SECOND FEATURE

ProDOS-DOS 3.3 DOUBLEBOOT

If you own both DOS 3.3 and ProDOS, you can use this system of programs to create a disk that will boot either disk operating system.

by Ken Manly, 35 Tillinghast Place, Buffalo, NY 14216

A good program starts with a good idea. The idea for this program came from a challenge to develop a disk that would boot ProDOS on an Apple with 64K (or more) RAM and would also boot DOS 3.3 on an Apple with less memory. Such a system would allow a single disk to provide programs for a wide variety of Apples, from 48K II's to the newest //c's. ProDOS-DOS 3.3 Doubleboot is my answer to this challenge.

USING DOUBLEBOOT

When you boot the Doubleboot disk, you may be greeted in one of several ways. If Doubleboot is set up as shown in **Listings 1-3**, it will greet you with a title and copyright notice, and then it will give you the prompt "Type <P> for ProDOS; <D> for DOS 3.3." When you make your choice, it will reply "Installing ProDOS" or "Installing DOS 3.3," and it will install the appropriate operating system.

This system gives you a choice of operating systems. Suppose, however, that you nearly always used one system and only occasionally used the other. Doubleboot can be set up to load one system automatically by default. For example, let us suppose it is set up to load ProDOS. When you boot the disk, Doubleboot displays the message "Installing ProDOS. Reboot and type <ESC> or hold an Apple key to install DOS 3.3." After a short pause, it installs ProDOS. On those occasions when you want DOS 3.3, you can install it by holding down either the closed- or open-Apple key during the boot process or by pressing <ESC> before the installation message appears on the screen.

> Doubleboot can be set up to load one system automatically by default.

When you are familiar with the Doubleboot system, you won't want to read the installation message every time you boot the disk. Doubleboot can be set to eliminate this pause. In this case, Doubleboot simply displays "Installing ProDOS" or "Installing DOS 3.3" after the title and copyright notice, and it immediately installs the appropriate system. However, you can still install the alternate system by pressing <ESC> or pressing an Apple key during the boot.

SETTING DOUBLEBOOT OPTIONS

You can set up Doubleboot for any of the options just described by running the CONFIGURE program (**Listing 4**). This program changes a few bytes in the program called DBBOOT (**Listing 3**), so you must make sure that DBBOOT is unlocked before you start (the CONFIGURE program reminds you of this).

CONFIGURE explains each option and gives you a choice. First, it asks whether you want automatic selection of one system or a menu display from which to choose. If you choose a menu, CONFIGURE goes directly to the message that allows you to save your choice in DBBOOT. If you choose automatic selection, CONFIGURE asks whether DBBOOT should install ProDOS or DOS 3.3 by default.

Finally, CONFIGURE asks whether it should pause with a message explaining how to obtain the non-default system. After answering that question, you are prompted to type Y to save your choices in the DBBOOT program. You can exit from CONFIGURE without making any changes by typing any key except Y.

AN OVERVIEW

The Doubleboot system consists of four programs:

1. DOUBLER (**Listing 1**) allows you to prepare a disk so that it can hold both DOS 3.3 and ProDOS programs.
2. DOS3.3 is DOS 3.3 disguised as a ProDOS system program that is saved on disk along with DOS LOADER (**Listing 2**), which will relocate it when DOS 3.3 is selected.
3. DBBOOT (**Listing 3**) is a program that chooses whether to run ProDOS or DOS 3.3, depending on your Apple and your choice.
4. CONFIGURE (**Listing 4**) is an Applesoft program that makes it easy to set the options available in DBBOOT.

Though you will need all four listings to create your Doubleboot system disk, the boot disk will only need to contain a copy of DBBOOT and DOS3.3 along with the normal ProDOS files, PRODOS and BASIC .SYSTEM. When this disk is booted, the program DBBOOT is run instead of ProDOS. This little program checks to see if 64K of memory is available. If 64K is not available, DBBOOT runs DOS3.3, and DOS 3.3 runs a Hello program, if one is available. If 64K is available, DBBOOT allows you to choose whether to install ProDOS or DOS 3.3, as described above. If you choose ProDOS, it will run BASIC.SYSTEM as usual, or any other available .SYSTEM program.

HYBRID DISKS

To be most useful, DBBOOT should be on a disk that can be used by either DOS 3.3 or ProDOS. (I described how to make such a disk, using a ZAP program, in a letter published in the July 1984 issue of *Nibble*, Vol. 5/No. 7.) If you don't have a ZAP program, however, you can use the DOUBLER program (**Listing 1**). DOUBLER is a ProDOS-based program that will alter a newly formatted ProDOS disk to make it a hybrid DOS 3.3/ProDOS disk.

To use it, boot ProDOS and use the ProDOS FILER program to format an empty disk. Then BRUN DOUBLER. At the prompt, put the empty ProDOS-formatted disk in drive 1 and press <RETURN>. It is important that the disk have no files on it, because DOUBLER can erase or damage files when it makes its alterations. In fact, because of this, DOUBLER refuses any disk except an empty ProDOS disk. When it alters the disk, DOUBLER creates a DOS 3.3 directory on it, and fixes the DOS 3.3 and ProDOS free space maps so that they don't interfere with each other.

Finally, DOUBLER lets you choose the name of the boot program. It must be a legal ProDOS file name exactly six characters long. Normally, it will be either DBBOOT (for those of you who want to be able to boot either operating system) or ProDOS (for those of you who don't).

DOUBLER assigns tracks 20 through 34 and half of track 17 (15.5 tracks) to DOS 3.3, and it assigns the rest of the disk (19.5 tracks) to ProDOS. DOUBLER allots an extra four tracks to the ProDOS part of the disk, because that part of the disk will have to hold both ProDOS and DOS 3.3.

CREATING THE SYSTEM

To create a Doubleboot system disk, you will need the ProDOS System Utilities disk, a DOS 3.3 System Master disk, a newly formatted ProDOS disk, and another ProDOS disk.

You can type the DOUBLER (**Listing 1**), DOSLOADER (**Listing 2**), DBBOOT (**Listing 3**), and CONFIGURE (**Listing 4**) programs directly from the listings shown. Since these programs will be accessed from the ProDOS section of the Doubleboot disk, they should be entered and saved under ProDOS. (For help in typing *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.) Save DOUBLER with the command:

```
BSAVE DOUBLER,A$2000,L$46C
```

Save DOSLOADER with the command:

```
BSAVE DOSLOADER,A$2000,L$29
```

(Since these files begin at the same address, they should be entered and saved separately)

To save DBBOOT as a SYS file (which

is very necessary), you must type two commands; first:

CREATE DBBOOT,TSYS

then:

BSAVE DBBOOT,A$2000,L$270,TSYS

Notice that ProDOS insists that you CREATE an empty file of type SYS before you try to BSAVE something into it. Apparently, BSAVE can handle the saving of any file type, but it can only make directory entries for BIN type files.

At this point you should use DOUBLER to convert the newly formatted ProDOS disk into a hybrid disk. **BRUN DOUBLER**, put the newly formatted ProDOS disk into drive 1 and type Y in reponse to the prompt. When you are asked for a name for the boot program, type **DBBOOT**. You can use this disk in the next step.

The last part of the system is DOS3.3. This program was created by moving the DOS 3.3 code from its normal location (at $9D00) to $2080 and adding the short program DOSLOADER to the beginning of it. If you buy the disk version of these programs, you will have DOS3.3 on the disk. If not, you can create it in just a few steps. Using an Apple with at least 48K RAM, boot an unaltered copy of DOS 3.3 (preferably from your System Master disk). To make sure that this copy of DOS will run a Hello program named HELLO, type the command:

LOAD HELLO

Remove the System Master disk and put a DOS 3.3 disk in the drive. To avoid altering the existing slot and drive defaults, immediately type the following:

```
CALL -151
AA5F:0
2080<9D00.BFFFM
BSAVE DOS,A$2080,L$2300
```

TABLE 1: Altering DOUBLER's Space Assignment

Number of Tracks for DOS 3.3	Value of Byte $2063	Extent of FF FF 00 00 Pattern
7.5	$1C (28)	$2452–$246A
9.5	$1A (26)	$244A–$246A
11.5	$18 (24)	$2442–$246A
13.5	$16 (22)	$243A–$246A
15.5	$14 (20)	$2432–$246A
17.5	$12 (18)	$242A–$246A
19.5	$0F (15)	$241E–$246A (except $2426,$2427)
21.5	$0D (13)	$2416–$246A (except $2426,$2427)
23.5	$0B (11)	$240E–$246A (except $2426,$2427)

Those of you who are paying attention are already asking, "What is this AA5F:0 nonsense?" Well, that makes DOS think that the last DOS command was INIT. And if DOS thinks that, it will run a Hello program when it does a cold start. To review briefly, the LOAD HELLO that you just typed told DOS *which* Hello program to run (the program named HELLO). The AA5F:0 tells it that it must run *some* Hello program.

