

```

LEN (OT$) > 15 OR LEFT$ (
OT$,1) < "A" GOTO 200: IF
OT$ = "SCR" THEN PRINT "
OUTPUT TO SCREEN.": GOTO
230
16 210 PRINT "SLOT#, DRIVE# OF "
;OT$;": ";: INPUT S$,DR$:
ON S$ < "1" OR S$ > "7"
OR DR$ < "1" OR DR$ > "2"
GOTO 210: OS = VAL (S$): O
D = VAL (DR$)
F1 220 IF IN$ = OT$ AND IS = OS
AND ID = OD THEN PRINT :
PRINT BL$ "FILE NAMES MUST
DIFFER!": PRINT "DISASTRO
US RESULTS OCCUR OTHERWIS
E.": GOTO 200
EB 230 PRINT : INPUT "ENTER KEY
FOR CODE: ";A$:KY = VAL (
A$): IF KY = 0 THEN PRINT
"RANGE FOR KEY IS 1E-38
<--> 1E+38.": GOTO 230
A4 240 L = 0: HOME : PRINT "MAKE
CERTAIN THAT THE DISKETT
E WITH": PRINT : PRINT IN
$; "IS IN SLOT #";IS; ", D
RIVE #";ID: PRINT : IF OT
$ < "SCR" THEN PRINT "A
ND THE DISKETTE FOR": PRIN
T : PRINT OT$; "IS IN SL
OT #";OS; ", DRIVE #";OD
87 250 RD = 1:XYZ = RND ( - ABS
(KY)): PRINT : PRINT "HIT
<RETURN> TO CODE, <ESC>
TO RESTART ": GET A$: ON
A$ < > CHR$ (13) AND A$
< > CHR$ (27) GOTO 250: O
N A$ = CHR$ (27) GOTO 160
: HOME : PRINT
4E 260 PRINT : PRINT : PRINT : H
TAB (4): IF ED = 1 THEN P
RINT "EN";
42 270 IF ED = - 1 THEN PRINT "D
E";
17 280 PRINT "CODING. PLEASE WA
IT FOR 'BEEP'":X = FRE (
PEEK ( - 16368)): REM CLE
AR KEYBOARD STROBE.
EB 290 REM ** ENCODE/DECODE **
37 300 S = IS:DR = ID: PRINT D$
+ "OPEN" + IN$ + ",S";IS;
",D";ID
76 310 IF OT$ = "SCR" THEN HOME
: GOTO 330
16 320 S = OS:DR = OD: PRINT D$
+ "OPEN" + OT$ + ",S";OS;
",D";OD: PRINT D$ + "CLOS
E" + OT$: PRINT D$ + "DEL
ETE" + OT$: PRINT D$ + "O
PEN" + OT$:HT = EI + EI *
PF
E3 330 O$ = "": IF OT$ < > "SCR"
THEN VTAB EI: HTAB (HT):
L = L + WN: PRINT " LINE
#";L
21 340 FOR R = ZR TO TFF: REM S
ET A MAXIMUM OF 256 CHARA
CTERS PER INPUT LINE
F6 350 S = IS:DR = ID: PRINT D$
+ "READ" + IN$
41 360 GET A$: PRINT : PRINT D$:
REM USE 'GET' RATHER TH
AN 'INPUT' SO -ALL- CHARA
CTERS CAN BE CAPTURED
29 370 A = ASC (A$): IF A = RTN
THEN 510: REM END OF LIN
E SIGNALLED BY 'RETURN' I
NPUT
F0 380 IF (A < TT) OR (A > OTE -
WN AND A < SX) THEN 490:
REM PASS ALL NON-ALPHA
NUMERIC INPUT
10 390 B = A + ED * INT ( RND (R
D) * NF): IF RD = ZR THEN

```

```

RD = WN: REM CREATE A N
EW ASCII VALUE USING THE
OLD VALUE AND A RANDOM #
FROM 0 TO NF
FA 400 IF A > = OTE THEN 440
86 410 IF B > = OTE THEN B = B -
OTE + TT
7C 420 IF B < TT THEN B = NF + W
N + B
18 430 GOTO 460
42 440 IF B > TFF THEN B = B - (
TFF + WN) + SX
86 450 IF B < SX THEN B = NF + W
N + B
81 460 IF B = RTN THEN RD = ZR:
GOTO 490: REM CAN'T USE
A 'RETURN' IN OUTPUT STRI
NG
CA 470 IF (B < TT) OR (B > = OTE
AND B < SX) THEN RD = ZR
: GOTO 490: REM DON'T AL
TER NON-ALPHANUMERIC OUTP
UT
28 480 A = B
43 490 O$ = O$ + CHR$ (A)
FC 500 NEXT
86 510 IF OT$ = "SCR" THEN 530
85 520 S = OS:DR = OD: PRINT D$
+ "WRITE" + OT$
84 530 PRINT O$
FA 540 PRINT D$
45 550 GOTO 330: REM KEEP GETTIN
G UNTIL THERE'S AN END-OF
-DATE ERROR, WHICH WILL B
E CONSIDERED THE END OF T
HE FILE
68 560 PRINT "CONVERSION COMPLET
E.": PRINT "WOULD YOU LIK
E TO DO ANOTHER?": PRINT
"<Y>ES OR <N>O ": GET
A$: PRINT A$: ON A$ < >
"Y" AND A$ < > "N" GOTO 5
60: IF A$ = "Y" THEN RUN
86 570 TEXT : PRINT : PRINT "APP
LECODER DONE.": END
62 580 REM ** ERROR HANDLING **
88 590 ECN = PEEK (222):ELN = PE
EK (218) + 256 * PEEK (21
9)
FE 600 PRINT : PRINT D$ + "CLOSE
" + IN$: PRINT
F1 610 PRINT D$ + "CLOSE" + OT$:
PRINT : PRINT BL$:
88 620 ON ECN = 5 GOTO 560: REM
'END OF FILE' HAS BEEN R
EACHED.
11 630 PRINT "ERROR! HIT <RETURN
> TO CONTINUE.":X = PEEK
( - 16368): GET A$: PRIN
T A$: IF A$ < > CHR$ (13
) THEN HTAB (1): GOTO 630
ED 640 PRINT
33 650 IF ECN = 4 THEN PRINT "TH
E DISKETTE IN SLOT #";S; ",
DRIVE #";DR: PRINT "IS
WRITE PROTECTED. PLEASE
REMOVE THE": PRINT "WRITE
PROTECT TAB.": GOSUB 790
: GOTO 240
E3 660 IF ECN < > 8 THEN 690
AB 670 PRINT "INPUT/OUTPUT ERROR
. ": PRINT "CHECK DISK DR
IVE #";DR; "IN SLOT #";S;
","
53 680 PRINT : PRINT "CODING MUS
T BE RESTARTED.": GOSUB 7
90: GOTO 240
05 690 IF ECN = 9 THEN PRINT "TH
E DISKETTE IN SLOT #";S; ",
DRIVE #";DR: PRINT "IS
FULL. PLEASE INSERT A DI
SKETTE": PRINT "WITH MORE
SPACE AND RESTART.": GOS
UB 790: GOTO 240

```

```

51 700 IF ECN < > 10 THEN 740
41 710 PRINT "THE FILE ";OT$; ",
IS LOCKED.": PRINT "UNLOC
K IT NOW?": PRINT "<Y>ES
OR <N>O ": GET A$: PRINT
A$: ON A$ < > "Y" AND A$
< > "N" GOTO 710
3E 720 IF A$ = "Y" THEN PRINT D$
+ "UNLOCK" + OT$ + ",S";
OS; ",D";OD: PRINT "FILE H
AS BEEN UNLOCKED."
A4 730 GOSUB 790: GOTO 240
9C 740 IF ECN = 11 THEN PRINT "E
RROR IN FILE NAME.": PRIN
T "FILE NAME MUST CONFORM
TO YOUR DOS.": GOSUB 790
: GOTO 180
48 750 IF ECN = 13 THEN PRINT "F
ILE-TYPE MISMATCH.": PRIN
T "ONLY TEXT FILES CAN BE
CODED.": PRINT "PLEASE C
HECK YOUR FILE-TYPE BY":
PRINT "LOOKING AT THE 'CA
TALOG'": GOSUB 790: GOTO
180
48 760 IF ECN = 255 THEN PRINT "
YOU TYPED A <CTRL-C>": P
RINT "IF THIS WAS DONE DU
RING CODING.": PRINT "THE
<CTRL-C> MAY HAVE BEEN S
ENT": PRINT "TO THE OUTPU
T FILE.": PRINT "REDO IF
THIS IS THE CASE.": GOSUB
790: GOTO 240
88 770 PRINT "ERROR NUMBER: ";EC
N: PRINT "IN LINE NUMBER:
";ELN: PRINT "PLEASE REF
ER TO YOUR PROGRAMMING":
PRINT "MANUAL FOR A FULL
DESCRIPTION.": PRINT "THI
S SCREEN WILL REMAIN DISP
LAYED": PRINT "UNTIL YOU
HIT <RESET>":R = 0
88 780 ON R > 0 GOTO 780: TEXT :
PRINT : PRINT "APPLECODE
R DONE.":R = 1: GOTO 780
C1 790 PRINT : PRINT "HIT <RETUR
N> TO RESTART, <ESC> TO E
ND ": GET A$: PRINT A$:
ON A$ < > CHR$ (13) AND A
$ < > CHR$ (27) GOTO 790:
IF A$ = CHR$ (13) THEN R
ETURN
69 800 PRINT : PRINT "VERIFY: EX
IT APPLECODE": PRINT "<Y
>ES OR <N>O ": GET A$: P
RINT A$: ON A$ < > "Y" A
ND A$ < > "N" GOTO 800: O
N A$ = "N" GOTO 790: TEXT
: PRINT : PRINT : PRINT
"APPLECODER DONE.": END

