file. When the entry is completed and the disk is whirring, PRODOS is being loaded, the table of SYS files within PRODOS is being updated, and PRODOS is being written back to disk. By sensing which version of PRODOS is active, the selector correctly handles all versions of /SELECTOR/ PRODOS. When this process is completed, the application table contains two files, the original one and the file you just added. By striking the arrow keys, you may move from file to file.

If PRODOS is locked when the SYS table is altered, a bell indicates that PRODOS has not been updated. The application table within the LC has been updated, however, so the revised table continues to be available as long as the computer is not rebooted or PRODOS is not re-executed.

During entry of the application pathname, any printable character is accepted. At the beginning of input, the Escape key returns you to the menu; in the middle of the line, Escape restarts input. The Delete and Left-Arrow keys produce a destructive backspace. Input is rejected if the pathname exceeds 64 characters or if the SYS table contains 206 characters. At this stage, no check is made for a full pathname or a valid file name. A maximum of 21 application names is permitted; thereafter, the Control-A function to add names is disabled.

When a valid on-line filename is selected and executed, the application is started up. If the above conditions do not prevail, a bell sounds and the application menu reappears. If you have placed an invalid filename within the table, reenter the selector, select the faulty entry, and press Control-D to delete it.

The more disk space you've got, the more valuable a program selector becomes. My IIGS system contains two UniDisk 3.5-inch drives and one Disk II drive. My Apple IIe is configured with one UniDisk 3.5, one Disk II, and one hard disk. With six and eight application names, respectively, in my selector tables, I now move between applications with a grace and speed which I could no longer do without.

With six and eight application names, respectively, in my selector tables, I now move between applications with a grace and speed which I could no longer do without.

SELECTOR CODE

The difficulty of fitting a sophisticated program into a $300 byte space is evinced by my writing five versions of the utility. Naturally, I have provided you with my favorite version. In many instances, I have sacrificed bells, whistles and text for a larger application table.

The selector code in **Listing 2** begins with a CLD instruction (**line 44**). Apple Technical Note 14 insists that this convention is necessary to distinguish customized selector/dispatcher code from the native variety. Although I know no program or portion of ProDOS which checks for this initial byte, Apple has a way of surprising us, and I recommend the one byte sacrifice for an assurance of safety.

After enabling motherboard ROM and resetting the system bit map, file closure is ensured and 80-column mode is initialized (**lines 45-59**). The first menu (SYS table) line is selected by placing a zero in SELECLIN (**lines 60-61**).

The table of SYS files extends from $1260-$12FE, and the byte at $12FF contains the number of entries, minus one, in the table (**lines 351-357**). Individual entries are in negative ASCII format (i.e., high bit set) with the final character in positive ASCII (i.e., high bit clear). Similar schemes are found in Applesoft and DOS 3.3 command tables. A zero marks the end of the table entries, which may hold a maximum of 190 characters ($1300-$1240-2 = $BE).

The menu is placed on the screen by **lines 65-87**. As stated above, a low ASCII character flags the end of one entry, and a zero marks the end of all entries. When printing is completed, IXSYSSEL indexes the beginning of the selected line, and IXSYSEND points to the end of the table.

Menu commands are obtained in **lines 91-109**. If an Up-Arrow or Left-Arrow is chosen, the selected line is decremented unless the first line is already highlighted, in which case the last line (MAXLIN) is selected (**lines 113-117**). If a Down-Arrow or Right-Arrow is struck, the above process is reversed (**lines 121-127**).

Lines 131-186 allow a new entry if no more than 20 table pathnames exist. The Add routine rejects pathnames longer than 64 characters and guards against the table length exceeding 190 characters. Verboten characters are not filtered out, but an invalid file cannot be executed.

A file name may be deleted (**lines 190-205**) if more than one table entry exists. After reducing the number of entries by one, the index to the selected file name is placed in the X-Register, and the index to the subsequent file name is stored in the Y-Register. Deletion is completed by overwriting the selected entry with the remaining entries and ensuring that a zero terminates the altered table.

When the SYS table is revised by adding or deleting a file name, the new table must be written to the /SELECTOR/PRODOS file and to the SYS table which lives in the LC. **Lines 209-212** point at the ProDOS filename, and **line 213** calls OPENREAD (**lines 281-307**) to open and read into memory the file whose name string is located by PTR. In the course of placing the file name into TXBUF2, the length of the prefix portion of the complete pathname is saved in PFXLEN for use later. The file is opened by a standard MLI call, and the returned reference number is put into other pertinent parameter lists. As calculated earlier, the maximum number of bytes that can be transferred to memory is $9F00. This number is stuffed into the R/W parameter list, and the READ call is made. File reading ends when the EOF marker is reached. No error is recorded unless zero bytes have been transferred to memory, a virtual impossibility within the context of this program. Control returns to **line 214** where the error status of the OPEN and READ calls is saved.

After write-enabling the second LC bank, the SYS table within that bank and /SELECTOR/PRODOS are updated (**lines 215-227**). In updating PRODOS, the System Global Page determines which version of PRODOS has been booted and therefore loaded. KVERSION values of one, two and three designate versions 1.1.1, 1.2 and 1.3, respectively. The OPEN/READ status is then restored and errors are reported (**lines 228-229**). After resetting the file MARK to zero (**lines 230-232**), /SELECTOR/PRODOS is written back to disk by placing the actual file size into the R/W parameter list and executing the WRITE call (**lines 233-240**). A bell denotes a call error (**line 241**), and **line 242** restarts the selection process.

Execution of a selected entry is made easy by the aforementioned steps. **Lines 246-253** point to, read and close the targeted file. Non-SYS files are rejected by making the GET_FILE_INFO call and testing the parameter list (**lines 254-260**). Setting the prefix to the SYS file directory is a nice touch. PFXLEN replaces the length byte of the complete pathname in TXBUF2, and SET_PREFIX does the work (**lines 263-267**). The length of the complete system file is saved (**lines 261-262**) and restored (**lines 268-269**) to accommodate AppleWorks, which demands that its file name resides in TXBUF2. Finally, **line 270** passes control to the file within the READ buffer, and the new interpreter is off and running.

POSTSCRIPT

I'll wager a double-scoop mocha chip ice cream cone that the *Nibble* selector will forever change your pattern of entering and exiting application programs. Use it well.

In the next episode of Disassembly Lines, I shall return to the IIGS. . .maybe! If a working disassembler appears by that time, we may even tackle some 65816 code. See you then.