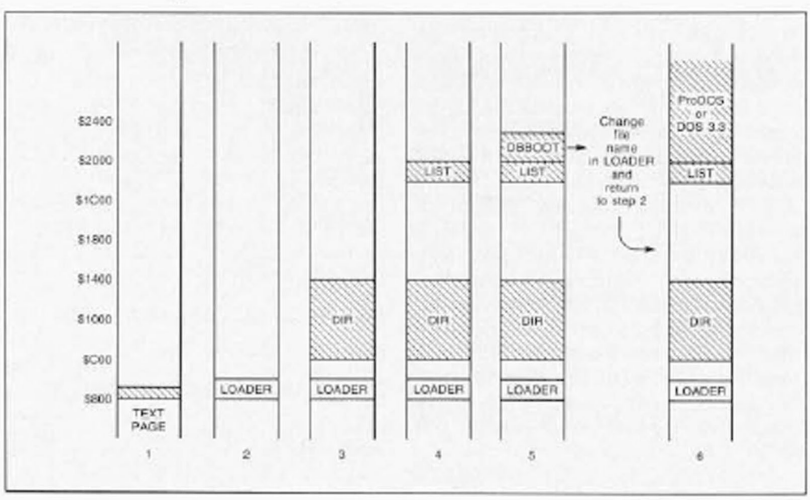
Now boot the ProDOS utilities disk, and, using the CONVERT program, transfer the file DOS to the ProDOS section of your hybrid disk. Put the utilities disk back in the drive and exit to BASIC.SYSTEM. Now combine DOS and DOSLOADER as a single ProDOS system file with the following series of commands. **BLOAD DOSLOADER** from your ProDOS disk. Then **BLOAD DOS** from your hybrid disk. Then with the hybrid disk still in the drive, type:

CREATE DOS3.3,TSYS

and

BSAVE DOS3.3,A$2000,L$2380,TSYS

FIGURE 1: Booting With Doubleboot



At this point you have all the files you need. Using the ProDOS FILER program, transfer the following files to the hybrid disk containing DOS3.3: DBBOOT, PRODOS, and BASIC.SYSTEM.

If you want to change the options described at the beginning of this article, you may do so now by running the CONFIGURE program. You may also add a STARTUP program to the ProDOS part of the disk and a HELLO program to the DOS 3.3 part of the disk, if you wish. This disk may be copied like any other with the ProDOS FILER program.

CUSTOMIZATION

You might want to change DOUBLER so that it assigns more or less space on the disk to ProDOS. This modification is reasonably simple if you assign whole tracks; I don't recommend changing the assignment by less, because each block corresponds to two sectors, and the locations of the corresponding blocks and sectors are complex. I also don't recommend making these changes if you are new to machine language programming.

To change the number of tracks that DOUBLER assigns to the two systems, you must change two parts of the DOUBLER program. One part controls the space accessible to ProDOS; the other controls the space accessible to DOS 3.3. The space accessible to ProDOS is controlled by the loop at **lines 111-116**; in particular, it is controlled by the value in byte $2063, which contains the number of the first track reserved for DOS 3.3.

The space accessible to DOS 3.3 is controlled a little differently. At the end of the DOUBLER listing (**lines 416-421**) there is a section in which the four-byte pattern FF FF 00 00 is repeated many times. Each four-byte repeat represents one track that DOS 3.3 can use. To increase the space for DOS 3.3, the pattern should be extended by changing some 00's to FF's in the earlier lines. To decrease the space for DOS 3.3, some FF's should be changed to 00's, starting with the FF's at $2432.

It is important that the changes in the two parts of the DOUBLER program be consistent. **Table 1** shows several acceptable combinations. For example, if you wanted DOUBLER to assign only 7.5 tracks to DOS 3.3 on your hybrid disks, you would change the byte at $2063 to $1C (28 decimal) and change all FF's to 00's between $2432 and $244F — leaving the FF FF 00 00 pattern extending from $2452 to $246A.

Finally, notice that bytes $2426 and $2427 should not be changed to FF even if the FF FF 00 00 pattern surrounds them. Those two bytes represent the DOS 3.3 catalog track, and they cannot be used for files.

INSIDE DOUBLER

DOUBLER uses only the simplest ProDOS calls — those that read or write a designated block. It first reads the ProDOS volume bit map (block 6) and checks to see that the disk is empty. If so, it clears the part of the bit map that corresponds to the tracks (20-34) that will be reserved for DOS 3.3. Then DOUBLER puts a DOS 3.3 VTOC and catalog on half of track 17. This is complicated somewhat by the irregular pattern by which ProDOS blocks are translated to DOS 3.3 sectors. **Table 2** shows this pattern for DOUBLER's track 17. Notice that the table gives both the DOS 3.3 sector number (also called the logical sector number) and the physical sector number. As you may know, numerically adjacent DOS sectors are physically separated on the disk to speed up disk operations.

Finally, DOUBLER asks for the name of the boot program (DBBOOT or PRODOS), and puts it into two places in the boot code in block 0. One of these is the file name that the boot code uses when it is looking through the directory to find the boot program (more on that in a moment). The other place is the error message that the boot code prints if it cannot find the program it is supposed to run. (The error message is usually something like *** UNABLE TO LOAD PRODOS ***.) DOUBLER's requirement for a six-character boot program name allows the name to fit neatly into the error message.

INSIDE DBBOOT

There are two questions you might ask about DBBOOT. How do we get the disk to run DBBOOT first, and how do we get DBBOOT to run anything else? The answer is the same for both questions, but to understand how DBBOOT functions you need a general understanding of how ProDOS boots up. **Figure 1** shows what is loaded into various locations in memory as the boot proceeds. Refer to the figure as we step through the procedure.

The boot process is started by transferring control to the ROM routine built into the disk controller card. If the card is in slot 6, this routine starts at location $C600. This routine knows how to do only one thing: it loads the contents of sector 0, track 0 into memory starting at location $800 (step 1). In the case of a ProDOS disk, this means that it loads the first half of block 0. It then runs this code, starting at $801, which then loads the rest of block 0 (step 2).

The code from block 0 is the ProDOS loader, and the next thing it does is to copy part of the ROM from the disk controller card. This gives it a subroutine that will load any block from the disk and place it anywhere in memory. Using this subroutine, the boot code from $855 to $895 loads the entire disk directory, and places it in memory starting at $C00 (step 3). The next part of the boot code ($896 to $8CB) searches through the directory for a SYS type file that is longer than one block, with the name stored at $902-$911. The name normally stored there is, of course, PRODOS.

Once the SYS type file is found, the file's block list is loaded at $1E00. The block list, analogous to the DOS 3.3 track/sector list, includes a list of the blocks in which the file is stored. Finally, the boot code $8E3-$8FE loads the file itself at $2000 (by loading each of the blocks in the block list) and jumps to location $2000.

Now you can answer the questions at the beginning of this section. We can get a disk to boot DBBOOT simply by putting that name in the boot code at the location corresponding to $902-$911. This is what the program DOUBLER does, among other things. Then, to get DBBOOT to run either ProDOS or DOS 3.3, all we have to do is to have it change the name at $902 and jump back to the boot code at $841. The boot code goes through the whole process again, starting at step 2, but this time it loads a different program. Simple, right?

INSIDE DOSLOADER

DOSLOADER is a short routine designed to make a copy of DOS 3.3 work like a ProDOS SYS program. To do this, DOSLOADER and a copy of DOS 3.3 are combined and saved as a unit. When the combined program is run, DOSLOADER first takes charge. DOSLOADER just moves DOS 3.3 back where it belongs (using the Monitor MOVE routine), and then jumps to the coldstart entry point.

There is only one important trick in DOSLOADER. It disconnects ProDOS by setting the page-zero input and output vectors to point to the Monitor input and output routines. This is important, because otherwise DOS 3.3 will find the addresses of the ProDOS input and output routines in the vectors, and it will save them and try to use them. The results would be fatal.

TABLE 2
Correspondence Between DOS 3.3 Sectors, Physical Sectors, and ProDOS Blocks for the Hybrid Disk Catalog Track

DOS Sector	Physical Sector	ProDOS Block*	Assignment
0	0	136 (1)	VTOC
1	13	143 (1)	Catalog sector 6
2	11	142 (2)	Catalog sector 5
3	9	142 (1)	Catalog sector 4
4	7	141 (2)	Catalog sector 3
5	5	141 (1)	Catalog sector 2
6	3	140 (2)	ProDOS
7	1	140 (1)	ProDOS
8	14	139 (2)	ProDOS
9	12	139 (1)	ProDOS
10	10	138 (2)	ProDOS
11	8	138 (1)	ProDOS
12	6	137 (2)	ProDOS
13	4	137 (1)	ProDOS
14	2	136 (2)	Catalog sector 7
15	15	143 (2)	Catalog sector 1

*The number in parentheses indicates the half of the ProDOS block referenced. Since each block contains 512 bytes, a one in this column indicates the first 256 bytes, while a two indicates the second 256 bytes.