```

Program 2: Filemaker

```

98 80 D$ = CHR$ (4): REM DISK C
OMMAND PREFIX
57 90 N$ = "STRANGE": PRINT D$ +
"OPEN" + N$: PRINT D$ + "
WRITE" + N$: REM CREATE TH
E TEXT FILE 'STRANGE' AND
PREPARE TO WRITE TO IT.
79 100 PRINT "5L+>AYF&4S;": REM
THIS IS THE PHRASE WHICH
APPLECODER WILL DECODE. C
AN YOU GUESS WHAT IT SAYS
NOW?
E2 110 PRINT D$ + "CLOSE" + N$
88 120 PRINT "''" + N$ + "''" + "
HAS BEEN CREATED.": PRINT
"CREATE.PHRASE DONE."
8F 130 END

```

128 File Viewer

Jeffrey D. Partch

This Commodore 128 utility packs a double punch. It can print the contents of any disk file or disassemble any machine language program directly from disk. Neither operation disturbs the program currently in memory. A disk drive is required.

Have you ever wanted to know the contents of a mysterious file in the disk directory, or needed to look at another file during a programming session? All too often, getting that information requires a lot of saving, loading, and listing. "128 File Viewer" allows you to display the contents of any program (PRG) or sequential (SEQ) disk file on the screen without harming the program in memory. It can also disassemble any machine language program directly from disk—again, without disturbing the current program. Since it adds a new command to the Commodore 128's BASIC, this program is very easy to use.

Because File Viewer is written entirely in machine language, it must be typed in using the "MLX" machine language entry program found elsewhere in this issue. Be sure to read and understand the instructions for using MLX before you begin entering the data for File Viewer. When you run MLX, you'll be asked for a starting address and an ending address for the data you'll be entering. Here are the addresses for File Viewer:

Starting address: 1350
Ending address: 17E7

When you finish entering the data, be sure to save a copy to disk before you leave MLX. Once you have a completed copy of File Viewer on disk, you can activate it with a command of the form:

BOOT "FILE VIEWER",Ddrive,Udevice

Of course, you should replace **FILE VIEWER** in this statement with the filename you used when saving the data with MLX. If your disk drive is device 8, the normal device number for Commodore drives, you can omit everything after the closing quotation mark (just use **BOOT "FILE VIEWER"**). The first optional parameter is used to specify the drive number for dual-drive systems with drive 1 in addition to drive 0. (Commodore 1541 and 1571 drives are always drive 0.) The second parameter is used to specify a device number other than 8. For instance, **BOOT "FILE VIEWER",D0,U9** boots the program from a disk in a drive addressed as device 9.

When the familiar **READY** prompt reappears, File Viewer has been installed and is ready to use. (Note that pressing the reset switch deactivates File Viewer.) This program works as an extension of the Commodore 128's BASIC, so it's as easy to use as any other BASIC statement. Here is the general syntax:

VIEW "filename,type",Ddrive,Udevice

Again, the last two parameters are not needed if you are using a single 1541 or 1571 disk drive addressed as device 8.

The *type* parameter is also optional in most cases, since File Viewer ordinarily determines for itself whether the file is a sequential or program file. If necessary, however, you can specify the type by including an *S* for sequential files or a *P* for program files. For instance, the statement **VIEW "SAMPLE,S"** displays the contents of the sequential file **SAMPLE**, while **VIEW "SAMPLE,P"** displays the contents of a program file of the same name.

File Viewer ordinarily reads and displays the entire file. You can slow the display by pressing the Commodore key or pause it completely by pressing **NO SCROLL**. Press **STOP** if you wish to terminate the display before you reach the end of the file.

Disassembly

To aid machine language programmers, File Viewer also includes a disassembly option. Here's the syntax to use:

VIEW "filename,M"

The *M* stands for machine language. It's not a Commodore file type, but simply a signal to File Viewer that you wish to disassemble the file rather than print it to the screen. When you choose this option, File Viewer reads the file from disk and disassembles it to the screen in standard 6502 assembler

format. Just as with the display option, you can slow the disassembly with the Commodore key, pause it with NO SCROLL, or cut it off by pressing STOP.

Redirecting Output

In most cases you'll want to look at a file on the screen. However, File Viewer also lets you divert the normal screen output to a printer or other peripheral device. This option is most useful for machine language disassemblies, since it allows you to create a hardcopy printout of the program which can be studied at leisure. However, you can also use it as a quick way to print a text file without loading it into memory.

Diverting output requires that you open a logical file to the desired device. For instance, say that you want to send the contents of a file to the printer. The statement OPEN 1,4 opens logical file 1 to the printer (which is usually device 4). Once the logical file is open, you must tell File Viewer where to send its output. This is done by adding the logical file number to the VIEW statement:

```
VIEW# file number,"filename,type",
Drive, Udevice
```

The logical file number must match the one you used when you opened the logical file, and must be in the range 1-127. As with normal format for the statement, the type, drive number, and device number parameters are optional. For example, if you open a file to the printer with OPEN 1,4, this statement makes a hardcopy printout of the file SAMPLE from a disk in the drive addressed as device 8:

```
VIEW#1,"SAMPLE"
```

In special cases, you may wish to divert output to a disk file, or even to a modem via the RS-232 interface. File Viewer isn't picky about where it sends output, as long as you have properly opened a logical file to the device. The *Commodore 128 System Guide* explains the syntax needed to open a file to disk or RS-232 interface.

Special Concerns

The VIEW statement works only in direct mode; you should not attempt to add it to a program. VIEW does not accept string or numeric

variables in place of its parameters (you can't use a statement like F\$ = "SAMPLE": VIEW F\$).

The file display mode of File Viewer is intended primarily for looking at text files—that is, files that consist of printable character codes. It is possible to view other types of files such as tokenized BASIC programs, but the display may be difficult or impossible to read in such cases. For instance, if the file contains values equivalent to control-code characters, displaying the file may clear the screen, change the printing color, and so forth. If you try to print a hardcopy of such a file, some of the values may be interpreted as spurious printer control codes, causing strange behavior such as unwanted form feeds or a change in printing mode. Similarly, disassembling a file that doesn't contain a machine language program produces meaningless results, but does no real harm.

The machine language for File Viewer occupies memory locations 4944-6114 (\$1350-\$17E2). If you disturb the contents of this area while File Viewer is active, the computer will probably lock up. The program also uses locations 4864-4937 (\$1300-\$1349) and 250-254 (\$FA-\$FE) for temporary storage. You may use these locations for your own purposes; however, every VIEW statement will overwrite the contents of these areas.

File Viewer also maintains a six-byte bank-switching routine beginning at location 2048 (\$0800) in both banks 0 and 1. This is necessary because some BASIC errors may occur while the system is operating in bank 1. In bank 0, this area is at the bottom of the BASIC runtime stack, where it isn't likely to be disturbed unless you run a program that uses a very large number of nested GOSUBs or FOR-NEXT loops. To accommodate this routine in bank 1, File Viewer bumps the start of variables up to address 2054 (\$0806) when you first BOOT the program, thereby protecting the bank 1 copy of the switching routine from being destroyed by the system. This results in a slight reduction of the amount of space available for scalar variables, but should have no noticeable effect on the operation of BASIC.