LISTING 1: QUIT CODE

NOTE: **This code already exists in PRODOS. There is no need to type it in.**

```
             *****************************************
             *                                       *
             *              QUIT CODE                *
             *        PRODOS 8, Version 1.3          *
             *    Interpreted by Sandy Mossberg      *
             *                                       *  Merlin-Pro
             *         Copyright (C) 1987            *
             *         by MicroSPARC, Inc.           *
             *         Concord, MA  01742            *
             *                                       *
             *****************************************

             CH       =      $24          ;cursor column
             CV       =      $25          ;cursor row
             TEMP     =      $DE          ;temporary storage
             TXBUF2   =      $280         ;secondary text buffer
             RESET    =      $3F2         ;RESET vector
             SCNROW5  =      $600         ;left margin of 5th screen row
             OPENBUF  =      $1800        ;io.buffer for OPEN call
             READBUF  =      $2000        ;data buffer for READ call
             MLI      =      $BF00        ;MLI call entry
             BITMAP   =      $BF58        ;system bit map
             INIT     =      $FB2F        ;initialize text screen
             HOME     =      $FC58        ;clear screen and home cursor
             CLREOL   =      $FC9C        ;clear to end of line
             RDKEY    =      $FD0C        ;get input character
             CROUT    =      $FD8E        ;output CR
             COUT     =      $FDED        ;output character
             SETNORM  =      $FE84        ;set normal text mode
             SETKBD   =      $FE89        ;set input from keyboard
             SETVID   =      $FE93        ;set output to screen
             BELL     =      $FF3A        ;output bell character

             *================================
             * INSTALL QUIT CODE IN LANGUAGE CARD:
             *================================
                      ORG    $20B7        ;loaded PRODOS file in low RAM
                                          ;V 1.2   = $20B7
                                          ;V 1.1.1 = $2099
20B7: A9 D5           LDA    #MLIQUIT     ;set MLI quit vector in
20B9: 8D 04 BF        STA    QUIT+1       ; System Global Page
20BC: A9 FC           LDA    #>MLIQUIT
20BE: 8D 05 BF        STA    QUIT+2
20C1: AD 83 C0        LDA    $C083        ;R/W enable 2nd 4K bank
20C4: AD 83 C0        LDA    $C083        ; of language card (LC)
20C7: AE 74 22        LDX    $2274        ;point to QUIT
20CA: AC 75 22        LDY    $2275        ; code table
20CD: 20 EC 28        JSR    $28EC        ;relocate QUIT code
20D0: A9 EE           LDA    #$EE         ;place ID byte in
20D2: 8D 00 D0        STA    $D000        ; 2nd LC bank
20D5: 20 18 25        JSR    $2518        ;set 1st LC bank
20D8: 90 03           BCC    CONTINUE     ;no error so continue
20DA: 4C 27 22        JMP    $2227        ;off to error handler
             CONTINUE                     ;relocator continues here
             *================================
             * SYSTEM GLOBAL PAGE VECTOR:
             *================================
                      ORG    $BF03        ;low RAM (all versions)

BF03: 4C D5 FC QUIT   JMP    MLIQUIT      ;execute quit code
             *================================
             * RELOCATE and EXECUTE INSTALLED QUIT CODE:
             *================================
                      ORG    $FCD5        ;1st 4K LC bank
                                          ;V 1.2   = $FCA9
                                          ;V 1.1.1 = $FCE5
FCD5: AD 83 C0 MLIQUIT LDA   $C083        ;R/W enable 2nd
FCD8: AD 83 C0        LDA    $C083        ; 4K LC bank
FCDB: A0 03           LDY    #$03         ;index 4 bytes
FCDD: B9 00 00 :1     LDA    $00,Y        ;save contents
FCE0: 48              PHA                 ; of $00-$03
FCE1: 88              DEY                 ; on stack
FCE2: 10 F9           BPL    :1
FCE4: A9 10           LDA    #>SELECTOR   ;set pointers for
FCE6: 85 03           STA    $03          ; moving QUIT code:
FCE8: A9 D1           LDA    #$D1         ;   FROM ($00 pointer)
FCEA: 85 01           STA    $01          ;     $D100-$D3FF in 2nd LC bank
FCEC: A9 00           LDA    #$00         ;   TO ($02 pointer)
FCEE: 85 00           STA    $00          ;     $1000-$12FF in low RAM
FCF0: 85 02           STA    $02
FCF2: A8              TAY                 ;zero Y
FCF3: A2 03           LDX    #$03         ;3 pages to transfer
FCF5: 88       :2     DEY
FCF6: B1 00           LDA    ($00),Y      ;transfer QUIT code
FCF8: 91 02           STA    ($02),Y
FCFA: 98              TYA
FCFB: D0 F8           BNE    :2
FCFD: E6 01           INC    $01          ;set next page of
FCFF: E6 03           INC    $03          ; code to move
FD01: CA              DEX
FD02: D0 F1           BNE    :2           ;3 pages not yet moved
FD04: 68       :3     PLA                 ;restore original
FD05: 95 00           STA    $00,X        ; contents of
FD07: E8              INX                 ; $00-$03 from stack
FD08: E0 04           CPX    #$04
FD0A: 90 F8           BCC    :3
FD0C: EA              NOP
FD0D: AD 8B C0        LDA    $C08B        ;R/W enable 1st
FD10: AD 8B C0        LDA    $C08B        ; 4K LC bank
FD13: A9 00           LDA    #SELECTOR    ;point RESET vector
FD15: 8D F2 03        STA    RESET        ; to beginning of
FD18: A9 10           LDA    #>SELECTOR   ; functioning
FD1A: 8D F3 03        STA    RESET+1      ; selector (QUIT) code
FD1D: 49 A5           EOR    #$A5         ;make power-up byte right to
FD1F: 8D F4 03        STA    RESET+2      ; prevent rebooting system
FD22: 4C 00 10        JMP    SELECTOR     ;<<<EXECUTE QUIT CODE>>>
             *================================
             * QUIT CODE (SELECTOR/DISPATCHER):
             *================================
                      ORG    $1000        ;low RAM
             *--------------------------------
             * Initialize Machine:
             *--------------------------------
1000: AD 82 C0 SELECTOR LDA  $C082        ;enable monitor (2nd bank) ROM
1003: 8D 0C C0        STA    $C00C        ;disable 80-column firmware
1006: 8D 0E C0        STA    $C00E        ;disable alternate char set
1009: 8D 00 C0        STA    $C000        ;disable 80-column store
100C: 20 84 FE        JSR    SETNORM
100F: 20 2F FB        JSR    INIT
1012: 20 93 FE        JSR    SETVID
1015: 20 89 FE        JSR    SETKBD
             *--------------------------------
             * Initialize System Bit Map:
             *--------------------------------
1018: A2 17           LDX    #$17         ;24 bit map bytes
101A: A9 01           LDA    #1
101C: 9D 58 BF        STA    BITMAP,X     ;reserve $BF00-$BFFF
101F: CA              DEX
1020: A9 00           LDA    #0
1022: 9D 58 BF :1     STA    BITMAP,X     ;free $800-$BEFF
1025: CA              DEX
1026: 10 FA           BPL    :1
1028: A9 CF           LDA    #$CF         ;reserve $000-$1FF, $400-$7FF
102A: 8D 58 BF        STA    BITMAP       ; and free $200-$3FF
             *--------------------------------
             * Show Current Prefix:
             *--------------------------------
102D: 20 58 FC SHOWPFX JSR   HOME
1030: 20 8E FD        JSR    CROUT
1033: A2 00           LDX    #TXPFX-ZTXT
1035: 20 D6 11        JSR    PRINT        ;print prefix request
1038: A9 03           LDA    #3           ;skip
103A: 85 25           STA    CV           ; skip
103C: 20 8E FD        JSR    CROUT        ; skip to 5th row
103F: 20 00 BF        JSR    MLI
1042: C7              HEX    C7           ;GET_PREFIX
1043: C8 12           DA     PPFX
1045: AE 80 02        LDX    TXBUF2       ;get #chars in prefix
1048: A9 00           LDA    #0           ;zero marks
104A: 9D 81 02        STA    TXBUF2+1,X   ; end of prefix
104D: AE 80 02        LDX    TXBUF2       ;again get #chars in prefix!
1050: F0 0B           BEQ    :2           ;null prefix found
1052: BD 80 02 :1     LDA    TXBUF2,X     ;get prefix char
1055: 09 80           ORA    #$80         ;upshift
1057: 9D FF 05        STA    SCNROW5-1,X  ;put char directly on screen
105A: CA              DEX
105B: D0 F5           BNE    :1           ;loop back until done
105D: A2 00    :2     LDX    #0           ;zero line input index
105F: C6 25           DEC    CV           ;go to start
1061: 20 8E FD        JSR    CROUT        ; of input line
             *--------------------------------
             * Get New Prefix:
             *--------------------------------
1064: 20 0C FD GETNPFX JSR   RDKEY        ;get input char
1067: C9 8D           CMP    #$8D         ;CR?
1069: F0 52           BEQ    SETNPFX      ;yes so input completed
106B: 48              PHA                 ;preserve A
106C: 20 9C FC        JSR    CLREOL       ;clear rest of line
106F: 68              PLA                 ;restore A
1070: C9 9B           CMP    #$9B         ;Escape?
1072: F0 B9           BEQ    SHOWPFX      ;yes, so start again
1074: C9 98           CMP    #$98         ;Control-X?
1076: F0 B5    RESTART BEQ   SHOWPFX      ;yes, so start again
1078: C9 89           CMP    #$89         ;TAB?
107A: F0 17           BEQ    :4           ;yes, so reject it
107C: C9 FF           CMP    #$FF         ;Delete?
107E: F0 04           BEQ    :1           ;yes, so destructive backspace
1080: C9 88           CMP    #$88         ;Backspace (Control-H)?
1082: D0 0D           BNE    :3           ;no
1084: E0 00    :1     CPX    #0           ;if at beginning of line,
1086: F0 03           BEQ    :2           ; then disable backspace,
1088: C6 24           DEC    CH           ; else backspace,
108A: CA              DEX                 ; decrement line index,
108B: 20 9C FC :2     JSR    CLREOL       ; destroy character
108E: 4C 64 10        JMP    GETNPFX      ; and return for more input
1091: B0 06    :3     BCS    :5           ;>CTL-H so continue processing
1093: 20 3A FF :4     JSR    BELL         ;ring bell
1096: 4C 64 10        JMP    GETNPFX      ; and return for more input
1099: C9 DB    :5     CMP    #"Z"+1
109B: 90 02           BCC    :6           ;not lower case
109D: 29 DF           AND    #$DF         ;upshift lower case
109F: C9 AE    :6     CMP    #"."
10A1: 90 F0           BCC    :4           ;allow period+
10A3: C9 DB           CMP    #"Z"+1
10A5: B0 EC           BCS    :4           ;disallow Z+
10A7: C9 BA           CMP    #"9"+1
10A9: 90 04           BCC    :7           ;allow 9-
10AB: C9 C1           CMP    #"A"
10AD: 90 E4           BCC    :4           ;disallow A-
10AF: E8       :7     INX                 ;bump input line index
10B0: E0 27           CPX    #$27         ;if prefix > 38 chars
10B2: B0 C2           BCS    RESTART      ; then restart
10B4: 9D 80 02        STA    TXBUF2,X     ;store char in buffer
```