CONCLUSION

ProDOS-DOS 3.3 Doubleboot was designed to be adaptable to many different kinds of Apples, but you will find it useful even if you have just one. With a Doubleboot disk in your boot drive, you can change operating systems instantly, and you can have your favorite utilities for both systems available on a single disk. Finally, if you have a single-drive system, you will find that Doubleboot is just what you need for converting files from one system to the other.

LISTING 1: DOUBLER

```
0000:                    1            LST   ON,NOA,G
0000:                    2 ;
0000:                    3 ********************************
0000:                    4 *                              *
0000:                    5 *           DOUBLER            *
0000:                    6 *        by Ken Manly          *
0000:                    7 *    Buffalo Chip Software     *
0000:                    8 *                              *
0000:                    9 *    Copyright (C) 1985        *
0000:                   10 *    by MicroSPARC, Inc.       *
0000:                   11 *    Concord, MA. 01742        *
0000:                   12 *                              *
0000:                   13 ********************************
0000:                   14 *   Apple ProDOS Assembler     *
0000:                   15 ********************************
0000:                   16
0000:          003C     17 A1L       EQU   $3C
0000:          00FC     18 PTR1      EQU   $FC
0000:          00FE     19 PTR2      EQU   $FE
0000:          0073     20 HIMEM     EQU   $73
0000:          0200     21 INPUT     EQU   $200
0000:          2000     22 SYSBEGIN  EQU   $2000
0000:          BE0C     23 PRINTERR  EQU   $BE0C
0000:          BE8B     24 BADCALL   EQU   $BE8B
0000:          BF00     25 MLI       EQU   $BF00
0000:          BF98     26 MACHID    EQU   $BF98
0000:          C030     27 CLICK     EQU   $C030
0000:          FC58     28 HOME      EQU   $FC58
0000:          FCA8     29 WAIT      EQU   $FCA8
0000:          FD0C     30 RDKEY     EQU   $FD0C
0000:          FD6F     31 GTLN1     EQU   $FD6F
0000:          FD8E     32 CROUT     EQU   $FD8E
0000:          FDED     33 COUT      EQU   $FDED
0000:                   34 ;
0000:                   35 ;ProDOS MLI function call codes
0000:                   36 ;
0000:          0080     37 RDBL_C    EQU   $80
0000:          0081     38 WRTBL_C   EQU   $81
0000:                   39 ;
----- NEXT OBJECT FILE NAME IS DOUBLER
2000:          2000     40           ORG   SYSBEGIN
2000:                   41           MSB   ON
2000:                   42 ;
2000:                   43 ; Greeting message
2000:                   44 ;  and opportunity to escape
2000:                   45 ;
2000:20 58 FC           46           JSR   HOME
2003:A0 21              47           LDY   #<HEADING
2005:A9 B3              48           LDA   #>HEADING
2007:20 79 21           49           JSR   MSG
200A:18                 50           CLC
200B:                   51 ;
200B:                   52 ; Error message only if we
200B:                   53 ;  return here with carry set
200B:                   54 ;
200B:90 0A    2017      55 REPEAT    BCC   NOERR
200D:A0 23              56           LDY   #<ERROR
200F:A9 79              57           LDA   #>ERROR
2011:20 79 21           58           JSR   MSG
2014:20 9A 21           59           JSR   PROBELL
2017:A0 22              60 NOERR     LDY   #<PROMPT
2019:A9 48              61           LDA   #>PROMPT
201B:20 79 21           62           JSR   MSG
201E:20 0C FD           63           JSR   RDKEY
2021:C9 D9              64           CMP   #'Y'
2023:F0 05    202A      65           BEQ   GO
2025:C9 F9              66           CMP   #'y'
2027:F0 01    202A      67           BEQ   GO
2029:60                 68           RTS
202A:                   69 ;
202A:                   70 ; Set up pointers to the ProDOS
202A:                   71 ;  I/O buffer
202A:                   72 ;
202A:A5 73              73 GO        LDA   HIMEM
202C:85 FC              74           STA   PTR1
202E:85 FE              75           STA   PTR2
2030:8D A5 23           76           STA   DATABUF
2033:A5 74              77           LDA   HIMEM+1
2035:85 FD              78           STA   PTR1+1
2037:85 FF              79           STA   PTR2+1
2039:8D A6 23           80           STA   DATABUF+1
203C:E6 FF              81           INC   PTR2+1
203E:                   82 ;
203E:                   83 ; Load block 6 (volume bitmap)
203E:                   84 ;
203E:A0 00              85           LDY   #<6
2040:A9 06              86           LDA   #>6
2042:20 64 21           87           JSR   READ
2045:B0 C4    200B      88           BCS   REPEAT
2047:                   89 ;
2047:                   90 ; Check for empty disk
2047:                   91 ;
2047:A0 22              92           LDY   #34
2049:A9 FF              93           LDA   #$FF
204B:D1 FC              94 CHKLP     CMP   (PTR1),Y
204D:D0 09    2058      95           BNE   NOGOOD
204F:88                 96           DEY
2050:D0 F9    204B      97           BNE   CHKLP
2052:A9 01              98           LDA   #1
2054:D1 FC              99           CMP   (PTR1),Y
2056:F0 0A    2062     100           BEQ   GOOD
2058:A0 23             101 NOGOOD    LDY   #<REJECT
205A:A9 20             102           LDA   #>REJECT
205C:20 79 21          103           JSR   MSG
205F:38                104           SEC
2060:B0 A9    200B     105           BCS   REPEAT
2062:                  106 ;
2062:                  107 ; Clear bits to protect
2062:                  108 ;  blocks 136, 141-143, (track 17)
2062:                  109 ;  and blocks 160-279 (tracks 20-34)
2062:                  110 ;
2062:A0 14             111 GOOD      LDY   #20
2064:A9 00             112           LDA   #0
2066:91 FC             113 LP0       STA   (PTR1),Y
2068:C8                114           INY
2069:C0 23             115           CPY   #35
206B:90 F9    2066     116           BCC   LP0
206D:A0 11             117           LDY   #17
206F:A9 78             118           LDA   #%01111000
2071:91 FC             119           STA   (PTR1),Y
2073:                  120 ;
2073:                  121 ; Write it back to block 6
2073:                  122 ;
2073:A0 00             123           LDY   #<6
2075:A9 06             124           LDA   #>6
2077:20 4F 21          125           JSR   WRITE
207A:B0 8F    200B     126           BCS   REPEAT
207C:                  127 ;
207C:                  128 ; Clear I/O buffer
207C:                  129 ;
207C:A0 00             130           LDY   #0
207E:98                131           TYA
207F:91 FE             132 LP1       STA   (PTR2),Y
2081:88                133           DEY
2082:D0 FB    207F     134           BNE   LP1
2084:91 FC             135 LP2       STA   (PTR1),Y
2086:88                136           DEY
2087:D0 FB    2084     137           BNE   LP2
2089:                  138 ;
2089:                  139 ; Build a DOS 3.3 VTOC and catalog track
2089:                  140 ;  Note that the catalog occupies
2089:                  141 ;  sectors 15,14,5,4,3,2,1, corresponding
2089:                  142 ;  to blocks 141-142 and parts of 136
2089:                  143 ;  and 143
2089:                  144 ;
2089:                  145 ; Put $11,$0F pointer in 1st half
2089:                  146 ;  and $11,$05 pointer in 2nd
2089:                  147 ;
2089:A0 01             148           LDY   #1
208B:A9 11             149           LDA   #$11
208D:91 FC             150           STA   (PTR1),Y
208F:91 FE             151           STA   (PTR2),Y
2091:C8                152           INY
2092:A9 0E             153           LDA   #$0E
2094:91 FC             154           STA   (PTR1),Y
2096:A9 05             155           LDA   #$05
2098:91 FE             156           STA   (PTR2),Y
209A:                  157 ;
209A:                  158 ; Store it in block 143
209A:                  159 ;  (track $11, sctrs $01,$0F)
209A:                  160 ;
209A:A0 00             161           LDY   #<143
209C:A9 8F             162           LDA   #>143
209E:20 4F 21          163           JSR   WRITE
20A1:B0 4F    20F2     164           BCS   REPEAT0
20A3:                  165 ;
20A3:                  166 ; Put $11,$02 pointer in 1st half
20A3:                  167 ;  and $11,$01 pointer in 2nd
20A3:                  168 ;
20A3:A0 02             169           LDY   #2
```

```
20A5:A9 02          170           LDA   #$02
20A7:91 FC          171           STA   (PTR1),Y
20A9:A9 01          172           LDA   #$01
20AB:91 FE          173           STA   (PTR2),Y
20AD:               174 ;
20AD:               175 ; Store it in block 142
20AD:               176 ;  (track $11, sctrs $02,$03)
20AD:               177 ;
20AD:A0 00          178           LDY   #<142
20AF:A9 8E          179           LDA   #>142
20B1:20 4F 21       180           JSR   WRITE
20B4:B0 3C    20F2  181           BCS   REPEAT0
20B6:               182 ;
20B6:               183 ; Put $11,$04 pointer in 1st half
20B6:               184 ;  and $11,$03 pointer in 2nd
20B6:               185 ;
20B6:A0 02          186           LDY   #2
20B8:A9 04          187           LDA   #$04
20BA:91 FC          188           STA   (PTR1),Y
20BC:A9 03          189           LDA   #$03
20BE:91 FE          190           STA   (PTR2),Y
20C0:               191 ;
20C0:               192 ; Store it in block 141
20C0:               193 ;  (track $11, sctrs $04,$05)
20C0:               194 ;
20C0:A0 00          195           LDY   #<141
20C2:A9 8D          196           LDA   #>141
20C4:20 4F 21       197           JSR   WRITE
20C7:B0 29    20F2  198           BCS   REPEAT0
20C9:               199 ;
20C9:               200 ; Build a VTOC in the 1st half
20C9:               201 ;  of the buffer--clear 2nd half
20C9:               202 ; Directory pointer points to $11,$0F
20C9:               203 ;  and bitmap protects tracks $00-$13
20C9:               204 ;
20C9:A0 C1          205           LDY   #$C1
20CB:B9 AA 23       206 LP3       LDA   VTOC,Y
20CE:91 FC          207           STA   (PTR1),Y
20D0:88             208           DEY
20D1:C0 FF          209           CPY   #$FF
20D3:D0 F6    20CB  210           BNE   LP3
20D5:A0 02          211           LDY   #2
20D7:A9 00          212           LDA   #0
20D9:91 FE          213 LP4       STA   (PTR2),Y
20DB:88             214           DEY
20DC:D0 FB    20D9  215           BNE   LP4
20DE:               216 ;
20DE:               217 ; Store it in block 136
20DE:               218 ;
20DE:A0 00          219           LDY   #<136
20E0:A9 88          220           LDA   #>136
20E2:20 4F 21       221           JSR   WRITE
20E5:B0 0B    20F2  222           BCS   REPEAT0
20E7:               223 ;
20E7:               224 ; Load block 0
20E7:               225 ;
20E7:A0 00          226           LDY   #<0
20E9:A9 00          227           LDA   #>0
20EB:20 64 21       228           JSR   READ
20EE:B0 02    20F2  229           BCS   REPEAT0
20F0:90 09    20FB  230           BCC   NAME
20F2:4C 0B 20       231 REPEAT0   JMP   REPEAT
20F5:               232 ;
20F5:               233 ; Get a new name for
20F5:               234 ;  the boot program
20F5:               235 ;
20F5:20 8E FD       236 BADNAME   JSR   CROUT
20F8:20 9A 21       237           JSR   PROBELL
20FB:A0 22          238 NAME      LDY   #<NAMEPROMPT
20FD:A9 AF          239           LDA   #>NAMEPROMPT
20FF:20 79 21       240           JSR   MSG
2102:20 6F FD       241           JSR   GTLN1
2105:E0 06          242           CPX   #6
2107:D0 EC    20F5  243           BNE   BADNAME
2109:8A             244           TXA
210A:A8             245           TAY
210B:               246 ;
210B:               247 ; Set up PTR1 to point to the
210B:               248 ;  filename in the boot code
210B:               249 ;  and PTR2 to point to the
210B:               250 ;  name in the error message
210B:               251 ;  in the boot code
210B:               252 ;
210B:A5 FF          253           LDA   PTR2+1
210D:85 FD          254           STA   PTR1+1
210F:A9 02          255           LDA   #$02
2111:85 FC          256           STA   PTR1
2113:A9 62          257           LDA   #$62
2115:85 FE          258           STA   PTR2
2117:AD A9 23       259           LDA   LENGTH
211A:B9 FF 01       260 NMLP      LDA   INPUT-1,Y
211D:C9 E0          261           CMP   #$E0
211F:90 02    2123  262           BCC   UC
2121:29 DF          263           AND   #$DF
2123:C9 AE          264 UC        CMP   #'.'
2125:90 CE    20F5  265           BCC   BADNAME
2127:C9 DB          266           CMP   #'['
2129:B0 CA    20F5  267           BCS   BADNAME
212B:91 FE          268           STA   (PTR2),Y
212D:29 7F          269           AND   #$7F
212F:91 FC          270           STA   (PTR1),Y
2131:88             271           DEY
2132:D0 E6    211A  272           BNE   NMLP
2134:               273 ;
2134:               274 ; The boot filename is preceded
2134:               275 ;  by a byte whose high nibble ($2)
2134:               276 ;  says that the file is more than
2134:               277 ;  one block long and whose low
2134:               278 ;  nibble gives the length of the
2134:               279 ;  filename
2134:               280 ;
2134:A9 26          281           LDA   #$26
2136:91 FC          282           STA   (PTR1),Y
2138:20 8E FD       283           JSR   CROUT
213B:               284 ;
213B:               285 ; Write it back to block 0
213B:               286 ;
213B:A0 00          287           LDY   #<0
213D:A9 00          288           LDA   #>0
213F:20 4F 21       289           JSR   WRITE
2142:B0 08    214C  290           BCS   REPEAT1
2144:               291 ;
2144:               292 ; Do it again? . . .
2144:               293 ;
2144:A0 23          294           LDY   #<BYE
2146:A9 3B          295           LDA   #>BYE
2148:20 79 21       296           JSR   MSG
214B:18             297           CLC
214C:4C 0B 20       298 REPEAT1   JMP   REPEAT
214F:               299 ;
214F:               300 ; Subroutine to write selected
214F:               301 ;  block to disk
214F:               302 ;
214F:8D A7 23       303 WRITE     STA   BLOCKNUM
2152:8C A8 23       304           STY   BLOCKNUM+1
2155:20 00 BF       305           JSR   MLI
2158:81             306           DFB   WRTBL.C
2159:A3 23          307           DW    PARMS
215B:90 06    2163  308           BCC   WROK
215D:20 8B BE       309           JSR   BADCALL
2160:20 0C BE       310           JSR   PRINTERR
2163:60             311 WROK      RTS
2164:               312 ;
2164:               313 ; Subroutine to read selected
2164:               314 ;  block from disk
2164:               315 ;
2164:8D A7 23       316 READ      STA   BLOCKNUM
2167:8C A8 23       317           STY   BLOCKNUM+1
216A:20 00 BF       318           JSR   MLI
216D:80             319           DFB   RDBL.C
216E:A3 23          320           DW    PARMS
2170:90 06    2178  321           BCC   RDOK
2172:20 8B BE       322           JSR   BADCALL
2175:20 0C BE       323           JSR   PRINTERR
2178:60             324 RDOK      RTS
2179:               325 ;
2179:               326 ; Message subroutine
2179:               327 ;
2179:84 FD          328 MSG       STY   PTR1+1
217B:85 FC          329           STA   PTR1
217D:A0 00          330           LDY   #0
217F:B1 FC          331 MSGLP     LDA   (PTR1),Y
2181:F0 16    2199  332           BEQ   MSGOUT
2183:2C 98 BF       333           BIT   MACHID
2186:30 06    218E  334           BMI   CHROUT
2188:C9 E0          335           CMP   #$E0
218A:90 02    218E  336           BCC   CHROUT
218C:29 DF          337           AND   #%11011111
218E:20 ED FD       338 CHROUT    JSR   COUT
2191:E6 FC          339           INC   PTR1
2193:D0 EA    217F  340           BNE   MSGLP
2195:E6 FD          341           INC   PTR1+1
2197:D0 E6    217F  342           BNE   MSGLP
2199:60             343 MSGOUT    RTS
219A:               344 ;
219A:               345 ; ProDOS bell
219A:               346 ;
219A:A9 20          347 PROBELL   LDA   #$20
219C:85 3C          348           STA   A1L
219E:A9 02          349 BLP       LDA   #$2
21A0:20 A8 FC       350           JSR   WAIT
21A3:8D 30 C0       351           STA   CLICK
21A6:A9 24          352           LDA   #$24
21A8:20 A8 FC       353           JSR   WAIT
21AB:8D 30 C0       354           STA   CLICK
21AE:C6 3C          355           DEC   A1L
21B0:D0 EC    219E  356           BNE   BLP
21B2:60             357           RTS
```