128 File Viewer

Please refer to the "MLX" article in this issue before entering the following program.

```
1350:20 97 13 20 FA 51 78 A9 88
1358:6F 8D 14 03 A9 13 8D 15 16
1360:03 58 A9 56 8D 00 0A A9 E2
1368:13 8D 01 0A 4C 03 40 0A CB
1370:06 88 30 1B B9 00 08 D9 2B
1378:C9 13 F0 F5 20 97 13 BA 06
1380:A9 00 9D 01 01 A9 13 9D B1
1388:07 01 A9 91 9D 06 01 4C 14
1390:65 FA A2 10 4C 29 14 A0 4D
1398:06 A9 08 8D 01 03 85 FB 21
13A0:85 30 A9 00 8D 00 03 85 C2
13A8:FA 84 2F A9 EF 8D 27 03 F4
13B0:A9 79 8D 26 03 88 A9 FA A6
13B8:8D B9 02 A2 01 B9 C9 13 14
13C0:91 FA 20 77 FF 88 10 F3 20
13C8:60 8D 03 FF 4C CF 13 20 CA
13D0:51 16 E0 0B D0 53 A0 FF 07
13D8:A2 04 A5 3E 85 FB A5 3D 8E
13E0:38 E9 04 85 FA B0 02 C6 DB
13E8:FB C8 CA 30 09 B9 36 14 4B
13F0:D1 FA F0 F5 D0 1C 24 7F FA
13F8:30 21 A0 00 C6 FA 30 3A 50
1400:B1 FA C9 20 F0 F4 C9 3A 24
1408:D0 08 F0 2E A2 0E 2C A2 E3
1410:17 2C A2 0B 2C A2 09 2C FD
1418:A2 04 2C A2 22 2C A2 08 51
1420:2C A2 05 2C A2 1E 2C A2 F2
1428:80 86 FC 20 A3 17 20 97 25
1430:13 A6 FC 4C 3F 4D 56 49 15
1438:45 57 20 51 16 20 86 03 33
1440:C9 23 D0 32 20 46 17 90 2C
1448:06 C9 2C F0 0E D0 C3 29 DE
1450:0F 20 27 17 20 80 03 D0 38
1458:EE F0 B7 E6 3D A5 FC F0 05
1460:AB 30 A9 20 59 FF 90 04 91
1468:A2 03 D0 C7 85 FE E0 04 27
1470:90 A3 A2 00 86 FC A6 98 2C
1478:E0 0A 90 04 A2 01 D0 B3 54
1480:20 C0 FF 20 87 16 C9 2C 42
1488:F0 09 A2 3F 86 FB D0 1C 95
1490:4D 50 53 C8 A2 02 B1 3D 28
1498:DD 90 14 F0 06 CA 10 F6 D7
14A0:4C 12 14 E6 3D E6 3D 88 EC
14A8:85 FB B1 3D 20 B4 16 20 BC
14B0:90 03 F0 07 C9 22 D0 E8 D1
14B8:20 D4 16 A5 FC D0 04 A9 20
14C0:08 85 FC A0 02 98 20 59 04
14C8:FF B0 04 A8 C8 10 F6 85 22
14D0:FD 85 05 20 53 17 A5 FE 3D
14D8:F0 0A A9 17 8D 27 03 A9 5B
14E0:B0 8D 26 03 20 DB 17 A4 FC
14E8:FB C0 53 F0 0F C0 50 F0 C5
14F0:21 C0 4D D0 03 4C A8 15 40
14F8:C9 01 F0 16 20 D2 FF 8A 9C
1500:20 D2 FF 20 CF FF 24 90 48
1508:08 20 D2 FF 20 6E 16 4F
1510:D0 F1 A0 20 A9 6A 85 FA 32
1518:A9 51 85 FB B1 FA 99 00 88
1520:13 88 10 FB A9 05 8D 1B 1F
1528:13 C8 B9 CF 17 99 21 13 B6
1530:C8 C0 06 90 F5 A9 D0 20 49
1538:D2 FF A0 03 20 CF FF 99 E9
1540:FA 00 88 10 F7 24 90 08 73
1548:68 20 6E 16 C8 B9 FA 00 01
1550:99 64 00 C0 01 90 F5 A2 45
1558:90 38 20 75 8C 20 44 8E 30
1560:20 E2 55 A9 20 20 D2 FF BF
1568:20 CF FF F0 C8 C9 CE F0 A1
1570:25 C9 FE F0 24 30 15 20 BA
1578:D2 FF C9 22 D0 EA 20 CF A9
1580:FF F0 B2 20 D2 FF C9 22 8B
1588:F0 DE D0 F2 AA A9 44 A0 51
1590:17 20 00 13 D0 D2 A0 C9 5C
1598:2C A0 09 20 CF FF 09 80 35
15A0:18 E9 01 AA A9 46 D0 E9 0E
15A8:48 A0 49 B9 0F B6 99 00 6A
15B0:13 88 10 F7 A9 FB 8D 3F 9F
15B8:13 A9 FA 8D 45 13 A9 13 EB
15C0:8D 34 13 A9 3E 8D 33 13 5D
15C8:68 85 FA 85 66 86 FB 86 0C
15D0:67 A9 0D 20 D2 FF 20 9B 2F
```

```

15D8:B8 A9 2D 20 D2 FF 20 CF 18
15E0:FF 24 90 08 48 20 A5 B8 6D
15E8:68 A0 00 84 67 84 68 99 6F
15F0:06 00 20 59 B6 48 AE AB 97
15F8:0A 86 FC C8 CA 30 0C 20 45
1600:CF FF 99 06 00 20 A5 B8 2C
1608:4C FB 15 A9 12 20 60 16 7E
1610:68 A2 03 20 A1 B6 A2 06 AE
1618:86 66 20 00 13 A9 20 20 C4
1620:60 16 A9 3B 20 D2 FF A0 D7
1628:00 E6 FC B9 06 00 48 29 33
1630:7F C9 20 68 B0 02 A9 20 1A
1638:20 D2 FF C8 C4 FC 90 EB DC
1640:68 20 6E 16 A6 FB A5 FA 43
1648:18 65 FC 90 01 E8 4C C9 90
1650:15 A9 00 A0 04 99 FA 00 F7
1658:88 10 FA 85 05 85 7A 60 18
1660:38 E5 EC A8 A9 20 20 D2 2B
1668:FF 88 D0 FA 18 60 48 28 7B
1670:70 12 20 E1 FF F0 01 60 A1
1678:A5 FE F0 05 A9 0D 20 D2 3A
1680:FF 4C 24 14 4C 27 14 20 CC
1688:86 03 C9 22 F0 02 D0 5B A0
1690:A9 30 8D 00 11 A9 3A 8D 80
1698:01 11 E6 3D B1 3D F0 13 B1
16A0:C9 22 F0 0F C9 2C F0 0B 35
16A8:99 02 11 C8 C0 11 D0 EC A9
16B0:4C 0F 14 60 48 C0 00 D0 65
16B8:03 4C 1E 14 98 48 A9 2C E3
16C0:99 02 11 C8 A5 FB 99 02 3B
16C8:11 C8 84 FA 18 68 65 3D 5A
16D0:85 3D 68 60 20 37 17 C9 F7
16D8:44 F0 06 C9 55 F0 1C D0 38
16E0:0A 20 46 17 90 0B 20 3A 7F
16E8:17 B0 10 4C 12 14 4C 0C 19
16F0:14 8D 00 11 C9 32 B0 F6 0B
16F8:20 37 17 C9 55 D0 EC 20 6A
1700:46 17 B0 E7 F0 E5 29 0F 2C
1708:85 FC 20 80 03 F0 0A B0 E4
1710:DA 29 0F 20 27 17 4C 0A 12
1718:17 A5 FC C9 08 B0 03 4C CC
1720:15 14 C9 1F B0 F9 60 48 7F
1728:A5 FC 0A 0A 65 FC 0A 85 03
1730:FC 68 65 FC 65 FC 60 20 74
1738:80 03 F0 14 C9 2C F0 03 AA
1740:4C 12 14 4C 80 03 20 80 31
1748:03 C9 30 F0 F9 4C 86 03 90
1750:68 68 60 A9 00 8D 00 FF A9
1758:AA 20 68 FF A5 FA A0 11 5C
1760:20 BD FF A5 05 A6 FC A4 C9
1768:FD 20 BA FF 20 0F F5 E6 05
1770:B7 E6 B7 20 C0 FF B0 28 BC
1778:A6 05 20 C6 FF 20 CF FF CB
1780:48 A9 02 24 90 D0 16 20 D3
1788:CF FF AA 68 20 D4 17 20 1D
1790:7D FF 0D 56 49 45 57 49 DB
1798:4E 47 0D 60 60 4C 18 14 D9
17A0:4C 21 14 20 CC FF A5 05 78
17A8:20 C3 FF A5 FE 4C C3 FF E2
17B0:20 D4 17 20 CC FF A6 FE BB
17B8:20 C9 FF 20 DB 17 20 79 60
17C0:EF 20 CC FF A6 05 20 C6 D8
17C8:FF 20 DB 17 4C 79 EF 29 3D
17D0:7F 4C D2 FF 85 41 86 42 AC
17D8:84 43 60 A5 41 A6 42 A4 4E
17E0:43 60 00 00 00 00 00 00

```

©

Back issues of *COMPUTE!*, *COMPUTE!'s Gazette*, or any magazine disks can be ordered by calling **800-346-6767** (in NY 212-887-8525). Some issues may no longer be available.

Filedump For IBM PC/PCjr

Harry Faulkner

A file-dump program is useful for anyone who needs to examine the contents of a disk file in detail. This utility was designed originally for use with accounting files, but it can display the contents of any sequential or random disk file. The program runs on any IBM PC with BASICA or MS-DOS computer with compatible BASIC.

Have you ever needed to examine the contents of a disk file? There are MS-DOS commands, DEBUG and TYPE, that provide limited access. The TYPE command displays a file by printing each byte as an ASCII character. DEBUG is somewhat more flexible, but it displays bytes only in ASCII and hexadecimal form.

"Filedump" provides more options than either of these commands. It lets you look at any random or sequential disk file and send the output to either a monitor or a printer. It displays each byte in ASCII form (if it is printable) and prints its integer, single-precision, and double-precision value. The program also allows you to start at any position within the file.

I wrote this program while developing home accounting programs that use both random and

sequential files. I needed a way to see if the programs were writing data to the correct places in these files, without having to run the program to get output. The program was written on a Leading Edge IBM-compatible computer; it runs without modification on the IBM PC with BASICA or PCjr with Cartridge BASIC.