LISTING 1: QUIT CODE *(continued)*

```
10B7: 20 ED FD  204           JSR   COUT        ;print char
10BA: 4C 64 10  205           JMP   GETNPFX     ;back for more input
                206 ------------------------------
                207 * Set New Prefix:
                208 ------------------------------
10BD: E0 00     209 SETNPFX   CPX   #0          ;if no input, then
10BF: F0 12     210           BEQ   GETPATH     ; print error message
10C1: 8E 80 02  211           STX   TXBUF2      ;save buffer length byte
10C4: 20 00 BF  212           JSR   MLI
10C7: C6        213           HEX   C6          ;SET_PREFIX
10C8: C8 12     214           DA    PPFX
10CA: 90 07     215           BCC   GETPATH     ;MLI call successful
10CC: 20 3A FF  216           JSR   BELL        ;MLI call error so ring bell
10CF: A9 00     217           LDA   #0          ;set Z-flag for next instruction
10D1: F0 A3     218 RESTART1  BEQ   RESTART     ;always restart
                219 ------------------------------
                220 * Get Pathname of Application:
                221 ------------------------------
10D3: 20 58 FC  222 GETPATH   JSR   HOME
10D6: 20 8E FD  223           JSR   CROUT
10D9: A2 28     224           LDX   #TXPATH-ZTXT
10DB: 20 D6 11  225           JSR   PRINT       ;print pathname request
10DE: A9 03     226 GETPATH1  LDA   #3           ;skip
10E0: 85 25     227           STA   CV          ; skip
10E2: 20 8E FD  228           JSR   CROUT       ; skip to 5th row
10E5: A2 00     229           LDX   #0
10E7: 20 0C FD  230 GETPATH2  JSR   RDKEY       ;get input char
10EA: C9 9B     231           CMP   #$9B        ;Escape?
10EC: D0 06     232           BNE   :1          ;no
10EE: A5 24     233           LDA   CH          ;Escape either goes to
10F0: D0 E1     234           BNE   GETPATH     ; start of application line
10F2: F0 DO     235           BEQ   RESTART1    ; or to request for prefix
10F4: C9 98     236 :1        CMP   #$98        ;Control-X?
10F6: F0 DB     237 :2        BEQ   GETPATH     ;yes, so get pathname request
10F8: C9 89     238           CMP   #$89        ;TAB?
10FA: F0 0D     239           BEQ   :5          ;yes, so reject it
10FC: C9 FF     240           CMP   #$FF        ;DELETE?
10FE: F0 04     241           BEQ   :3          ;yes, so destructive backspace
1100: C9 88     242           CMP   #$88        ;Backspace (Control-H)?
1102: D0 03     243           BNE   :4          ;no
1104: 4C C0 11  244 :3        JMP   BCKSPC      ;go to destructive backspace
1107: B0 06     245 :4        BCS   :6          ;>CTL-H so continue processing
1109: 20 3A FF  246 :5        JSR   BELL        ;ring bell
110C: 4C E7 10  247           JMP   GETPATH2    ; and return for more input
110F: C9 8D     248 :6        CMP   #$8D        ;Return
1111: F0 29     249           BEQ   :9          ;yes, so input completed
1113: C9 DB     250           CMP   #'Z'+1
1115: 90 02     251           BCC   :7          ;not lower case
1117: 29 DF     252           AND   #$DF        ;upshift lower case
1119: C9 AE     253 :7        CMP   #'.'
111B: 90 EC     254           BCC   :5          ;allow period+
111D: C9 DB     255           CMP   #'Z'+1
111F: B0 E8     256           BCS   :5          ;disallow Z+
1121: C9 BA     257           CMP   #'9'+1
1123: 90 04     258           BCC   :8          ;allow 9-
1125: C9 C1     259           CMP   #'A'
1127: 90 E0     260           BCC   :5          ;disallow A-
1129: 48        261 :8        PHA               ;preserve A
112A: 20 9C FC  262           JSR   CLREOL      ;clear rest of line
112D: 68        263           PLA               ;restore A
112E: 20 ED FD  264           JSR   COUT        ;print char
1131: E8        265           INX               ;bump input line index
1132: E0 27     266           CPX   #$27        ;if pathname > 38 chars,
1134: B0 C0     267           BCS   :2          ; then clear line
1136: 9D 80 02  268           STA   TXBUF2,X    ;store char in buffer
1139: 4C E7 10  269           JMP   GETPATH2
113C: A9 A0     270 :9        LDA   #' '
113E: 20 ED FD  271           JSR   COUT        ;print space
1141: 8E 80 02  272           STX   TXBUF2      ;save buffer length byte
                273 ------------------------------
                274 * Execute Application File:
                275 ------------------------------
1144: 20 00 BF  276           JSR   MLI
1147: C4        277           HEX   C4          ;GET_FILE_INFO
1148: A1 12     278           DA    PFINFO
114A: 90 03     279           BCC   EXECFILE    ;MLI call successful
114C: 4C E2 11  280           JMP   HANDLERR    ;MLI call error
114F: AD A5 12  281 EXECFILE  LDA   FIFILID
1152: C9 FF     282           CMP   #$FF
1154: F0 05     283           BEQ   :1          ;SYS file found so proceed
1156: A9 01     284           LDA   #1          ;SYS file not found
1158: 4C E2 11  285           JMP   HANDLERR    ; so print error message
115B: A9 00     286 :1        LDA   #0          ;set parmlist to
115D: 8D BA 12  287           STA   CLREFNUM    ; close all files
1160: 20 00 BF  288           JSR   MLI
1163: CC        289           HEX   CC          ;CLOSE
1164: B9 12     290           DA    PCLOSE
1166: 90 03     291           BCC   :2          ;MLI call successful
1168: 4C E2 11  292           JMP   HANDLERR    ;MLI call error
116B: AD A4 12  293 :2        LDA   FIACESS     ;isolate read-protection
116E: 29 01     294           AND   #1          ; access attribute bit
1170: D0 05     295           BNE   :3          ;file read-enabled
1172: A9 27     296           LDA   #$27        ;report catch-all
1174: 4C E2 11  297           JMP   HANDLERR    ; I/O ERROR
1177: 20 00 BF  298 :3        JSR   MLI
117A: C8        299           HEX   C8          ;OPEN
117B: B3 12     300           DA    POPEN
117D: 90 03     301           BCC   :4          ;MLI call successful
117F: 4C E2 11  302           JMP   HANDLERR    ;MLI call error
1182: AD B8 12  303 :4        LDA   OPREFNUM    ;stuff file reference number
1185: 8D BC 12  304           STA   RDREFNUM    ; in READ and
1188: 8D C4 12  305           STA   EOREFNUM    ; GET_EOF parmlists
118B: 20 00 BF  306           JSR   MLI
118E: D1        307           HEX   D1          ;GET_EOF
118F: C3 12     308           DA    PEOF
1191: B0 4F     309           BCS   HANDLERR    ;MLI call error
1193: AD C7 12  310           LDA   EOFVAL+2    ;if 24-bit file size
1196: F0 04     311           BEQ   :5          ; indicates that size
1198: A9 27     312           LDA   #$27        ; of file exceeds 64K,
119A: D0 46     313           BNE   HANDLERR    ; then report I/O ERROR
119C: AD C5 12  314 :5        LDA   EOFVAL      ;place 16-bit
119F: 8D BF 12  315           STA   RDCOUNT     ; file size