LISTING 1: DOUBLER *(continued)*

```
21B3:                  358 ;
21B3:                  359 ; Messages
21B3:                  360 ;
21B3:A0 A0 A0 A0       361 HEADING   ASC          DOUBLER by Ken Manly'
21B7:A0 A0 A0 C4
21BB:CF D5 C2 CC
21BF:C5 D2 A0 E2
21C3:F9 A0 CB E5
21C7:EE A0 CD E1
21CB:EE EC F9
21CE:8D                362           DFB   $8D
21CF:C3 EF F0 F9       363           ASC   'Copyright (C) by MicroSPARC, Inc'
21D3:F2 E9 E7 E8
21D7:F4 A0 A8 C3
21DB:A9 A0 E2 F9
21DF:A0 CD E9 E3
21E3:F2 EF D3 D0
21E7:C1 D2 C3 AC
21EB:A0 C9 EE E3
21EF:8D 8D             364           DFB   $8D,$8D
21F1:C4 CF D5 C2       365           ASC   'DOUBLER will create a DOS 3.3/ProDOS'
21F5:CC C5 D2 A0
21F9:F7 E9 EC EC
21FD:A0 E3 F2 E5
2201:E1 F4 E5 A0
2205:E1 A0 C4 CF
2209:D3 A0 B3 AE
220D:B3 AF D0 F2
2211:EF C4 CF D3
2215:8D                366           DFB   $8D
2216:E8 F9 E2 F2       367           ASC   'hybrid disk from a newly formatted'
221A:E9 E4 A0 E4
221E:E9 F3 EB A0
2222:E6 F2 EF ED
2226:A0 E1 A0 EE
222A:E5 F7 EC F9
222E:A0 E6 EF F2
2232:ED E1 F4 F4
2236:E5 E4
2238:8D                368           DFB   $8D
2239:D0 F2 EF C4       369           ASC   'ProDOS disk.'
223D:CF D3 A0 E4
2241:E9 F3 EB AE
2245:8D 8D 00          370           DFB   $8D,$8D,0
2248:D0 F5 F4 A0       371 PROMPT    ASC   'Put the disk to be altered in drive 1'
224C:F4 E8 E5 A0
2250:E4 E9 F3 EB
2254:A0 F4 EF A0
2258:E2 E5 A0 E1
225C:EC F4 E5 F2
2260:E5 E4 A0 E9
2264:EE A0 E4 F2
2268:E9 F6 E5 A0
226C:B1
226D:8D                372           DFB   $8D
226E:E1 EE E4 A0       373           ASC   'and type "Y" to continue.'
2272:F4 F9 F0 E5
2276:A0 A2 D9 A2
227A:A0 F4 EF A0
227E:E3 EF EE F4
2282:E9 EE F5 E5
2286:AE
2287:8D 8D             374           DFB   $8D,$8D
2289:C1 EE F9 A0       375           ASC   'Any other key will end the program.'
228D:EF F4 E8 E5
2291:F2 A0 EB E5
2295:F9 A0 F7 E9
2299:EC EC A0 E5
229D:EE E4 A0 F4
22A1:E8 E5 A0 F0
22A5:F2 EF E7 F2
22A9:E1 ED AE
22AC:0D 0D 00          376           DFB   $8D,$8D,0
22AF:D4 F9 F0 E5       377 NAMEPROMPT ASC  'Type a six-character name for the boot'
22B3:A0 E1 A0 F3
22B7:E9 F8 AD E3
22BB:E8 E1 F2 E1
22BF:E3 F4 E5 F2
22C3:A0 EE E1 ED
22C7:E5 A0 E6 EF
22CB:F2 A0 F4 E8
22CF:E5 A0 E2 EF
22D3:EF F4
22D5:8D                378           DFB   $8D
22D6:F0 F2 EF E7       379           ASC   'program (such as PRODOS or DBBOOT)--'
22DA:F2 E1 ED A0
22DE:A8 F3 F5 E3
22E2:E8 A0 E1 F3
22E6:A0 D0 D2 CF
22EA:C4 CF D3 A0
22EE:EF F2 A0 C4
22F2:C2 C2 CF CF
22F6:D4 A9 A0 A0
22FA:8D                380           DFB   $8D
22FB:EC E5 F4 F4       381           ASC   'letters and numerals only, please.'
22FF:E5 F2 F3 A0
```