Filedump For IBM PC/PCjr

For instructions on entering this program, please refer to "COMPUTE!'s Guide to Typing in Programs" elsewhere in this issue.

```

61 70 GOSUB 1000 'Initialize t
he field and obtain file f
or opening
KH 80 RETRY=0
ON 100 COUNT=1:EIGHT$=SPACE$(8):
FOUR$=SPACE$(4):TWO$=SPAC
E$(2):ONE$=SPACE$(1):EIGH
TTEMP$=SPACE$(8):FOURTEMP
$=SPACE$(4):TWOTEMP$=SPAC
E$(2) 'initialize variab
les
ME 110 WHILE NOT EOF(1)
KH 120 GET #1
KI 125 IF COUNT<1ST GOTO 190
HH 130 LSET ONE$=INCHAR$
MH 140 EIGHTTEMP$=RIGHT$(EIG
HT$,7):EIGHT$=EIGHTTEMP$+
INCHAR$: 'Add new charac
ter to right end of strin
gs
HI 150 FOURTEMP$=RIGHT$(FOUR
$,3):FOUR$=FOURTEMP$+INCH
AR$
AE 160 TWOTEMP$=RIGHT$(TWO$,
1):TWO$=TWOTEMP$+INCHAR$

```

```

KP 170 INTEGER=CVI(TWO$):SGL
PRE!=CVS(FOUR$):DBLPRE#=C
VD(EIGHT$)
LL 180 GOSUB 2000
PI 190 COUNT=COUNT+1
DN 200 WEND
HM 205 IF COUNT<=IST THEN PRINT
"STARTING BYTE, ";IST;";
PAST ENDING BYTE, ";COUNT
-1;"; OF FILE.":PRINT "PL
EASE TRY AGAIN.":CLOSE #1
:RETRY=1:GOSUB 1014
OP 206 IF RETRY>0 THEN GOTO 80
MH 210 IF POPT%<>1 THEN LPRINT:L
PRINT:LPRINT,TAB(10),"End
of file": ELSE PRINT:PRI
NT:PRINT, TAB(10),"End of
file"
IC 220 CLOSE #1:END
MM 1000 REM *** Initialize print
out, get filename, and i
nitialize field ***
EK 1005 ON ERROR GOTO 1100
MC 1010 CLS
JN 1012 INPUT "Enter name of fil
e to be dumped? ",FILE$
JA 1014 OPEN FILE$ FOR INPUT AS
#1:CLOSE #1:OPEN FILE$ A
S #1 LEN=1: FIELD #1, 1
AS INCHAR$
LH 1016 INPUT "Starting byte #:
",IST
LI 1021 PRINT "Output device: (M
) monitor"
EG 1023 PRINT " (P
) printer"
HP 1025 INPUT " Choice: ",
POPT$:POPT%=CINT(INSTR("
MmPp",POPT$)/2):IF POPT%
<1 OR POPT%>2 THEN BEEP:
PRINT"Invalid entry. Cho
ose M or P": GOTO 1025
EH 1030 IF POPT%=1 THEN CLS:GOSU
B 1500 'Initialize scr
een or printer
FC 1035 IF POPT%=2 THEN GOSUB 15
00 'Initialize screen
or printer
IH 1040 RETURN
IC 1100 IF ERR=53 THEN PRINT "Fi
le, ";FILE$;"; not found
. Please try again.":BEE
P:RESUME 1012
KN 1110 PRINT "Error # ";ERR;" o
ccurred at line ";ERL
HJ 1120 END
FJ 1500 REM *** Subroutine to pr
int header line ***
NB 1510 INIT$=" BYTE # Char
. Int. Sgl. Prec
. val. Dbl. Preci
s. val."
AC 1520 IF POPT%=1 THEN PRINT IN
IT$:PRINT
GJ 1530 IF POPT%=2 THEN LPRINT I
NIT$:LPRINT
JB 1540 RETURN
ED 2000 REM *** Subroutine to ou
tput values. ***
CB 2010 REM *** This sub outputs
15 lines to the monitor
and 50 lines to the
PL 2020 REM *** printer then ask
s if the user wants to o
utput more data.
JH 2030 IF POPT%=1 THEN PRINT TA
B(5); COUNT; TAB(17); ON
E$;
KL 2040 IF POPT%=2 THEN LPRINT T
AB(5); COUNT; TAB(17); O
NE$;
DP 2050 IF COUNT>1 AND POPT%=1 T

```

```

HEN PRINT TAB(23); INTEG
ER;
ED 2060 IF COUNT>1 AND POPT%=2 T
HEN LPRINT TAB(23); INTE
GER;
KN 2070 IF COUNT>3 AND POPT%=1 T
HEN PRINT TAB(34); SGLPR
E!;
JC 2080 IF COUNT>3 AND POPT%=2 T
HEN LPRINT TAB(34); SGLP
RE!;
QI 2090 IF COUNT>7 AND POPT%=1 T
HEN PRINT TAB(55); DBLP
RE#;
MD 2100 IF COUNT>7 AND POPT%=2 T
HEN LPRINT TAB(55); DBLP
RE#;
EK 2110 IF POPT%=1 THEN PRINT
GB 2120 IF POPT%=2 THEN LPRINT
PM 2130 IF POPT%=1 AND ((COUNT-I
ST+1) MOD 15) =0 THEN GO
SUB 2500
NB 2140 IF POPT%=2 AND ((COUNT-I
ST+1) MOD 50) =0 THEN GO
SUB 2500
JN 2150 RETURN
EL 2500 REM *** Subroutine to qu
ery for more output **
*
HE 2510 LOCATE ,,1:PRINT:PRINT "
Hit A to abort, any othe
r key for more "; 'turn
cursor on
MD 2520 EN$=INKEY$:IF EN$="" GOT
O 2520
DC 2530 IF INSTR("Aa",EN$) > 0 T
HEN RETURN 220
HN 2540 IF POPT%=1 THEN CLS
GE 2550 GOSUB 1500
KI 2560 RETURN

```

Use your Commodore Printer with your PC compatible



Upgraded to an IBM PC™ and want to use your Commodore printer? With our new IBM/GBM INTER-FACE you can use your Commodore compatible printer with your PC compatible. Supports virtually all text applications. Some graphics support of graphics characters and graphics printouts when using Commodore printers which support graphics. Cable, disk, and instructions supplied. Requires DOS 2.0 or up. IBM PC is a TM of IBM Corp.

\$34.95

Call (206) 624-4985 now and order. Visa & Mastercard welcome! Add \$2.00 for shipping & handling.

Omnitronix Inc.

760 Harrison St., Seattle, WA 98109

COMPUTE! TOLL FREE Subscription Order Line

1-800-247-5470

In IA 1-800-532-1272

COMPUTE! Disk Subscriptions

COMPUTE! Disks are available for the following computers:

- Apple II series
- Commodore 64 and 128
- Atari 400/800 /XL/XE
- IBM PC and PCjr

Each error-free disk contains all the programs from the previous three issues of *COMPUTE!*. With a disk subscription, you'll receive one disk—for the machine you specify—every three months. To subscribe, call toll free **800-247-5470** (in Iowa 800-532-1272).

LEARN PROGRAMMING



MASTER COMPUTERS IN YOUR OWN HOME

Now you can write programs and get a computer to do just what you want. Get the most out of any computer, and avoid having to pay the high price of pre-packaged software.

LEARN AT YOUR OWN PACE IN YOUR SPARE TIME

Our independent study program allows you to learn about computers, operations, applications and programming in your spare time, at home. Our instructors provide you with one-on-one counseling.

LEARN EVEN BEFORE YOU DECIDE ON A COMPUTER

Everything is explained in simple language. You will enjoy learning to use a computer—EVEN IF YOU DON'T OWN ONE. Learn to program on any personal computer; IBM, APPLE, COMMODORE, TRS, and more.

BE YOUR OWN COMPUTER EXPERT

Programming is the best way to learn to use computers, and we can show you the best—and most economical—way to learn programming! Send today for your free information package. No obligation. No salesman will call.

halix

CENTER FOR COMPUTER EDUCATION

INSTITUTE

1543 W. OLYMPIC # 226 Los Angeles, CA 90015-3894

HALIX INSTITUTE CENTER FOR COMPUTER EDUCATION DEPT 61-3
1543 W. OLYMPIC # 226 LOS ANGELES, CA 90015-3894

YES! Send me information on how I can learn about computers and programming at home!

Name _____ Age _____

Address _____

City _____ State _____ Zip _____

DOS Calc

Steve Kelly

This convenient, menu-driven utility for the Commodore 64 includes all the disk functions of the familiar "DOS Wedge" program, plus a powerful scientific calculator. A disk drive is required to use the program's DOS functions.

"DOS Calc" is two programs in one. Not only is it a complete replacement for the "DOS Wedge" program supplied on the 1541 or 1571 Test/Demo disk, but it has a built-in scientific calculator. The program is entirely menu-driven and employs easy-to-use, pop-down windows.

DOS Calc is written in machine language, so you must enter it with "MLX," the machine language entry program found elsewhere in this issue. Here are the addresses you need to type and save the program with MLX:

Starting address: 0801
Ending address: 16E8

After you have saved a copy of DOS Calc, it can be loaded and run just like any BASIC program. When you run it, DOS Calc relocates itself to the safe memory area beginning at address 49152, a process that takes only an instant.

Once the READY prompt returns, DOS Calc is ready to use. Like the "DOS Wedge," DOS Calc works only in immediate mode (when you are not running a program). To enter DOS Calc, type @ and press RETURN. DOS Calc displays the main menu window with the following selections:

- A. Exit
- B. Directory
- C. Load
- D. Load relocated
- E. Save
- F. Resave
- G. Bsave
- H. Re-Bsave
- I. Scratch
- J. Rename
- K. Copy
- L. Validate
- M. Initialize
- N. Read error channel
- O. New a disk
- P. Calculator

To select a function from the main menu, simply press the letter shown next to the function you want. For instance, pressing A exits DOS Calc and returns you to BASIC. When you choose any other function, DOS Calc opens a new screen window in which the interaction for that function takes place.