11A2: AD C6 12  316           LDA   EOFVAL+1    ; into READ
11A5: 8D C0 12  317           STA   RDCOUNT+1   ; parmlist
11A8: 20 00 BF  318           JSR   MLI
11AB: CA        319           HEX   CA          ;READ
11AC: BB 12     320           DA    PREAD
11AE: 08        321           PHP               ;save READ error status
11AF: 20 00 BF  322           JSR   MLI
11B2: CC        323           HEX   CC          ;CLOSE
11B3: B9 12     324           DA    PCLOSE
11B5: 90 04     325           BCC   :7          ;should be BCC :62
11B7: 28        326 :6        PLP
11B8: D0 28     327 :61       BNE   HANDLERR    ;JMP HANDLERR would be safer
11BA: 28        328 :62       PLP
11BB: B0 FA     329 :7        BCS   :6          ;should be BCS :61
11BD: 4C 00 20  330           JMP   READBUF     ;<<<EXECUTE LOADED SYS FILE>>>
                331 ------------------------------
                332 * Perform Destructive Backspace:
                333 ------------------------------
11C0: A5 24     334 BCKSPC    LDA   CH          ;if cursor at start of line,
11C2: F0 0F     335           BEQ   :1          ; then get another char,
11C4: CA        336           DEX               ; decrement line index,
11C5: A9 A0     337           LDA   #' '        ; print a
11C7: 20 ED FD  338           JSR   COUT        ; space,
11CA: C6 24     339           DEC   CH          ; go back 2 columns to
11CC: C6 24     340           DEC   CH          ; compensate for COUT advance,
11CE: 20 ED FD  341           JSR   COUT        ; destroy character with space,
11D1: C6 24     342           DEC   CH          ; and again compensate for COUT
11D3: 4C E7 10  343 :1        JMP   GETPATH2    ;get another pathname char
                344 ------------------------------
                345 * Print Message:
                346 ------------------------------
11D6: BD 11 12  347 PRINT     LDA   ZTXT,X
11D9: F0 06     348           BEQ   :1
11DB: 20 ED FD  349           JSR   COUT
11DE: E8        350           INX
11DF: D0 F5     351           BNE   PRINT
11E1: 60        352 :1        RTS
                353 ------------------------------
                354 * Error Handler:
                355 ------------------------------
11E2: 85 DE     356 HANDLERR  STA   TEMP        ;save MLI error code
11E4: A9 0C     357           LDA   #$0C        ;skip
11E6: 85 25     358           STA   CV          ; skip
11E8: 20 8E FD  359           JSR   CROUT       ; skip to 13th row
11EB: A5 DE     360           LDA   TEMP        ;restore MLI error code
11ED: C9 01     361           CMP   #1          ;QUIT code for no SYS file found
11EF: D0 04     362           BNE   :1
11F1: A2 48     363           LDX   #TXNOTSYS-ZTXT
11F3: D0 16     364           BNE   :3          ;always
11F5: C9 40     365 :1        CMP   #$40        ;INVALID PATHNAME SYNTAX
11F7: F0 10     366           BEQ   :2
11F9: C9 44     367           CMP   #$44        ;PATHNAME NOT FOUND
11FB: F0 0C     368           BEQ   :2
11FD: C9 45     369           CMP   #$45        ;VOLUME DIRECTORY NOT FOUND
11FF: F0 08     370           BEQ   :2
1201: C9 46     371           CMP   #$46        ;FILE NOT FOUND
1203: F0 04     372           BEQ   :2
1205: A2 62     373           LDX   #TXIOERR-ZTXT
1207: D0 02     374           BNE   :3          ;always (HEX 2C saves 1 byte)
1209: A2 79     375 :2        LDX   #TXFNF-ZTXT
120B: 20 D6 11  376 :3        JSR   PRINT       ;print error message
120E: 4C DE 10  377           JMP   GETPATH1
                378 ------------------------------
                379 * Messages:
                380 ------------------------------
                381 ZTXT      =     *
1211: C5 CE D4  382 TXPFX     ASC   !ENTER PREFIX (PRESS "RETURN"!
1214: C5 D2 A0 D0 D2 C5 C6 C9
121C: D8 A0 A8 D0 D2 C5 D3 D3
1224: A0 A2 D2 C5 D4 D5 D2 CE
122C: A2
122D: A0 D4 CF  383           ASC   " TO ACCEPT)"00
1230: A0 C1 C3 C3 C5 D0 D4 A9
1238: 00
1239: C5 CE D4  384 TXPATH    ASC   "ENTER PATHNAME OF NEXT APPLICATION"00
123C: C5 D2 A0 D0 C1 D4 C8 CE
1244: C1 CD C5 A0 CF C6 A0 CE
124C: C5 D8 D4 A0 C1 D0 D0 CC
1254: C9 C3 C1 D4 C9 CF CE 00
125C: 87        385 TXNOTSYS HEX   87
125D: CE CF D4  386           ASC   !NOT A TYPE "SYS" FILE!00
1260: A0 C1 A0 D4 D9 D0 C5 A0
1268: A2 D3 D9 D3 A2 A0 C6 C9
1270: CC C5 00
1273: 87        387 TXIOERR   HEX   87
1274: C9 AF CF  388           ASC   "I/O ERROR            "00
1277: A0 C5 D2 D2 CF D2 A0 A0
127F: A0 A0 A0 A0 A0 A0 A0 A0
1287: A0 A0 00
128A: 87        389 TXFNF     HEX   87
128B: C6 C9 CC  390           ASC   "FILE/PATH NOT FOUND  "00
128E: C5 AF D0 C1 D4 C8 A0 CE
1296: CF D4 A0 C6 CF D5 CE C4
129E: A0 A0 00
                391 ------------------------------
                392 * Parameter Lists:
                393 ------------------------------
12A1: 0A        394 PFINFO    HEX   0A          ;SET/GET_FILE_INFO PARMLIST
12A2: 80 02     395           DA    TXBUF2      ;pathname pointer
12A4: 00        396 FIACESS   HEX   00          ;access
12A5: 00        397 FIFILID   HEX   00          ;file.type
12A6: 00 00     398           DW    0           ;aux.type
12A8: 00        399           HEX   00          ;storage.type
12A9: 00 00     400           DW    0           ;blocks.used
12AB: 00 00     401           DW    0           ;mod.date
12AD: 00 00     402           DW    0           ;mod.time
12AF: 00 00     403           DW    0           ;create.date
12B1: 00 00     404           DW    0           ;create.time
                405 ------------------------------
12B3: 03        406 POPEN     HEX   03          ;OPEN PARMLIST
12B4: 80 02     407           DA    TXBUF2      ;pathname.pointer
12B6: 00 18     408           DA    OPENBUF     ;io.buffer
12B8: 00        409 OPREFNUM  HEX   00          ;ref.num
                410 ------------------------------
12B9: 01        411 PCLOSE    HEX   01          ;CLOSE PARMLIST
```

```
12BA: 00         CLREFNUM HEX  00          ;ref.num
                 *-------------------------------
12BB: 04         PREAD    HEX  04          ;READ PARMLIST
12BC: 00         RDREFNUM HEX  00          ;ref.num
12BD: 00 20               DA   READBUF     ;data buffer
12BF: 00 00      RDCOUNT  DW   0           ;request count
12C1: 00 00               DW   0           ;trans count
                 *-------------------------------
12C3: 02         PEOF     HEX  02          ;SET/GET_EOF PARMLIST
12C4: 00         EOREFNUM HEX  00          ;ref.num
12C5: 00 00 00   EOFVAL   DS   3
                 *-------------------------------
12C8: 01         PPFX     HEX  01          ;SET/GET_PREFIX PARMLIST
12C9: 80 02               DA   TXBUF2      ;pathname pointer
                 *-------------------------------
12CB: 00 00 00            DS   \

--End assembly, 889 bytes, Errors: 0
```