LISTING 1: DOUBLER *(continued)*

```
2303:E1 EE E4 A0
2307:EE F5 ED E5
230B:F2 E1 EC F3
230F:A0 EF EE EC
2313:F9 AC A0 F0
2317:EC E5 E1 F3
231B:E5 AE
231D:8D 8D 00          382           DFB   $8D,$8D,0
2320:CE EF F4 A0       383 REJECT    ASC   'Not an empty ProDOS disk'
2324:E1 EE A0 E5
2328:ED F0 F4 F9
232C:A0 D0 F2 EF
2330:C4 CF D3 A0
2334:E4 E9 F3 EB
2338:8D 8D 00          384           DFB   $8D,$8D,0
233B:CF CB AC A0       385 BYE       ASC   "OK, That one's done . . ."
233F:D4 E8 E1 F4
2343:A0 EF EE E5
2347:A7 F3 A0 E4
234B:EF EE E5 A0
234F:AE A0 AE A0
2353:AE
2354:8D                386           DFB   $8D
2355:C9 E6 A0 F9       387           ASC   'If you want, you can do it again.'
2359:EF F5 A0 F7
235D:E1 EE F4 AC
2361:A0 F9 EF F5
2365:A0 E3 E1 EE
2369:A0 E4 EF A0
236D:E9 F4 A0 E1
2371:E7 E1 E9 EE
2375:AE
2376:8D 8D 00          388           DFB   $8D,$8D,0
2379:C4 E9 F3 EB       389 ERROR     ASC   "Disk error--unable to complete the job."
237D:A0 E5 F2 F2
2381:EF F2 AD AD
2385:F5 EE E1 E2
2389:EC E5 A0 F4
238D:EF A0 E3 EF
2391:ED F0 EC E5
2395:F4 E5 A0 F4
2399:E8 E5 A0 EA
239D:EF E2 AE
23A0:8D 8D 00          390           DFB   $8D,$8D,0
23A3:                  391 ;
23A3:                  392 ; Parameter table for BLOCK READ
23A3:                  393 ;  and BLOCK WRITE
23A3:                  394 ;
23A3:03                395 PARMS     DFB   3
23A4:60                396 UNITNUM   DFB   $60
23A5:00 00             397 DATABUF   DW    0
23A7:00 00             398 BLOCKNUM  DW    0
23A9:                  399 ;
23A9:          0001    400 LENGTH    DS    1
23AA:                  401 ;
23AA:                  402 ; Image of DOS 3.3 VTOC
23AA:                  403 ;  (slightly modified)
23AA:                  404 ;
23AA:04 11 0F 03       405 VTOC      DFB   4,$11,$0F,3,0,1,0,0,0,0,0,0
23AE:00 01 00 00
23B2:00 00 00 00
23B6:00 00 00 00       406           DFB   0,0,0,0,0,0,0,0,0,0,0,0
23BA:00 00 00 00
23BE:00 00 00 00
23C2:00 00 00 00       407           DFB   0,0,0,0,0,0,0,0,0,0,0,0
23C6:00 00 00 00
23CA:00 00 00 00
23CE:00 00 00 7A       408           DFB   0,0,0,$7A,0,0,0,0,0,0,0,0
23D2:00 00 00 00
23D6:00 00 00 00
23DA:14 01 00 00       409           DFB   $14,1,0,0,$23,$10,0,1,0,0,0,0
23DE:23 10 00 01
23E2:00 00 00 00
23E6:00 00 00 00       410           DFB   0,0,0,0,0,0,0,0,0,0,0,0
23EA:00 00 00 00
23EE:00 00 00 00
23F2:00 00 00 00       411           DFB   0,0,0,0,0,0,0,0,0,0,0,0
23F6:00 00 00 00
23FA:00 00 00 00
23FE:00 00 00 00       412           DFB   0,0,0,0,0,0,0,0,0,0,0,0
2402:00 00 00 00
2406:00 00 00 00
240A:00 00 00 00       413           DFB   0,0,0,0,0,0,0,0,0,0,0,0
240E:00 00 00 00
2412:00 00 00 00
2416:00 00 00 00       414           DFB   0,0,0,0,0,0,0,0,0,0,0,0
241A:00 00 00 00
241E:00 00 00 00
2422:00 00 00 00       415           DFB   0,0,0,0,0,0,0,0,0,0,0,0
2426:00 00 00 00
242A:00 00 00 00
242E:00 00 00 00       416           DFB   0,0,0,0,$FF,$FF,0,0,$FF,$FF,0,0
2432:FF FF 00 00
2436:FF FF 00 00
243A:FF FF 00 00       417           DFB   $FF,$FF,0,0,$FF,$FF,0,0,$FF,$FF,0,0
243E:FF FF 00 00
2442:FF FF 00 00
```

```
2446:FF FF 00 00   418           DFB   $FF,$FF,0,0,$FF,$FF,0,0,$FF,$FF,0,0
244A:FF FF 00 00
244E:FF FF 00 00
2452:FF FF 00 00   419           DFB   $FF,$FF,0,0,$FF,$FF,0,0,$FF,$FF,0,0
2456:FF FF 00 00
245A:FF FF 00 00
245E:FF FF 00 00   420           DFB   $FF,$FF,0,0,$FF,$FF,0,0,$FF,$FF,0,0
2462:FF FF 00 00
2466:FF FF 00 00
246A:FF FF         421           DFB   $FF,$FF

** SUCCESSFUL ASSEMBLY := NO ERRORS,0,$FF,$FF,0,0
2432:FF FF 00 00
2436:FF FF 00 00
243A:FF FF 00 00   417           DFB   $FF,$FF,0,0,$FF,$FF,0,0,$FF,$FF,0,0
243E:FF FF 00 00
2442:FF FF 00 00
2446:FF FF 00 00   418           DFB   $FF,$FF,0,0,$FF,$FF,0,0,$FF,$FF,0,0
244A:FF FF 00 00
244E:FF FF 00 00
2452:FF FF 00 00   419           DFB   $FF,$FF,0,0,$FF,$FF,0,0,$FF,$FF,0,0
2456:FF FF 00 00
245A:FF FF 00 00
245E:FF FF 00 00   420           DFB   $FF,$FF,0,0,$FF,$FF,0,0,$FF,$FF,0,0
2462:FF FF 00 00
2466:FF FF 00 00
246A:FF FF         421           DFB   $FF,$FF
```