Directory

The directory function (B) is perhaps the most frequently used DOS function. When you press B, DOS Calc opens a window and prompts you to enter a wildcard specification. As a convenience, it prints the asterisk (*) wildcard in the window. If you wish to view all the files on the disk, simply press RETURN. To view only selected files, change the wildcard accordingly.

For example, suppose that you want to see all the files beginning with GAME. Type GAME* and press RETURN. After you enter the wildcard, DOS Calc opens a third window which contains all the requested directory information. To exit this window and return to the main menu, press A. If the disk contains more files than will fit in

the window, you can press B to view the next page of the directory. Press C to enter a new wildcard and begin a new directory search.

File Functions

The next eight DOS functions all concern existing files. They are Load, Load Relocated, Resave, Bsave, Re-Bsave, Scratch, Rename, and Copy. After you select any of these functions from the main menu, DOS Calc asks you to enter the name of the file you wish to manipulate. If you enter a null filename (no name), DOS Calc automatically displays the directory for the current disk. This directory differs from the normal directory display, however, in that each filename is prefaced by a letter. To select a file for the current function, simply press the letter key that appears in front of the filename. If you change your mind and decide not to complete the operation, press A to exit.

For instance, let's try the Resave command, which deletes an existing program from disk and replaces it with the BASIC program currently in memory. When you choose Resave from the main menu, DOS Calc opens a window asking for a filename. If you press RETURN without entering a name, DOS Calc opens a secondary directory window from which you can select the file by pressing a single key. Once this is done, the program completes the resave. You should avoid using any wildcards in the directory function when it appears as part of a Resave or Re-Bsave command.

The Copy and Rename commands require two filenames. The

first name you enter is used as the name of the new file for the Copy command or the new name of the existing file for the Rename command. This name must be typed in (it cannot be derived from the directory). The second name indicates the original file; this name can be taken from the directory.

DOS Calc provides two functions that are not included in the "DOS Wedge" program. The B in Bsave and Re-Bsave stands for binary, indicating a binary file containing machine language, graphics data, or something other than a BASIC program. Since these functions save a designated area of memory, you must supply two addresses: the beginning and ending addresses of the area to be saved.

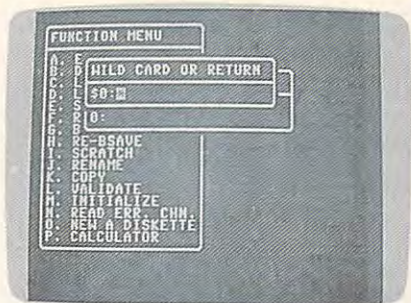
The Validate, Initialize, and Read Error Channel commands require little explanation. Since no information is required for these commands, DOS Calc simply performs them without any introductory prompts. The New command, which formats a disk, requires either one or two items of information. The first item—the disk name—is mandatory. To reformat an already formatted disk, you can supply a new disk name without anything else. To format a disk that has never been used, you must supply a two-character disk ID after the disk name, separating the two items with a comma.

For additional information on DOS operations, refer to the user's manual for your disk drive.

Scientific Calculator

DOS Calc also includes a built-in calculator which is very useful for math, engineering, and various scientific applications. This calculator is different from the familiar pocket calculators used by most people. It uses a notation called *Reverse Polish Notation* (RPN), which allows you to solve complex formulas without parentheses.

When you select the calculator from the main menu, two secondary windows appear on the screen. On the right is the function window, which lists all the calculator functions preceded, as usual, by the letter key which invokes them. To the left is the result window, which is subdivided into three sections.



"DOS Calc" is entirely menu-driven and employs easy-to-use, pop-down windows.

The result area contains either answers from calculations or keyboard input from you. In the middle is the stack display, which holds intermediate results. The bottom area is a message area where the calculator displays error messages such as *Divide by Zero*.

Notation

Before looking at the calculator functions, let's establish some rules for entering numbers. You can enter any number in either standard notation (2534.56 is an example) or in scientific notation (123E34 is an example). Negative numbers are entered somewhat differently than you might expect, since the minus (-) sign is already used to signify subtraction (see below). Use the shifted minus sign (hold down SHIFT and press the minus key) as a substitute. The second trick has to do with the E symbol used in scientific notation. This calculator also uses the letter E for another function (it causes the stack to wrap down; see below). As a substitute, use the ↑ symbol (the up-arrow key directly to the right of the asterisk). Whenever you enter ↑ as part of a number, DOS Calc puts an E in the number you are entering.

Examples

To start with an easy exercise, let's look at how to perform $2 + 5$. In RPN there is no need for an equal sign (=). In this case, you enter both numbers followed by the plus sign (+), which is the operator you wish to use. Press the number 2. DOS Calc prints a 2 in the result window. Now press RETURN. The number 2 shifts to the right and also appears as the top entry in the stack area. Press the number 5. The top

line of the result window now shows that number. Press the plus key. The 2 on the stack is replaced by a 0 and the result line displays a 7. Simple arithmetic operations use the same general procedure. To subtract 2 from 5, you would enter the numbers 5 and 2, followed by the minus key.

More complex equations demonstrate the power of an RPN calculator. Let's solve the equation $2048/(8*4)$. Begin by entering 2048 and pressing RETURN. Now type 8, followed by RETURN, and then a 4. At this point the stack contains all the numbers for the equation, in correct order. Now press the asterisk (*) key to perform multiplication. The result line should display 32. Press the slash (/) to perform division. The calculator displays the final result, which is 64.

Each time you pressed RETURN in this example, the calculator pushed the number from the result line onto the stack. All other numbers were pushed down one location, and the fourth entry on the stack was lost.

There are seven other functions which, like RETURN, have some effect on the calculator's stack. The first is Clear Entry: This function clears the result line and leaves all other entries untouched. The Clear Stack function clears the result line and the entire stack. The next four functions cause the entire stack, including the result line, to wrap or roll in either direction. A wrap allows either the top or bottom entry to move from top to bottom, or vice versa. A roll causes the top or bottom entry to be lost. The seventh stack function, Exchange, swaps the result line with the top stack entry. This operation is useful if numbers are entered in the wrong order and need to be corrected.

All the remaining functions are standard math operations which operate on the result line. Except for Power and Pi, these functions affect only the result line. The Power function raises the number on the top of the stack to the power specified in the result line; the answer is left on the result line and the stack shifts up one location. The Pi function enters the value of π (3.1415926) in the result line just as if you had typed it from the keyboard.

Using an RPN calculator may seem strange at first, but once you become familiar with its powerful features, you may never want to go back to using a simple pocket calculator.