END OF LISTING 1

LISTING 2: SELECTOR.CODE

```
                 ****************************************
                 *                                      *
                 *           SELECTOR.CODE              *
                 *         by Sandy Mossberg            *
                 *                                      * Merlin-Pro
                 *         Copyright (C) 1987           *
                 *         by MicroSPARC, Inc.          *
                 *         Concord, MA  01742           *
                 *                                      *
                 ****************************************

                 CV       =    $25         ;cursor row
                 TXBUF2   =    $280        ;secondary text buffer
                 MLI      =    $BF00       ;MLI call entry
                 BITMAP   =    $BF58       ;system bit map
                 KVERSION =    $BFFF       ;PRODOS version
                 KEY      =    $C000       ;keypress storage
                 STROBE   =    $C010       ;keyboard strobe
                 C3ROM    =    $C300       ;entry to 80-column mode
                 CLREOP   =    $FC42       ;clear to end of text screen
                 RDKEY    =    $FD0C       ;get input character
                 CROUT    =    $FD8E       ;output carriage return
                 COUT     =    $FDED       ;output character
                 SETINV   =    $FE80       ;set inverse text mode
                 SETNORM  =    $FE84       ;set normal text mode
                 BELL     =    $FF3A       ;output bell character
                 *-------------------------------
                 PTR      =    $F9         ;pointer
                 CURLIN   =    $FB         ;SYS table line-1 being printed
                 SELECLIN =    $FC         ;selected line-1 in SYS table
                 IXSYSEND =    $FD         ;index to end of SYS table
                 IXSYSSEL =    $FE         ;index to selected line in table
                 PFXLEN   =    $FF         ;length of selected prefix
                 OPENBUF  =    $1C00       ;io buffer for OPEN call
                 RWBUF    =    $2000       ;data buffer for READ/WRITE call
                 PRO1TBL  =    $5940       ;SYS table in PRODOS 1.1
                 PRO23TBL =    $5B40       ;SYS table in PRODOS 1.2/1.3
                 LCTBL    =    $D340       ;SYS table in 2nd LC bank

                          ORG  $1000
                 *-------------------------------
                 * Initialize System:
                 *-------------------------------
1000: D8         SELSTART CLD              ;be safe not sorry
1001: AD 82 C0            LDA  $C082       ;enable monitor ROM
1004: A9 00               LDA  #0
1006: A2 16               LDX  #$16
1008: 9D 58 BF   :1       STA  BITMAP,X    ;free pages 8-$BE
100B: CA                  DEX
100C: D0 FA               BNE  :1
100E: E8                  INX
100F: 8E 6F BF            STX  BITMAP+$17  ;protect page $BF
1012: A9 CF               LDA  #$CF        ;reserve pages 0-1, 4-7
1014: 8D 58 BF            STA  BITMAP      ; and free pages 2-3
                 *-------------------------------
                 * Restart Selection:
                 *-------------------------------
1017: 20 B5 11   RESTART  JSR  CLOSEALL    ;ensure all files closed
101A: 20 00 C3            JSR  C3ROM       ;enable 80-columns, clear screen
101D: A9 00               LDA  #0
101F: 85 FC               STA  SELECLIN    ;select 1st line
                 *-------------------------------
                 * Print Table of SYS Files:
                 *-------------------------------
1021: A2 00      PRTTBL   LDX  #0
1023: 86 25               STX  CV          ;cursor at 1st row
1025: 86 FB               STX  CURLIN      ;start at 1st line
1027: 20 8E FD   :1       JSR  CROUT       ;down one row
102A: A5 FB               LDA  CURLIN
102C: C5 FC               CMP  SELECLIN
102E: D0 05               BNE  :2          ;current line not selected
1030: 86 FE               STX  IXSYSSEL    ;index to start of selected line
1032: 20 80 FE            JSR  SETINV      ;make current line inverse
1035: BD 40 12   :2       LDA  SYSTBL,X
1038: F0 15               BEQ  :4          ;end of SYS table
103A: 10 06               BPL  :3          ;final line char
103C: 20 ED FD            JSR  COUT        ;print nonfinal line char
103F: E8                  INX
1040: D0 F3               BNE  :2          ;always
1042: 09 80      :3       ORA  #$80        ;convert last line char
1044: 20 ED FD            JSR  COUT        ; to negative ASCII
1047: 20 84 FE            JSR  SETNORM     ;make subsequent lines normal
104A: E6 FB               INC  CURLIN      ;bump SYS table line
104C: E8                  INX
104D: D0 D8               BNE  :1          ;always
104F: 86 FD      :4       STX  IXSYSEND    ;index to end of SYS table
1051: 20 42 FC            JSR  CLREOP
                 *-------------------------------
                 * Get Menu Command:
                 *-------------------------------
1054: AD 00 C0   GETKEY   LDA  KEY         ;get keypress
1057: 10 FB               BPL  GETKEY      ;not there
1059: 8D 10 C0            STA  STROBE      ;gotcha so reset strobe
105C: C9 8B               CMP  #$8B        ;UP ARROW
105E: F0 1E               BEQ  UP
1060: C9 8A               CMP  #$8A        ;DOWN ARROW
1062: F0 25               BEQ  DOWN
1064: C9 95               CMP  #$95        ;-->
1066: F0 21               BEQ  DOWN
1068: C9 84               CMP  #$84        ;^D
106A: D0 03               BNE  :1