END OF LISTING 1

LISTING 2: DOSLOADER

```
0000:                1           LST   ON,NOA
0000:                2 ;
0000:                3 ********************************
0000:                4 *                              *
0000:                5 *          DOSLOADER           *
0000:                6 *        by Ken Manly          *
0000:                7 *    Buffalo Chip Software     *
0000:                8 *                              *
0000:                9 *     Copyright (C)  1985      *
0000:               10 *     by MicroSPARC, Inc.      *
0000:               11 *     Concord, MA.  01742      *
0000:               12 *                              *
0000:               13 ********************************
0000:               14 ;
0000:        003C   15 A1L        EQU   $3C
0000:        003E   16 A2L        EQU   $3E
0000:        0042   17 A4L        EQU   $42
0000:        2000   18 SYSBEGIN   EQU   $2000
0000:        2080   19 BEGIN      EQU   $2080
0000:        22FF   20 DOSLENGTH  EQU   $22FF
0000:        9D00   21 DOS        EQU   $9D00
0000:        9D84   22 DOSCLD     EQU   $9D04
0000:        C082   23 RDROM      EQU   $C082
0000:        FE2C   24 MOVE       EQU   $FE2C
0000:        FE89   25 SETKBD     EQU   $FE89
0000:        FE93   26 SETVID     EQU   $FE93
0000:               27 ;
0000:               28           MSB   ON
----- NEXT OBJECT FILE NAME IS DOSLOADER
2000:        2000   29           ORG   SYSBEGIN
2000:               30 ;
2000:               31 ; Make sure Applesoft is on
2000:               32 ;
2000:AD 82 C0       33           LDA   RDROM
2003:               34 ;
2003:               35 ; Disconnect ProDOS
2003:               36 ;
2003:20 89 FE       37           JSR   SETKBD
2006:20 93 FE       38           JSR   SETVID
2009:               39 ;
2009:               40 ; DOS 3.3 image occupies $2080-$437F
2009:               41 ; Set up monitor to move it back
2009:               42 ;  to $9D00-$BFFF
2009:               43 ;
2009:A9 20          44           LDA   #<BEGIN
200B:85 3D          45           STA   A1L+1
200D:A9 80          46           LDA   #>BEGIN
200F:85 3C          47           STA   A1L
2011:A9 43          48           LDA   #<BEGIN+DOSLENGTH
2013:85 3F          49           STA   A2L+1
2015:A9 7F          50           LDA   #>BEGIN+DOSLENGTH
2017:85 3E          51           STA   A2L
2019:A9 9D          52           LDA   #<DOS
201B:85 43          53           STA   A4L+1
201D:A9 00          54           LDA   #>DOS
201F:85 42          55           STA   A4L
2021:A0 00          56           LDY   #0
2023:               57 ;
2023:               58 ; Move it
2023:               59 ;
2023:20 2C FE       60           JSR   MOVE
2026:               61 ;
2026:               62 ; Jump to DOS cold start
2026:               63 ;
2026:4C 84 9D       64           JMP   DOSCLD
```

END OF LISTING 2

LISTING 3: DBBOOT

```
0000:                1           LST   ON,NOA,G
0000:                2 ;
0000:                3 ********************************
0000:                4 *                              *
0000:                5 *          DBBOOT              *
0000:                6 *        by Ken Manly          *
0000:                7 *    Buffalo Chip Software     *
0000:                8 *                              *
0000:                9 *     Copyright (C)  1985      *
0000:               10 *     by MicroSPARC, Inc.      *
0000:               11 *     Concord, MA.  01742      *
0000:               12 *                              *
0000:               13 ********************************
0000:               14 *   Apple ProDOS Assembler     *
0000:               15 ********************************
0000:               16 ;
0000:        003C   17 A1L        EQU   $3C
0000:        003E   18 A2L        EQU   $3E
0000:        0042   19 A4L        EQU   $42
0000:        009B   20 ESC        EQU   $9B
0000:        00FE   21 PTR1       EQU   $FE
0000:        0841   22 REBOOT     EQU   $841
0000:        0902   23 PATHBUF    EQU   $902
0000:        0962   24 ERRNAME    EQU   $962
0000:        2000   25 SYSBEGIN   EQU   $2000
0000:        2100   26 RELOC      EQU   $2100
0000:        C000   27 KBD        EQU   $C000
0000:        C00C   28 CLR80VID   EQU   $C00C
0000:        C010   29 KBSTRB     EQU   $C010
0000:        C030   30 CLICK      EQU   $C030
0000:        C061   31 PBUTN0     EQU   $C061
0000:        C062   32 PBUTN1     EQU   $C062
0000:        C082   33 ROMRD      EQU   $C082
0000:        C083   34 RAMRD      EQU   $C083
0000:        E000   35 BASICBYTE  EQU   $E000
0000:        FB2F   36 INIT       EQU   $FB2F
0000:        FBB3   37 IDBYTE     EQU   $FBB3
0000:        FC58   38 HOME       EQU   $FC58
0000:        FCA8   39 WAIT       EQU   $FCA8
0000:        FD0C   40 RDKEY      EQU   $FD0C
0000:        FD8E   41 CROUT      EQU   $FD8E
0000:        FDDA   42 PRBYTE     EQU   $FDDA
0000:        FDED   43 COUT       EQU   $FDED
0000:        FE89   44 SETKBD     EQU   $FE89
0000:        FE93   45 SETVID     EQU   $FE93
0000:        FE84   46 SETNORM    EQU   $FE84
0000:        FF3A   47 BELL       EQU   $FF3A
0000:               48 ;
0000:               49 ;
0000:               50           MSB   ON
0000:        FF3A   51           SYS
----- NEXT OBJECT FILE NAME IS DBBOOT
2000:        2000   52           ORG   SYSBEGIN
2000:               53 ;
2000:               54 ; Jump over flags
2000:               55 ;
2000:4C 06 20       56           JMP   BEGIN
2003:               57 ;
2003:               58 ; Option flags
2003:               59
2003:               60 ; Automatic-0; Menu=$80
2003:80             61 MENUFL    DFB   $80
2004:               62 ;
2004:               63 ; ProDOS=0; DOS3.3=$80
2004:00             64 DFLTFL    DFB   0
2005:               65 ;
```