DOS Calc

Please refer to the "MLX" article in this issue before entering the following program.

```
0801:0B 08 0A 00 9E 32 30 36 2E
0809:31 00 00 00 A0 00 A2 10 0C
0811:A9 00 85 FB A9 C0 85 FC BE
0819:A9 01 85 FD A9 08 85 FE 46
0821:B1 FD 91 FB C8 D0 F9 E6 DF
0829:FE E6 FC CA D0 F2 4C 31 DA
0831:C0 AD 08 03 8D AA C0 AD 84
0839:09 03 8D AB C0 A9 4D 8D CF
0841:08 03 A9 C0 8D 09 03 A9 97
0849:FF 8D 6A C1 60 A5 3A C9 FD
0851:FF F0 03 6C AA C0 AD 00 78
0859:02 C9 40 D0 F6 A9 8E 20 8D
0861:D2 FF A9 13 A0 01 A2 01 90
0869:20 84 C1 A9 AD A0 C0 20 0F
0871:E9 C1 A9 BB A0 C0 38 20 70
0879:FB C1 20 66 C2 A9 00 8D AC
0881:AC C0 8D BD CE A9 20 8D 90
0889:8C CE A9 41 A0 52 20 12 7D
0891:C3 AA BD 4A C1 8D 9F C0 CE
0899:BD 4B C1 8D A0 C0 20 E2 97
08A1:FC 4C 7D C0 20 6C C2 6C A3
08A9:02 A0 00 00 00 46 55 4E F4
08B1:43 54 49 4F 4E 20 4D 45 69
08B9:4E 55 FF 45 58 49 54 00 2B
08C1:44 49 52 45 43 54 4F 52 41
08C9:59 00 4C 4F 41 44 00 4C 6C
08D1:4F 41 44 20 52 45 4C 4F F3
08D9:43 41 54 45 44 00 53 41 C4
08E1:56 45 00 52 45 53 41 56 E3
08E9:45 00 42 53 41 56 45 00 08
08F1:52 45 2D 42 53 41 56 45 D7
08F9:00 53 43 52 41 54 43 48 96
0901:00 52 45 4E 41 4D 45 00 FE
0909:43 4F 50 59 00 56 41 4C 58
0911:49 44 41 54 45 00 49 4E 51
0919:49 54 49 41 4C 49 5A 45 A3
0921:00 52 45 41 44 20 45 52 04
0929:52 2E 20 43 48 4E 2E 00 FF
0931:4E 45 57 20 41 20 44 49 05
0939:53 4B 45 54 54 45 00 43 B0
0941:41 4C 43 55 4C 41 54 4F 24
0949:52 FF A4 C0 3B C9 DD CA AC
0951:DA CA 6D CB 6A CB 64 CB 05
0959:67 CB 63 CC E0 CC DD CC 0E
0961:54 CD 58 CD 60 CD 5A CE B6
0969:2C C5 FF 00 41 00 00 00 0D
0971:00 00 00 00 00 00 00 00 83
0979:00 00 00 0A 0D 0E A0 A8 B6
0981:B0 A4 AC B4 48 8A EE 6A AA
0989:C1 AE 6A C1 9D 72 C1 8D 59
0991:6B C1 98 9D 75 C1 68 9D D7
0999:78 C1 E0 00 D0 26 AD A7 96
09A1:02 8D 70 C1 AD A7 02 8D DF
09A9:6F C1 AD A7 02 8D 71 C1 FE
09B1:38 20 F0 FF 8E 6D C1 8C 40
09B9:6E C1 A9 00 8D A7 02 A9 61
09C1:00 8D A7 02 A9 04 85 FC B1
09C9:AE 6A C1 BD 7E C1 85 FE E6
09D1:20 AC C2 AD 6A C1 D0 05 53
09D9:A9 93 20 D2 FF 20 F9 C2 0E
09E1:A9 00 20 CF C2 20 F6 C2 11
09E9:60 A2 41 8E 6C C1 18 20 A0
09F1:FB C1 A9 02 20 CF C2 20 AD
09F9:F6 C2 60 85 FB 84 FC A9 32
0A01:18 90 02 A9 38 8D 15 C2 05
0A09:A9 62 20 D2 FF AC 6A C1 05
0A11:BE 78 C1 CA CA 18 90 16 75
0A19:AD 6C C1 20 D2 FF A9 2E 71
```

```
0A21:20 D2 FF A9 20 20 D2 FF BB
0A29:EE 6C C1 CA CA A0 00 77
0A31:B1 FB F0 07 C9 FF F0 03 DE
0A39:C8 D0 02 A9 20 D2 FF E7
0A41:CA D0 ED B1 FB 85 FD 84 3E
0A49:FE A9 62 20 D2 FF 20 F6 63
0A51:C2 A5 FD D0 10 E6 FE 18 2F
0A59:A5 FB 65 FE 85 FB 90 02 1B
0A61:E6 FC 4C 08 C2 60 A9 04 21
0A69:20 CF C2 60 AE 6A C1 BD 40
0A71:7E C1 85 FC A9 04 85 FE 1D
0A79:20 AC C2 CE 6A C1 AE 6A 30
0A81:C1 E0 FF F0 A0 BD 72 C1 AB
0A89:8D 6B C1 20 F9 C2 60 AD C2
0A91:6F C1 8D A7 02 AD 70 C1 63
0A99:8D A7 02 AD 71 C1 8D A7 CE
0AA1:02 AE 6D C1 AC 6E C1 18 E6
0AA9:20 F0 FF 60 A5 01 29 FE 92
0AB1:85 01 A0 00 84 FB 84 FD F7
0AB9:A2 04 B1 FB 91 FD C8 D0 FC
0AC1:F9 E6 FC E6 FE CA D0 F2 52
0AC9:A5 01 09 01 85 01 60 48 5B
0AD1:AA BD F0 C2 20 D2 FF AC ED
0AD9:6A C1 BE 78 C1 CA CA A9 6B
0AE1:60 20 D2 FF CA D0 F8 6B 7C
0AE9:AA BD F1 C2 20 D2 FF 60 D9
0AF1:75 69 AB B3 6A 6B EE 6B 16
0AF9:C1 AE 6A C1 BD 7B C1 8D F0
0B01:A7 02 BC 75 C1 AE 6B C1 BB
0B09:18 20 F0 FF 60 A9 41 AC 2A
0B11:6C C1 85 FB 84 FC 20 E4 7B
0B19:FF C9 00 F0 F9 C5 FB 90 20
0B21:F5 C5 FC B0 F1 38 E5 FB 86
0B29:0A 60 48 A9 8F 85 FB A9 34
0B31:CE 85 FC 68 85 FD 18 20 AA
0B39:F0 FF A9 00 85 FE A5 C6 37
0B41:85 CC F0 FA 78 A5 CF F0 06
0B49:0C A5 CE AE 87 02 A0 00 19
0B51:84 CF 20 13 EA 58 20 E4 B0
0B59:FF C9 14 F0 21 C9 0D F0 AE
0B61:39 C9 20 90 D9 C9 60 B0 FA
0B69:B5 A6 FD EA FE F0 CF AC 2A
0B71:BD CE 91 FB 20 D2 FF E6 3F
0B79:FE EE BD CE D0 C0 A4 FE 41
0B81:F0 BC C6 FE CE BD CE 88 9B
0B89:A9 9D 20 D2 FF A9 20 20 14
0B91:D2 FF A9 9D 20 D2 FF 4C B8
0B99:3E C3 A6 FE AC BD CE F0 6F
0BA1:0C A9 20 88 D1 FB D0 04 D8
0BA9:88 CA D0 F8 C8 8C BD CE 23
0BB1:8A 60 46 55 4E 43 54 49 B4
0BB9:4F 4E 53 3A FF 45 58 49 28
0BC1:54 00 43 4C 45 41 52 20 23
0BC9:45 4E 54 52 59 00 43 4C 63
0BD1:45 41 52 20 53 54 41 43 D8
0BD9:4B 00 57 52 41 50 20 55 86
0BE1:50 00 57 52 41 50 20 44 FF
0BE9:4F 57 4E 00 52 4F 4C 4C FB
0BF1:20 55 50 00 52 4F 4C 4C 2C
0BF9:20 44 4F 57 4E 00 45 58 E5
0C01:43 48 41 4E 47 45 00 43 6C
0C09:48 41 4E 47 45 20 53 49 6E
0C11:47 4E 00 49 4E 54 45 07 8A
0C19:45 52 00 53 51 55 41 52 52
0C21:45 20 52 4F 4F 54 00 50 3F
0C29:4F 57 45 52 00 50 49 00 60
0C31:53 49 4E 45 00 43 4F 53 62
0C39:49 4E 45 00 54 41 4E 47 BD
0C41:45 4E 54 00 41 52 43 54 48
0C49:41 4E 47 45 4E 54 00 44 DA
0C51:45 47 2D 3E 52 41 44 00 87
0C59:52 41 44 2D 3E 44 45 47 1B
0C61:00 4C 4F 47 2F 45 00 45 BE
0C69:58 50 2F 45 FF 53 54 41 33
0C71:43 4B 3A 00 00 00 00 00 45
0C79:00 00 00 FF 52 45 53 B8
0C81:55 4C 54 3A 00 00 00 FF 85
0C89:4D 45 53 53 41 47 45 53 3E
0C91:3A FF 49 4C 4C 45 47 41 FB
0C99:4C 20 51 55 41 4E 54 49 94
0CA1:54 59 FF 4F 56 45 52 46 E1
0CA9:4C 4F 57 FF 44 49 56 49 E3
```

```
0CB1:53 49 4F 4E 20 42 59 20 71
0CB9:5A 45 52 4F FF 55 4E 4B CC
0CC1:4E 4F 57 4E FF 00 00 00 A4
0CC9:00 00 00 00 00 00 00 00 E1
0CD1:00 00 00 00 00 00 00 00 E9
0CD9:00 00 00 00 00 00 41 42 B6
0CE1:43 44 45 46 47 48 49 4A F1
0CE9:4B 4C 4D 4E 4F 50 51 52 F9
0CF1:53 54 55 56 57 58 59 0D A3
0CF9:AE C6 BF C6 DB C6 EC C6 1A
0D01:F8 C6 04 C7 20 C7 3C C7 A6
0D09:53 C7 5B C7 60 C7 65 C7 5B
0D11:77 C7 88 C7 8D C7 92 C7 DE
0D19:97 C7 9C C7 B1 C7 BA C7 EA
0D21:BF C7 7E C6 8A C6 96 C6 AC
0D29:A2 C6 70 C6 A9 12 A0 15 AC
0D31:A2 00 20 84 C1 A9 B2 A0 A3
0D39:C3 20 E9 C1 A9 BD A0 C3 DF
0D41:38 20 FB C1 20 66 C2 A9 E4
0D49:13 A0 01 A2 00 20 84 C1 AA
0D51:A9 7D A0 C4 20 E9 C1 A9 D5
0D59:6D A0 C4 18 20 FB C1 A9 8A
0D61:02 20 CF C2 20 F6 C2 A9 B6
0D69:88 A0 C4 18 20 FB C1 A9 28
0D71:7A A0 C4 18 20 FB C1 20 9F
0D79:66 C2 AD 00 03 48 AD 01 C2
0D81:03 48 A9 78 8D 00 03 A9 08
0D89:C8 8D 01 03 20 DB C6 20 D9
0D91:F1 C7 A9 00 A0 DE 20 12 9E
0D99:C3 6A 48 A9 13 8D 6B C1 3B
0DA1:20 F9 C2 A9 87 A0 C4 18 9D
0DA9:20 FB C1 68 C9 30 90 07 C8
0DB1:C9 3A B0 03 4C 25 C6 C9 D3
0DB9:5E D0 04 A9 45 D0 F5 C9 75
0DC1:D2 D0 04 A9 2D D0 ED C9 6C
0DC9:2D F0 E9 C9 14 D0 03 4C 47
0DD1:60 C6 A0 00 A2 1A D9 DE F1
0DD9:C4 F0 06 C8 CA D0 F7 F0 5A
0DE1:B1 C0 0C F0 05 A9 00 8D F1
0DE9:7D C6 98 48 20 D4 C7 68 58
0DF1:0A AA BD F8 C4 8D 08 C6 36
0DF9:BD F9 C4 8D 09 C6 20 C4 4B
0E01:C7 BA CA CA 8E 7A C8 20 C5
0E09:07 C6 B0 03 20 CC C7 4C B0
0E11:8F C5 20 20 20 20 20 20 3E
0E19:20 20 20 20 20 20 20 20 45
0E21:20 30 FF 00 00 48 AD 24 F9
0E29:C6 D0 0B AD 7D C6 D0 06 C7
0E31:20 20 C7 20 00 C8 68 8D E1
0E39:24 C6 AC 23 C6 C0 10 F0 2B
0E41:1C 99 12 C6 C8 8C 23 C6 06
0E49:A9 FF 99 12 C6 A9 03 8D FE
0E51:6B C1 20 F9 C2 A9 12 A0 B8
0E59:C6 18 20 FB C1 4C 92 C5 CC
0E61:AC 23 C6 F0 FB 88 A9 FF C1
0E69:99 12 C6 8C 23 C6 D0 2C
0E71:20 20 C7 20 CC C7 A9 01 7A
0E79:8D 7D C6 38 60 00 20 04 5F
0E81:C7 A0 C4 A9 C5 20 67 B8 13
0E89:18 60 20 04 C7 A0 C4 A9 02
0E91:C5 20 50 B8 18 60 20 04 B4
0E99:C7 A0 C4 A9 C5 20 28 BA AE
0EA1:18 60 20 04 C7 A0 C4 A9 1A
0EA9:C5 20 0F BB 18 60 20 6C 3D
0EB1:C2 20 6C C2 68 68 68 8D 34
0EB9:01 03 68 8D 00 03 60 20 E9
0EC1:2E C7 A9 FF 8D 7D C6 A9 B5
0EC9:30 8D 12 C6 A9 FF 8D 13 8B
0ED1:C6 A9 00 8D 23 C6 8D 24 08
0ED9:C6 38 60 A2 19 A0 00 98 81
0EE1:99 C5 C4 C8 CA D0 F9 20 0F
0EE9:C7 C6 38 60 20 04 C7 A2 EB
0EF1:D9 A0 C4 20 D4 BB 38 60 24
0EF9:A9 D9 A0 C4 20 A2 BB 20 E4
0F01:20 C7 18 60 A2 14 A0 00 D0
0F09:B9 CA C4 99 C5 C4 C8 CA 86
0F11:D0 F6 A9 00 A8 A2 05 99 FD
0F19:D9 C4 C8 CA D0 F9 38 60 5A
0F21:A2 14 A0 13 B9 C5 C4 99 E2
0F29:CA C4 88 CA D0 F6 A9 00 51
0F31:A8 A2 05 99 C5 C4 C8 CA 24
0F39:D0 F9 38 60 A2 05 A0 00 B5
```