106C: 4C 10 11            JMP  DELETE
106F: C9 8D      :1       CMP  #$8D        ;CR
1071: D0 03               BNE  :2
1073: 4C 7E 11            JMP  CR
1076: C9 81      :2       CMP  #$81        ;^A
1078: F0 1E               BEQ  ADD
107A: C9 88               CMP  #$88        ;<--
107C: D0 D6               BNE  GETKEY
                 *-------------------------------
                 * UP Handler:
                 *-------------------------------
107E: C6 FC      UP       DEC  SELECLIN    ;upsy-daisy if not
1080: 10 9F               BPL  PRTTBL      ; above 1st line
1082: AD FF 12            LDA  MAXLIN      ;would be above 1st line
1085: 85 FC               STA  SELECLIN    ; so select last line
1087: 10 98      TOPRTTBL BPL  PRTTBL      ;always redisplay SYS table
                 *-------------------------------
                 * DOWN Handler:
                 *-------------------------------
1089: A5 FC      DOWN     LDA  SELECLIN
108B: CD FF 12            CMP  MAXLIN
108E: 90 04               BCC  :1          ;not on last line
1090: A9 FF               LDA  #-1         ;on last line so
1092: 85 FC               STA  SELECLIN    ; select 1st line
1094: E6 FC      :1       INC  SELECLIN    ;down-town
1096: 10 89               BPL  PRTTBL      ;always redisplay SYS table
                 *-------------------------------
                 * ADD Handler:
                 *-------------------------------
1098: AD FF 12   ADD      LDA  MAXLIN
109B: C9 14               CMP  #20         ;allow no more than 21
109D: B0 E8               BCS  TOPRTTBL    ; files in SYS table
109F: A9 15               LDA  #21
10A1: 85 25               STA  CV          ;cursor at 23rd row after CR
10A3: A2 06               LDX  #6
10A5: BD 31 12   :1       LDA  TXTPATH,X
10A8: 20 ED FD            JSR  COUT
10AB: CA                  DEX
10AC: 10 F7               BPL  :1
10AE: 20 42 FC            JSR  CLREOP
10B1: A6 FD               LDX  IXSYSEND    ;index to end of SYS table
10B3: 20 0C FD   ADDKEY   JSR  RDKEY       ;get line char
10B6: C9 FF               CMP  #$FF        ;Delete
10B8: F0 28               BEQ  ADDBS
10BA: C9 88               CMP  #$88        ;<--
10BC: F0 24               BEQ  ADDBS
10BE: C9 9B               CMP  #$9B        ;Escape
10C0: F0 2F               BEQ  ADDESC
10C2: C9 8D               CMP  #$8D        ;Return
10C4: F0 38               BEQ  ADDCR
10C6: C9 A0               CMP  #" "        ;allow no other
10C8: 90 E9               BCC  ADDKEY      ; Control char
10CA: C9 E0               CMP  #$E0
10CC: 90 02               BCC  :1          ;not upper case
10CE: 29 DF               AND  #$DF        ;lower case so upshift
10D0: 9D 40 12   :1       STA  SYSTBL,X    ;store char in SYS table
10D3: 20 ED FD            JSR  COUT        ;print line char
10D6: E8                  INX
10D7: E0 BE               CPX  #MAXLIN-SYSTBL-1 ;wipe out char if
10D9: B0 07               BCS  ADDBS       ; TABLE exceeds 206 chars
10DB: 8A                  TXA              ;note carry clear for subtract
10DC: E5 FD               SBC  IXSYSEND    ;wipe out char if pathname
10DE: C9 40               CMP  #64         ; exceeds 64 chars (4 levels)
10E0: 90 D1               BCC  ADDKEY      ;length OK so get another char
10E2: E4 FD      ADDBS    CPX  IXSYSEND
10E4: F0 CD               BEQ  ADDKEY      ;disable backspace in 1st column
10E6: A9 88               LDA  #$88        ;printing Control-H through
10E8: 20 ED FD            JSR  COUT        ; COUT does the backspace
10EB: 20 42 FC            JSR  CLREOP      ;kill char under cursor
10EE: CA                  DEX
10EF: D0 C2               BNE  ADDKEY      ;always get another char
10F1: E4 FD      ADDESC   CPX  IXSYSEND    ;Escape in middle of line restarts
10F3: D0 A3               BNE  ADD         ; line entry (carry always set)
10F5: A9 00      ADDESC1  LDA  #0          ;Escape at start of line aborts ADD
10F7: 9D 40 12            STA  SYSTBL,X    ;hex zero marks end of SYS table
10FA: 90 33               BCC  SAVSYS      ;save table if ADD successful
10FC: B0 15               BCS  TOTORST     ;always escape from ADD mode
10FE: E4 FD      ADDCR    CPX  IXSYSEND    ;if no input then exit
1100: F0 F3               BEQ  ADDESC1     ; gracefully (carry always set)
1102: 18                  CLC              ;flag successful ADD
1103: BD 3F 12            LDA  SYSTBL-1,X  ;mark end of entry by
1106: 29 7F               AND  #$7F        ; stripping high bit
1108: 9D 3F 12            STA  SYSTBL-1,X  ; (i.e. convert to pos ASCII)
110B: EE FF 12            INC  MAXLIN      ;bump line count
110E: D0 E5               BNE  ADDESC1     ;always
                 *-------------------------------
                 * DELETE Handler:
                 *-------------------------------
1110: AD FF 12   DELETE   LDA  MAXLIN      ;prevent deleting single
1113: F0 66      TOTORST  BEQ  TORSTRT1    ; file from SYS table
1115: CE FF 12            DEC  MAXLIN      ;reduce SYS table file count
1118: A5 FE               LDA  IXSYSSEL    ;locate selected file position
111A: AA                  TAX
111B: CA                  DEX              ;INX later compensates
111C: A8                  TAY
111D: B9 40 12   :1       LDA  SYSTBL,Y    ;locate position of next file
1120: 10 03               BPL  :2          ;final char found
1122: C8                  INY
1123: D0 F8               BNE  :1          ;get another nonfinal char
1125: C8         :2       INY              ;index to next file
1126: E8                  INX              ;index to selected file
```