LISTING 3: DBBOOT *(continued)*

```
2005:                   66 ; Max prompting=0; min prompting=$80
2005:00                 67 XPRTFL   DFB   0
2006:                   68 ;
2006:                   69 ; Identify the computer
2006:                   70 ;
2006:AD B3 FB           71 BEGIN    LDA   IDBYTE
2009:C9 06              72          CMP   #$06
200B:F0 01   200E       73          BEQ   SETIIE
200D:18                 74          CLC
200E:6E 60 22           75 SETIIE   ROR   IIE
2011:                   76 ;
2011:                   77 ; Make sure ROM is on and
2011:                   78 ;  initialize keyboard & screen
2011:                   79 ;
2011:AD 82 C0           80          LDA   ROMRD
2014:8D 0C C0           81          STA   CLR80VID
2017:20 84 FE           82          JSR   SETNORM
201A:20 2F FB           83          JSR   INIT
201D:20 93 FE           84          JSR   SETVID
2020:20 89 FE           85          JSR   SETKBD
2023:                   86 ;
2023:                   87 ; Print the title
2023:                   88 ;  and the copyright notice
2023:                   89 ;
2023:20 58 FC           90          JSR   HOME
2026:A0 21              91          LDY   #<HEADER
2028:A9 70              92          LDA   #>HEADER
202A:20 36 21           93          JSR   MSG
202D:20 57 21           94          JSR   PROBELL
2030:                   95 ;
2030:                   96 ; Assume we will load DOS 3.3, and
2030:                   97 ;  put its name in the pathname buffer
2030:                   98 ;
2030:AE 69 22           99          LDX   NM3.3
2033:8A                100          TXA
2034:09 20             101          ORA   #$20
2036:8D 02 09          102          STA   PATHBUF
2039:BD 69 22          103 LP1      LDA   NM3.3,X
203C:9D 62 09          104          STA   ERRNAME,X
203F:29 7F             105          AND   #$7F
2041:9D 02 09          106          STA   PATHBUF,X
2044:CA                107          DEX
2045:D0 F2   2039      108          BNE   LP1
2047:                  109 ;
2047:                  110 ; Check to see if we have 64K of memory
2047:                  111 ;  by reading a byte from the language
2047:                  112 ;  card area, changing it, and seeing
2047:                  113 ;  if the altered byte can be stored
2047:                  114 ;
2047:18                115          CLC
2048:6E 61 22          116          ROR   MEM
204B:AD 83 C0          117          LDA   RAMRD
204E:AD 83 C0          118          LDA   RAMRD
2051:AD 00 E0          119          LDA   BASICBYTE
2054:A8                120          TAY
2055:49 55             121          EOR   #$55
2057:8D 00 E0          122          STA   BASICBYTE
205A:4D 00 E0          123          EOR   BASICBYTE
205D:8C 00 E0          124          STY   BASICBYTE
2060:8D 82 C0          125          STA   ROMRD
2063:F0 0E   2073      126          BEQ   BIG64
2065:                  127 ;
2065:                  128 ; If we do not have 64K, start the
2065:                  129 ;  installation message and suppress
2065:                  130 ;  the message that offers an
2065:                  131 ;  alternative.
2065:                  132 ;
2065:A0 21             133          LDY   #<INSTMS
2067:A9 BD             134          LDA   #>INSTMS
2069:20 36 21          135          JSR   MSG
206C:38                136          SEC
206D:6E 05 20          137          ROR   XPRTFL
2070:4C F9 20          138          JMP   DOS3
2073:38                139 BIG64    SEC
2074:6E 61 22          140          ROR   MEM
2077:                  141 ;
2077:                  142 ; We do have 64K--check to see if
2077:                  143 ;  we should offer a menu
2077:                  144 ;
2077:2C 03 20          145          BIT   MENUFL
207A:10 32   20AE      146          BPL   AUTO
207C:                  147 ;
207C:                  148 ; Display menu
207C:                  149 ;  and check the response
207C:                  150 ;
207C:A0 21             151          LDY   #<MENU
207E:A9 E6             152          LDA   #>MENU
2080:20 36 21          153          JSR   MSG
2083:20 0C FD          154 QUERY    JSR   RDKEY
2086:C9 E0             155          CMP   #$E0
2088:90 02   208C      156          BCC   UC
208A:29 D0             157          AND   #$D0
208C:C9 D0             158 UC       CMP   #'P'
208E:F0 06   2096      159          BEQ   GOODKEY
2090:C9 C4             160          CMP   #'D'
2092:F0 02   2096      161          BEQ   GOODKEY
2094:D0 ED   2083      162          BNE   QUERY
2096:48                163 GOODKEY  PHA
2097:20 ED FD          164          JSR   COUT
209A:20 8E FD          165          JSR   CROUT
209D:20 8E FD          166          JSR   CROUT
20A0:A0 21             167          LDY   #<INSTMS
20A2:A9 BD             168          LDA   #>INSTMS
20A4:20 36 21          169          JSR   MSG
20A7:                  170 ;
20A7:                  171 ; Retrieve the keystroke and test
20A7:                  172 ;  whether the lower nibble is 4
20A7:                  173 ;  (D) or 0 (P)
20A7:                  174 ;
20A7:68                175          PLA
20A8:29 04             176          AND   #$04
20AA:F0 2A   20D6      177          BEQ   PRODOS
20AC:D0 4B   20F9      178          BNE   DOS3
20AE:                  179 ;
20AE:                  180 ; We come here if we are not
20AE:                  181 ;  using the menu
20AE:                  182 ;
20AE:                  183 ; See if user is requesting the
20AE:                  184 ;  non-default system by holding one
20AE:                  185 ;  of the Apple keys/paddle buttons
20AE:                  186 ;  or by typing <ESC>
20AE:                  187 ; Notice that if both buttons seem
20AE:                  188 ;  to be down, we assume that we
20AE:                  189 ;  have a ][+ with no paddles at
20AE:                  190 ;  all, and we do not load 3.3!
20AE:                  191 ;
20AE:A0 21             192 AUTO     LDY   #<INSTMS
20B0:A9 BD             193          LDA   #>INSTMS
20B2:20 36 21          194          JSR   MSG
20B5:AD 61 C0          195          LDA   PBUTN0
20B8:4D 62 C0          196          EOR   PBUTN1
20BB:30 0E   20CB      197          BMI   ALT
20BD:AD 00 C0          198          LDA   KBD
20C0:C9 9B             199          CMP   #ESC
20C2:F0 07   20CB      200          BEQ   ALT
20C4:2C 04 20          201 DFLT     BIT   DFLTFL
20C7:10 0D   20D6      202          BPL   PRODOS
20C9:30 2E   20F9      203          BMI   DOS3
20CB:38                204 ALT      SEC
20CC:6E 05 20          205          ROR   XPRTFL
20CF:2C 04 20          206          BIT   DFLTFL
20D2:10 25   20F9      207          BPL   DOS3
20D4:30 00   20D6      208          BMI   PRODOS
20D6:                  209 ;
20D6:                  210 ; We are clear to load ProDOS
20D6:                  211 ;  --print a message
20D6:                  212 ;  announcing that fact
20D6:                  213 ;
20D6:A0 21             214 PRODOS   LDY   #<PDMS
20D8:A9 D3             215          LDA   #>PDMS
20DA:20 36 21          216          JSR   MSG
20DD:20 57 21          217          JSR   PROBELL
20E0:                  218 ;
20E0:                  219 ; Put the PRODOS name
20E0:                  220 ;  in the pathname buffer
20E0:                  221 ;  and in the error message
20E0:                  222 ;
20E0:AE 62 22          223          LDX   NMPSYS
20E3:8A                224          TXA
20E4:09 20             225          ORA   #$20
20E6:8D 02 09          226          STA   PATHBUF
20E9:BD 62 22          227 LP2      LDA   NMPSYS,X
20EC:9D 62 09          228          STA   ERRNAME,X
20EF:29 7F             229          AND   #$7F
20F1:9D 02 09          230          STA   PATHBUF,X
20F4:CA                231          DEX
20F5:D0 F2   20E9      232          BNE   LP2
20F7:F0 0A   2103      233          BEQ   INSTALL
20F9:                  234 ;
20F9:                  235 ; Print a message announcing DOS 3.3
20F9:                  236 ;
20F9:A0 21             237 DOS3     LDY   #<D3MS
20FB:A9 DC             238          LDA   #>D3MS
20FD:20 36 21          239          JSR   MSG
2100:20 3A FF          240          JSR   BELL
2103:                  241 ;
2103:                  242 ; If we are not in expert mode
2103:                  243 ;  and if we are not in menu mode
2103:                  244 ;  we want to tell everyone that
2103:                  245 ;  they could have had the other
2103:                  246 ;  operating system by using
2103:                  247 ;  <ESC> or one of the Apple keys
2103:                  248 ;
2103:2C 03 20          249 INSTALL  BIT   MENUFL
2106:30 28   2130      250          BMI   INSTALL3
2108:2C 05 20          251          BIT   XPRTFL
210B:30 23   2130      252          BMI   INSTALL3
210D:A0 22             253          LDY   #<ALTMS
210F:A9 0D             254          LDA   #>ALTMS
```

```
2111:20 36 21       255           JSR   MSG
2114:2C 04 20       256           BIT   DFLTFL
2117:10 07   2120   257           BPL   INSTALL1
2119:A0 21          258           LDY   #<PDMS
211B:A9 D3          259           LDA   #>PDMS
211D:4C 24 21       260           JMP   INSTALL2
2120:A0 21          261 INSTALL1  LDY   #<D3MS
2122:A9 DC          262           LDA   #>D3MS
2124:20 36 21       263 INSTALL2  JSR   MSG
2127:               264 ;
2127:               265 ; Let the message sit on the
2127:               266 ;  screen for a few seconds
2127:               267 ;
2127:A2 B0          268           LDX   #$B0
2129:8A             269 WTLP      TXA
212A:20 A8 FC       270           JSR   WAIT
212D:CA             271           DEX
212E:D0 F9   2129   272           BNE   WTLP
2130:               273 ;
2130:               274 ; Clear the keyboard and run whatever
2130:               275 ;  is called for in the pathname buffer
2130:               276 ;
2130:AD 10 C0       277 INSTALL3  LDA   KBSTRB
2133:4C 41 08       278           JMP   REBOOT
2136:               279 ;
2136:               280 ; Message subroutine
2136:               281 ;
2136:84 FF          282 MSG       STY   PTR1+1
2138:85 FE          283           STA   PTR1
213A:A0 00          284           LDY   #0
213C:B1 FE          285 MSGLP     LDA   (PTR1),Y
213E:F0 16   2156   286           BEQ   MSGOUT
2140:2C 60 22       287           BIT   IIE
2143:30 06   214B   288           BMI   CHROUT
2145:C9 E0          289           CMP   #$E0
2147:90 02   214B   290           BCC   CHROUT
2149:29 DF          291           AND   #%11011111
214B:20 ED FD       292 CHROUT    JSR   COUT
214E:E6 FE          293           INC   PTR1
2150:D0 EA   213C   294           BNE   MSGLP
2152:E6 FF          295           INC   PTR1+1
2154:D0 E6   213C   296           BNE   MSGLP
2156:60             297 MSGOUT    RTS
2157:               298 ;
2157:               299 ; Pretty ProDOS bell
2157:               300 ;
2157:A9 20          301 PROBELL   LDA   #$20
2159:85 3C          302           STA   A1L
215B:A9 02          303 BLP       LDA   #$2
215D:20 A8 FC       304           JSR   WAIT
2160:8D 30 C0       305           STA   CLICK
2163:A9 24          306           LDA   #$24
2165:20 A8 FC       307           JSR   WAIT
2168:8D 30 C0       308           STA   CLICK
216B:C6 3C          309           DEC   A1L
216D:D0 EC   215B   310           BNE   BLP
216F:60             311           RTS
2170:               312 ;
2170:A0 A0 A0 A0    313 HEADER    ASC   '     Doubleboot System by Ken Manly'
2174:A0 C4 EF F5
2178:E2 EC E5 E2
217C:EF EF F4 A0
2180:D3 F9 F3 F4
2184:E5 ED A0 E2
2188:F9 A0 CB E5
218C:EE A0 CD E1
2190:EE EC F9
2193:8D             314           DB    $8D
2194:C3 EF F0 F9    315           ASC   'Copyright (C) 1985 by MicroSPARC, Inc.'
2198:F2 E9 E7 E8
219C:F4 A0 A8 C3
21A0:A9 A0 B1 B9
21A4:B8 B5 A0 E2
21A8:F9 A0 CD E9
21AC:E3 F2 EF D3
21B0:D0 C1 D2 C3
21B4:AC A0 C9 EE
21B8:E3 AE
21BA:8D 8D 00       316           DB    $8D,$8D,0
21BD:A0 A0 A0 A0    317 INSTMS    ASC   '          Installing '
21C1:A0 A0 A0 A0
21C5:A0 A0 C9 EE
21C9:F3 F4 E1 EC
21CD:EC E9 EE E7
21D1:A0
21D2:00             318           DB    0
21D3:D0 F2 EF C4    319 PDMS      ASC   'ProDOS'
21D7:CF D3
21D9:8D 8D 00       320           DB    $8D,$8D,0
21DC:C4 CF D3 A0    321 D3MS      ASC   'DOS 3.3'
21E0:B3 AE B3
21E3:8D 8D 00       322           DB    $8D,$8D,0
21E6:D4 F9 F0 E5    323 MENU      ASC   'Type <P> for ProDOS; <D> for DOS 3.3  '
21EA:A0 BC D0 BE
21EE:A0 E6 EF F2
21F2:A0 D0 F2 EF
21F6:C4 CF D3 BB
21FA:A0 BC C4 BE
21FE:A0 E6 EF F2
2202:A0 C4 CF D3
2206:A0 B3 AE B3
220A:A0 A0
220C:00             324           DB    0
220D:A0 A0 A0 A0    325 ALTMS     ASC   '          Reboot and type <ESC>'
2211:A0 A0 A0 A0
2215:A0 D2 E5 E2
2219:EF EF F4 A0
221D:E1 EE E4 A0
2221:F4 F9 F0 E5
2225:A0 BC C5 D3
2229:C3 BE
222B:8D             326           DB    $8D
222C:A0 A0 A0 A0    327           ASC   '          or hold an Apple key'
2230:A0 A0 A0 A0
2234:A0 EF F2 A0
2238:E8 EF EC E4
223C:A0 E1 EE A0
2240:C1 F0 F0 EC
2244:E5 A0 EB E5
2248:F9
2249:8D             328           DB    $8D
224A:A0 A0 A0 A0    329           ASC   '          to install '
224E:A0 A0 A0 A0
2252:A0 A0 F4 EF
2256:A0 E9 EE F3
225A:F4 E1 EC EC
225E:A0
225F:00             330           DB    0
2260:               331 ;
2260:       0001    332 IIE       DS    1
2261:       0001    333 MEM       DS    1
2262:06 D0 D2 CF    334 NMPSYS    STR   'PRODOS'
2266:C4 CF D3
2269:06 C4 CF D3    335 NM3.3     STR   'DOS3.3'
226D:B3 AE B3
2270:               336 ;
2270:       2270    337 END       EQU   *

** SUCCESSFUL ASSEMBLY := NO ERRORSE1 EC EC
225E:A0
225F:00             330           DB    0
2260:               331 ;
2260:       0001    332 IIE       DS    1
2261:       0001    333 MEM       DS    1
2262:06 D0 D2 CF    334 NMPSYS    STR   'PRODOS'
2266:C4 CF D3
2269:06 C4 CF D3    335 NM3.3     STR   'DOS3.3'
226D:B3 AE B3
2270:               336 ;
2270:       2270    337 END       EQU   *
```