0F41:B9 CA C4 99 C5 C4 C8 CA BE
0F49:D0 F6 A2 CA A0 C4 20 D4 BB
0F51:BB 38 60 A5 66 49 FF 85 9F
0F59:66 18 60 20 CC BC 18 60 A8
0F61:20 71 BF 18 60 A0 C4 A9 1E
0F69:CA 20 8C BA 20 C4 C7 20 F5
0F71:7B BF 20 04 C7 18 60 A0 81
0F79:AE A9 A8 20 A2 BB AD 7D 4D
0F81:C6 D0 03 20 C7 18 60 4A
0F89:20 6B E2 18 60 20 64 E2 9F
0F91:18 60 20 B4 E2 18 60 20 7B
0F99:0E E3 18 60 A0 C7 A9 AC E4
0FA1:20 A2 BB A0 C4 A9 C5 20 72
0FA9:0F BB 18 60 86 65 2E E0 4E
0FB1:D4 A0 C7 A9 AC 20 28 BA E6
0FB9:18 60 20 EA B9 18 60 20 BD
0FC1:ED BF 18 60 A0 C4 A9 C5 01
0FC9:20 A2 BB 60 A2 C5 A0 C4 50
0FD1:20 D4 BB 60 AD 24 C6 F0 2F
0FD9:17 A9 12 85 7A A9 C6 85 16
0FE1:7B 20 79 00 20 F3 BC 20 5F
0FE9:1B BC 20 CC C7 20 C7 C6 AA
0FF1:60 A9 03 8D 6B C1 20 F9 80
0FF9:C2 A0 C4 A9 C5 20 2E C8 A8
1001:A9 08 8D 6B C1 20 F9 C2 A5
1009:A0 C4 A9 CA 20 2E C8 20 F7
1011:F6 C2 A0 C4 A9 CF 20 2E B8
1019:C8 20 F6 C2 A0 C4 A9 D4 F0
1021:20 2E C8 20 F6 C2 A0 C4 C0
1029:A9 D9 20 2E C8 60 20 A2 67
1031:BB 20 DD BD 20 59 C8 A0 26
1039:00 B9 00 01 F0 03 C8 D0 CD
1041:F8 A2 0F 88 B9 00 01 9D 5E
1049:67 C8 CA 88 C0 FF D0 F4 CD
1051:A9 67 A0 C8 18 20 FB C1 BB
1059:60 A9 20 A0 00 A2 10 99 66
1061:67 C8 CA 88 C0 F9 60 20 7C
1069:20 20 20 20 20 20 20 89
1071:20 20 20 20 20 20 39 FF A3
1079:8A A2 00 9A 48 A9 13 8D CD
1081:6B C1 20 F9 C2 68 C9 0E C4
1089:D0 09 A9 92 A0 C4 18 20 1B
1091:FB C1 60 C9 0F D0 09 A9 A0
1099:A3 A0 C4 18 20 FB C1 60 A2
10A1:C9 14 D0 09 A9 AC A0 C4 5C
10A9:18 20 FB C1 60 A9 BD A0 3F
10B1:C4 18 20 FB C1 60 57 49 85
10B9:4C 44 20 43 41 52 44 20 45
10C1:4F 52 20 52 45 54 55 52 BF
10C9:4E FF 20 20 20 20 20 20 F8
10D1:20 20 20 20 20 20 20 F1
10D9:20 20 20 20 20 20 20 F9
10E1:20 20 20 20 FF 20 20 20 01
10E9:20 20 20 20 20 20 20 0A
10F1:20 20 20 20 20 20 20 12
10F9:20 20 20 20 20 20 20 1A
1101:20 20 20 20 20 20 FF 45 08
1109:58 49 54 00 4E 45 58 54 C0
1111:20 50 41 47 45 00 4E 45 FF
1119:57 20 53 45 41 52 43 48 CF
1121:FF 24 30 3A 2A 20 20 20 2F
1129:20 20 20 20 20 20 20 4B
1131:20 20 20 20 20 20 3A 2A F1
1139:FF 00 00 20 79 CD 90 01 82
1141:60 A9 16 A0 06 A2 04 20 AD
1149:84 C1 A9 B6 A0 C8 20 E9 11
1151:C1 A9 34 A0 C9 18 20 FB 3A
1159:C1 20 66 C2 A9 24 85 FB 42
1161:A9 C9 85 FC AD BD CE 48 95
1169:A9 00 8D BD CE A2 07 A0 9D
1171:0A A9 10 20 3A C3 D0 07 6D
1179:A9 2A 8D 24 C9 A9 01 18 FD
1181:69 03 A2 21 A0 C9 20 BD A9
1189:FF 68 8D BD CE A9 01 A2 15
1191:08 A0 00 8C 3A C9 20 BA 9C
1199:FF 20 C0 FF 20 6F CD 90 C6
11A1:05 A9 00 4C 2C CA 20 6C AE
11A9:C2 A2 01 20 C6 FF 20 CF 3E
11B1:FF C9 22 D0 F9 A0 00 99 83
11B9:CA C8 20 CF FF C8 C9 00 2B
11C1:D0 F5 20 CC FF A9 23 A0 28
11C9:02 A2 01 20 84 C1 A9 CA 01
11D1:A0 C8 20 E9 C1 A9 07 A0 7C
11D9:39 C8 20 FB C1 A9 11 8D 17
11E1:39 C9 A2 01 20 C6 FF A2 36
11E9:04 A8 20 CF FF CA D0 F9 FF
11F1:20 B3 CA 20 CF FF C9 00 7E
11F9:F0 06 99 E5 C8 C8 D0 F3 A6
1201:20 CC FF AD EA C8 C9 22 73
1209:D0 41 A9 E5 A0 C8 38 20 32
1211:FB C1 CE 39 C9 D0 CB 20 5A
1219:66 C2 20 0D C3 C9 02 D0 10
1221:0B AD 3A C9 D0 F4 20 6C 21
1229:C2 4C C5 C9 48 A9 01 20 22
1231:C3 FF 68 C9 04 D0 06 20 70
1239:6C C2 4C 41 C9 48 20 8A 1C
1241:CD 68 AE AC C0 D0 17 20 9E
1249:6C C2 60 EE 3A C9 A9 E5 81
1251:A0 C8 18 20 FB C1 A9 01 38
1259:20 C3 FF 4C 17 CA C9 00 BA
1261:D0 05 20 6C C2 38 60 4A FB
1269:18 69 04 AA A0 0C 20 F0 85
1271:FF A9 03 A2 03 A0 FF 20 45
1279:BA FF A9 00 20 BD FF 20 48
1281:C0 FF A2 03 20 C6 FF A2 49
1289:10 AC BD CE 20 CF FF 99 5F
1291:8F CE C9 22 F0 04 C8 CA 80
1299:D0 F2 8C BD CE A9 FF 99 07
12A1:8F C2 20 CC FF A9 03 20 DE
12A9:C3 FF 20 6C C2 A9 00 85 BC
12B1:D4 18 60 20 91 B3 20 DD CD
12B9:BD A0 00 B9 01 01 F0 06 74
12C1:99 E5 C8 C8 D0 F5 A9 20 A3
12C9:99 E5 C8 C8 60 4C 4F 41 ED
12D1:44 FF 4E 41 4D 45 00 30 A5
12D9:3A FF A9 01 2C A9 00 8D F5
12E1:0D CB A9 18 A0 06 A2 05 9D
12E9:20 84 C1 A9 CD A0 CA 20 B8
12F1:E9 C1 A9 D2 A0 CA 18 20 5E
12F9:FB C1 20 66 C2 A2 09 A0 4A
1301:09 A9 10 20 2A C3 F0 31 8D
1309:A9 01 A2 08 A0 00 20 BA 19
1311:FF AD BD CE A2 8F A0 CE AA
1319:20 BD FF A9 00 85 9D A6 51
1321:2B A4 2C 20 D5 FF 86 2D 76
1329:84 2E 20 60 CD 90 06 20 04
1331:6C C2 4C 44 A6 20 6C C2 5D