LISTING 2: SELECTOR.CODE *(continued)*

```
1127: B9 40 12  203           LDA   SYSTBL,Y    ;overwrite selected file with
112A: 9D 40 12  204           STA   SYSTBL,X    ; remainder of SYS file table
112D: D0 F6     205           BNE   :2          ;loop back if not end of table
                206 *-------------------------------
                207 * Save Revised SYS Table to PRODOS and Language Card:
                208 *-------------------------------
112F: A9 21     209 SAVSYS    LDA   #PRODONAM   ;point to pathname
1131: 85 F9     210           STA   PTR         ; /SELECTOR/PRODOS
1133: A9 12     211           LDA   #>PRODONAM
1135: 85 FA     212           STA   PTR+1
1137: 20 BC 11  213           JSR   OPENREAD    ;setup, OPEN and READ file
113A: 08        214           PHP               ;save OPEN/READ status
113B: AD 81 C0  215           LDA   $C081       ;write-enable 2nd LC bank
113E: AD 81 C0  216           LDA   $C081
1141: A2 C0     217           LDX   #MAXLIN-SYSTBL+1
1143: BD 3F 12  218 :1        LDA   SYSTBL-1,X  ;move SYS table to 2nd LC bank
1146: 9D 3F D3  219           STA   LCTBL-1,X   ; regardless of OP/READ status
1149: AC FF BF  220           LDY   KVERSION
114C: C0 02     221           CPY   #2
114E: B0 05     222           BCS   :2          ;PRODOS version >1.1.1
1150: 9D 3F 59  223           STA   PR01TBL-1,X ;save in PRODOS 1.1.1 file
1153: 90 03     224           BCC   :3          ;always
1155: 9D 3F 5B  225 :2        STA   PR023TBL-1,X ;save in PRODOS 1.2/1.3 file
1158: CA        226 :3        DEX
1159: D0 E8     227           BNE   :1
115B: 28        228           PLP               ;restore OPEN/READ status
115C: B0 1A     229           BCS   TORSTRT     ;OPEN/READ call error
115E: 20 00 BF  230           JSR   MLI         ;reset MARK to start of file
1161: CE        231           HEX   CE          ;SET_MARK
1162: 07 12     232           DA    MARKPARM    ; (no call error expected)
1164: AD 05 12  233           LDA   RWTRANS     ;put true length
1167: 8D 03 12  234           STA   RWCOUNT     ; of PRODOS
116A: AD 06 12  235           LDA   RWTRANS+1   ; into R/W
116D: 8D 04 12  236           STA   RWCOUNT+1   ; parmlist
1170: 20 00 BF  237           JSR   MLI         ;write file to disk
1173: CB        238           HEX   CB          ;WRITE
1174: FF 11     239           DA    RWPARM
1176: 90 03     240           BCC   TORSTRT1    ;no WRITE call error
1178: 20 3A FF  241 TORSTRT   JSR   BELL        ;gong means call is wrong
117B: 4C 17 10  242 TORSTRT1  JMP   RESTART     ;all's well if no bell
                243 *-------------------------------
                244 * CR Handler (Execute Selected File):
                245 *-------------------------------
117E: A5 FE     246 CR        LDA   IXSYSSEL    ;point to selected file
1180: 69 3F     247           ADC   #SYSTBL-1   ;compensate for carry set
1182: 85 F9     248           STA   PTR
1184: A9 12     249           LDA   #>SYSTBL
1186: 85 FA     250           STA   PTR+1
1188: 20 BC 11  251           JSR   OPENREAD    ;OPEN and READ selected file
118B: B0 E8     252           BCS   TORSTRT     ;OPEN/READ call error
118D: 20 B5 11  253           JSR   CLOSEALL    ;CLOSE selected file
1190: 20 00 BF  254           JSR   MLI         ;get info to check for SYS file
1193: C4        255           HEX   C4          ;GET_FILE_INFO
1194: 0F 12     256           DA    GFIPARM
1196: B0 E0     257           BCS   TORSTRT     ;GET_FILE_INFO call error
1198: AD 13 12  258           LDA   GFIFITYP
119B: C9 FF     259           CMP   #$FF        ;SYS file code
119D: D0 D9     260           BNE   TORSTRT     ;nonSYS file is a no-no
119F: AD 80 02  261           LDA   TXBUF2      ;save length of system
11A2: 48        262           PHA               ; file on stack
11A3: A5 FF     263           LDA   PFXLEN      ;set length of
11A5: 8D 80 02  264           STA   TXBUF2      ; file prefix
11A8: 20 00 BF  265           JSR   MLI         ;set file prefix
11AB: C6        266           HEX   C6          ;SET_PREFIX
11AC: 0C 12     267           DA    PFXPARM     ; (no call error expected)
11AE: 68        268           PLA               ;restore length of system
11AF: 8D 80 02  269           STA   TXBUF2      ; file from stack
11B2: 4C 00 20  270           JMP   RWBUF       ;<<<EXECUTE SELECTED FILE>>>
                271 *-------------------------------
                272 * Close All Files:
                273 *-------------------------------
11B5: 20 00 BF  274 CLOSEALL  JSR   MLI
11B8: CC        275           HEX   CC          ;CLOSE
11B9: F7 11     276           DA    CLPARM
11BB: 60        277           RTS
                278 *-------------------------------
                279 * Setup, Open and Read File:
                280 *-------------------------------
11BC: A0 00     281 OPENREAD  LDY   #0          ;copy file name to text buffer
11BE: 8C 03 12  282           STY   RWCOUNT
11C1: B1 F9     283 :1        LDA   (PTR),Y     ;get char
11C3: 10 0E     284           BPL   :3          ;final char
11C5: C9 AF     285           CMP   #"/"
11C7: D0 02     286           BNE   :2
11C9: 84 FF     287           STY   PFXLEN      ;save length of file prefix
11CB: 29 7F     288 :2        AND   #$7F        ;convert to positive ASCII
11CD: 99 81 02  289           STA   TXBUF2+1,Y  ;copy nonfinal char to buffer
11D0: C8        290           INY
11D1: D0 EE     291           BNE   :1          ;always
11D3: 99 81 02  292 :3        STA   TXBUF2+1,Y  ;copy final char to buffer
11D6: C8        293           INY
11D7: 8C 80 02  294           STY   TXBUF2      ;save length byte
11DA: 20 00 BF  295           JSR   MLI         ;open file
11DD: C8        296           HEX   C8          ;OPEN
11DE: F9 11     297           DA    OPPARM
11E0: B0 14     298           BCS   :4          ;OPEN call error
11E2: AD FE 11  299           LDA   OPREFNUM    ;stuff file reference#
11E5: 8D 00 12  300           STA   RWREFNUM    ; in R/W parmlist
11E8: 8D 08 12  301           STA   MKREFNUM    ; and SET_MARK parmlist
11EB: A9 9F     302           LDA   #$9F        ;maximum bytes transferable:
11ED: 8D 04 12  303           STA   RWCOUNT+1   ; $8F00-$2000=$9F00
11F0: 20 00 BF  304           JSR   MLI         ;read file from disk
11F3: CA        305           HEX   CA          ;READ
11F4: FF 11     306           DA    RWPARM
11F6: 60        307 :4        RTS               ;check error on return to caller
                308 *-------------------------------
                309 * Parameter Lists:
                310 *-------------------------------
11F7: 01 00     311 CLPARM    DFB   1,0          ;CLOSE PARMLIST (all files)
                312 *-------------------------------
11F9: 03        313 OPPARM    HEX   03          ;OPEN PARMLIST
11FA: 80 02     314           DA    TXBUF2      ;pathname pointer
11FC: 00 1C     315           DA    OPENBUF     ;io buffer
11FE: 00        316 OPREFNUM  HEX   00          ;ref num
                317 *-------------------------------
11FF: 04        318 RWPARM    HEX   04          ;READ/WRITE PARMLIST
1200: 00        319 RWREFNUM  HEX   00          ;ref num
1201: 00 20     320           DA    RWBUF       ;data buffer
1203: 00 00     321 RWCOUNT   DA    0           ;request count
1205: 00 00     322 RWTRANS   DA    0           ;trans count
                323 *-------------------------------
1207: 02        324 MARKPARM  HEX   02          ;SET_MARK PARMLIST
1208: 00        325 MKREFNUM  HEX   00          ;ref num
1209: 00 00 00  326           HEX   000000      ;MARK=0 (start of file)
                327 *-------------------------------
120C: 01        328 PFXPARM   HEX   01          ;SET_PREFIX PARMLIST
120D: 80 02     329           DA    TXBUF2      ;pathname pointer
                330 *-------------------------------
120F: 0A        331 GFIPARM   HEX   0A          ;GET_FILE_INFO PARMLIST
1210: 80 02     332           DA    TXBUF2      ;pathname pointer
1212: 00        333           HEX   00          ;access code
1213: 00        334 GFIFITYP  HEX   00          ;file type
1214: 00 00 00  335           DS    13          ;other stuff
1217: 00 00 00 00 00 00 00 00
121F: 00 00
                336 *-------------------------------
                337 * Pathname of Target PRODOS File:
                338 *-------------------------------
1221: AF D3 C5  339 PRODONAM  DCI   "/SELECTOR/PRODOS"
1224: CC C5 C3 D4 CF D2 AF D0
122C: D2 CF C4 CF 53
                340 *-------------------------------
                341 * Text:
                342 *-------------------------------
1231: A0 BA C8  343 TXTPATH   REV   "PATH: "
1234: D4 C1 D0
1237: 8D        344           HEX   8D
                345 TXTEND
                346           ERR   --1/SYSTBL  ;trap extension beyond SYSTBL
1238: 00 00 00  347           DS    SELSTART+$240-TXTEND
123B: 00 00 00 00 00
                348 *-------------------------------
                349 * Table of System Files:
                350 *-------------------------------
1240: AF D3 C5  351 SYSTBL    DCI   "/SELECTOR/BASIC.SYSTEM"
1243: CC C5 C3 D4 CF D2 AF C2
1248: C1 D3 C9 C3 AE D3 D9 D3
1253: D4 C5 4D
1256: 00        352           HEX   00          ;end-of-table marker
                353 STBLEND
1257: 00 00 00  355           DS    SELSTART+$300-STBLEND-1
                356
12FF: 00        357 MAXLIN    HEX   00          ;highest line-1 in SYS table

--End assembly, 768 bytes, Errors: 0
```