END OF LISTING 3

LISTING 4: CONFIGURE

```
100  REM ************************
110  REM *      CONFIGURE         *
120  REM *  (CONFIGURES DBBOOT)   *
130  REM *      BY KEN MANLY      *
140  REM *BUFFALO CHIP SOFTWARE*
150  REM * COPYRIGHT (C) 1985    *
160  REM * BY MICROSPARC, INC.   *
170  REM * CONCORD, MA 01742     *
180  REM ************************
190  DIM FL(2): TEXT : HOME : VTAB 2: PRINT "
     THIS PROGRAM WILL HELP YOU SET THE": PRINT
     : PRINT "OPTIONS AVAILABLE IN THE PROGRA
     M DBBOOT"
200  VTAB 8: PRINT "YOU WILL NEED A DISK WITH
      AN UNLOCKED": PRINT : PRINT "COPY OF DB
     BOOT.  IF YOU ARE READY, TYPE": PRINT : PRINT
     "<RETURN> TO CONTINUE.  ANY OTHER KEY": PRINT
     : PRINT "WILL END THE PROGRAM.": PRINT
210  GET Q$: PRINT Q$: IF Q$ <  > CHR$ (13) THEN
     320
220  VTAB 8: CALL  - 958: PRINT "DBBOOT CAN O
     FFER YOU A MENU WHEN IT RUNS": PRINT "OR
      IT CAN AUTOMATICALLY RUN EITHER": PRINT
     : PRINT "PRODOS OR DOS 3.3 BY DEFAULT."
230  VTAB 18: PRINT "  1  AUTOMATIC": PRINT "
       2  MENU": PRINT : PRINT "CHOOSE ONE ";
     : GET Q$: PRINT Q$: ON Q$ <  > "1" AND Q
     $ <  > "2" GOTO 230:FL(0) = (Q$ = "2"): IF
     FL(0) THEN 290
240  VTAB 8: CALL  - 958: PRINT "DBBOOT CAN R
     UN EITHER PRODOS OR DOS 3.3": PRINT : PRINT
     "BY DEFAULT."
250  VTAB 18: PRINT "  1  PRODOS": PRINT "  2
       DOS 3.3": PRINT : PRINT "CHOOSE ONE ";
     : GET Q$: PRINT Q$: ON Q$ <  > "1" AND Q
     $ <  > "2" GOTO 250:FL(1) = (Q$ = "2")
260  VTAB 8: CALL  - 958: PRINT "WHEN IT BOOT
     S ONE SYSTEM BY DEFAULT,": PRINT : PRINT
     "DBBOOT CAN PAUSE TO REMIND YOU HOW YOU"
     : PRINT : PRINT "COULD HAVE CHOSEN THE O
     THER SYSTEM.": PRINT
270  PRINT "THIS IS A HELP FOR NOVICES BUT JU
     ST A": PRINT : PRINT "NUISANCE FOR EXPER
     TS."
280  VTAB 18: PRINT "  1  NOVICE": PRINT "  2
       EXPERT": PRINT : PRINT "CHOOSE ONE ";:
      GET Q$: PRINT Q$: ON Q$ <  > "1" AND Q$
      <  > "2" GOTO 280:FL(2) = (Q$ = "2")
290  VTAB 8: CALL  - 958: PRINT "NO OTHER CHO
     ICES NEED TO BE MADE.": PRINT : PRINT "T
     YPE <Y> TO MODIFY THE COPY OF": PRINT : PRINT
     "DBBOOT THAT'S ON THIS DISK.  ANY OTHER"
     : PRINT : PRINT "KEY WILL QUIT WITH NO C
     HANGES MADE.  ";
300  GET Q$: PRINT Q$: IF Q$ <  > "Y" AND Q$ <
      > "y" THEN 320
310 D$ =  CHR$ (4):N$ = "DBBOOT.A$2000": PRINT
     D$"BLOAD"N$",TSYS": FOR I = 0 TO 2: POKE
     8195 + I,128 * FL(I): NEXT : PRINT D$"BS
     AVE"N$",L$28F,TSYS"
320  HOME : PRINT "GOODBYE": END
```

END OF LISTING 4