END OF LISTING 2

```
KEY PERFECT 5.0
RUN ON
SELECTOR.CODE
```

CODE-5.0	ADDR# - ADDR#	CODE-4.0
F86CD9D9	1000 - 104F	284B
D4BB60D7	1050 - 109F	2708
25528080	10A0 - 10EF	25A7
9374AB2E	10F0 - 113F	22FE
96E4D5FE	1140 - 118F	2395
13FA3866	1190 - 11DF	28DC
DC08F326	11E0 - 122F	23B2
A0C7E197	1230 - 127F	27D1
5678BE35	1280 - 12CF	00
6A4C770D	12D0 - 12FF	00
D73B7408 = PROGRAM TOTAL =		0300

LISTING 3: SELECTOR.INST

```
10  REM   **********************
20  REM   * SELECTOR.INST       *
30  REM   * BY SANDY MOSSBERG   *
40  REM   * COPYRIGHT (C) 1987 *
50  REM   * BY MICROSPARC, INC *
60  REM   * CONCORD, MA  01742 *
70  REM   **********************
80 D$ =  CHR$ (4)
90 VERS =  PEEK (49151)
100  IF  NOT VERS THEN  PRINT "PRODOS not sup
     ported": END
110  IF VERS = 1 THEN START = 5700: GOTO 130
120 START = 5900
130  PRINT D$"BLOAD PRODOS,TSYS,A$2000"
140  PRINT D$"BLOAD SELECTOR.CODE,A$"START
150  PRINT D$"BSAVE PRODOS,TSYS,A$2000"
160  PRINT D$"-PRODOS"
```

END OF LISTING 3