1339:60 20 9C CC 90 CA B0 F5 FE
1341:28 52 45 29 20 28 42 29 9A
1349:20 53 41 56 45 FF 4E 41 E9
1351:4D 45 00 30 3A FF 00 53 97
1359:54 41 52 54 3A 00 00 45 A0
1361:4E 44 3A FF A9 00 2C A9 57
1369:C0 2C A9 80 2C A9 00 85 C5
1371:02 A9 18 A0 06 A2 05 20 F4
1379:84 C1 A9 40 A0 CB 20 E9 E9
1381:C1 A9 4E A0 CB 18 20 FB C1
1389:C1 24 02 50 08 A9 56 A0 13
1391:CB 18 20 FB C1 20 66 C2 85
1399:A2 09 A0 09 A9 10 20 2A EF
13A1:C3 D0 06 24 02 30 66 10 8E
13A9:60 24 02 10 0B EE AC C0 78
13B1:A9 53 8D 8C CE 20 BE CE 3F
13B9:AD BD CE A2 8F A0 CE 20 E6
13C1:BD FF 24 02 50 1F A2 0B BA
13C9:A0 0E 20 14 CC B0 F7 48 6A
13D1:98 48 A2 0D A0 0C 20 14 05
13D9:CC B0 F7 84 FD 85 FE 68 46
13E1:85 FB 68 85 FC A9 01 A2 62
13E9:08 A0 00 84 9D 20 BA FF 67
13F1:24 02 50 09 A9 FB A6 FD CD
13F9:A4 FE 4C 03 CC A9 2B A6 F5
1401:2D A4 2E 20 D8 FF 20 60 18
1409:CD 20 6C C2 60 20 9C CC 63
1411:90 97 B0 F5 A9 12 85 FB 79
1419:A9 C6 85 FC A9 00 8D BD 6E
1421:CE A9 05 20 34 C3 A9 0D CE
1429:91 FB 98 AA A0 00 A9 2F 5E
1431:D1 FB B0 1D B1 FB C9 3A 74
1439:B0 17 C8 CA D0 F0 20 D9 A9
1441:C7 20 9B BC 18 A5 62 65 16
1449:63 38 D0 05 A5 64 A4 65 09
1451:18 60 53 43 52 41 54 43 BF
1459:48 FF 4E 41 4D 45 00 53 56
1461:30 3A FF A9 18 A0 06 A2 BC
1469:05 20 84 C1 A9 52 A0 CC 6D
1471:20 E9 C1 A9 5A A0 CC 18 FD
1479:20 FB C1 20 66 C2 A9 53 CF
1481:8D 8C CE A2 09 A0 0A A9 20
1489:10 20 2A C3 F0 07 20 BE E5
1491:CE 20 6C C2 60 20 9C CC 6C
1499:90 F4 B0 F5 EE AC C0 20 88
14A1:3B C9 08 A9 09 8D 6B C1 8C
14A9:20 F9 C2 A9 8C A0 CE 18 EF
14B1:20 FB C1 A9 00 8D AC C0 0C
14B9:28 60 43 4F 50 59 20 4F E2
14C1:52 20 52 45 4E 41 4D 45 11
14C9:FF 4E 45 57 20 4E 41 4D AD
14D1:45 00 00 4F 4C 44 20 4E 93
14D9:41 4D 45 00 FF A9 43 2C F7
14E1:A9 52 8D 8C CE A9 13 A0 D1
14E9:09 A2 05 20 84 C1 A9 BA 1B
14F1:A0 CC 20 E9 C1 A9 C9 A0 29
14F9:CC 18 20 FB C1 20 66 C2 70
1501:A2 09 A0 A0 A9 10 20 2A 6B
1509:C3 F0 19 AC BD CE A9 3D F8
1511:99 8F CE EE BD CE A2 0B 2E
1519:A0 0A A9 10 20 2A C3 F0 6E
1521:07 20 BE CE 20 6C C2 60 34
1529:AD BD CE 8D 44 CD EE AC 30
1531:C0 20 3B C9 B0 EE A9 0B 67
1539:8D 6B C1 20 F9 CE A2 98 FC
1541:A0 CE 18 69 00 90 01 C8 16
1549:18 20 FB C1 A9 00 8D AC 38
1551:C0 4C 21 CD A9 56 D0 02 3A
1559:A9 49 8D 8C CE 4C BE CE 19
1561:20 79 CD 20 90 CD 08 20 A1
1569:CE CD 20 8A CD 28 60 20 0B
1571:90 CD 90 04 20 CE CD 38 B9
1579:60 A9 0F A2 08 A8 20 BA 28
1581:FF A9 00 20 BD FF 20 C0 07
1589:FF 60 A9 0F 20 C3 FF 60 62
1591:A2 0F 20 C6 FF B0 27 A0 F2
1599:00 20 CF FF C9 D0 F0 06 30
15A1:99 15 CE C8 D0 F3 A9 FF ED
15A9:99 15 CE 20 CC FF A9 30 AB
15B1:CD 15 CE D0 07 CD 16 CE 59
15B9:D0 02 18 60 38 60 A0 00 5A
15C1:A2 0D B9 32 CE 99 15 CE B0
15C9:C8 CA D0 F6 F0 EE A9 1E 49
15D1:A0 05 A2 0C 20 84 C1 A9 E2
15D9:F4 A0 CD 20 E9 C1 A9 15 21
15E1:A0 CE 18 20 FB C1 20 66 A2
15E9:C2 A9 00 A0 FE 20 12 C3 4A
15F1:20 6C C2 60 44 49 53 4B DE
15F9:20 45 52 52 4F 52 20 44 3D
1601:49 53 50 4C 41 59 00 50 35
1609:52 45 53 53 20 41 4E 59 4B
1611:20 4B 45 59 FF 20 20 20 3F
1619:20 20 20 20 20 20 20 45
1621:20 20 20 20 20 20 20 4D
1629:20 20 20 20 20 20 20 55
1631:20 FF 44 45 56 49 43 45 ED
1639:20 45 52 52 4F 52 FF 4E 48
1641:45 57 20 41 20 44 49 53 F5
1649:4B 45 54 54 45 FF 4E 41 44
1651:4D 45 2C 49 44 00 4E 30 7E
1659:3A FF A9 18 A0 06 A2 05 C0
1661:20 84 C1 A9 3F A0 CE 20 CB
1669:E9 C1 A9 4E A0 CE 18 20 A5
1671:FB C1 20 66 C2 A2 09 A0 C9
1679:0A A9 13 20 2A C3 F0 08 C3
1681:A9 4E 8D 8C CE 20 BE CE D3
1689:20 6C C2 60 49 30 3A 20 DE
1691:20 20 20 20 20 20 20 BD
1699:20 20 20 20 20 20 20 C5
16A1:20 20 20 20 20 20 20 CD
16A9:20 20 20 20 20 20 20 D5
16B1:20 20 20 20 20 20 20 DD
16B9:20 20 20 20 20 00 20 79 BE
16C1:CD A2 0F 20 C9 FF A0 00 F0
16C9:AE BD CE E8 E8 E8 B9 8C 10
16D1:CE 20 D2 FF C8 CA D0 F6 D1
16D9:20 CC FF AD AC C0 F0 03 71
16E1:4C 8A CD 4C 63 00 00 00 70

Diskcheck: Apple Sector Editor For DOS 3.3

Steve Meyles

A disk editor is useful for learning about disk organization and many other tasks. This program, recommended for intermediate and advanced programmers, provides all the basic functions you need to examine and edit the contents of an Apple DOS 3.3 disk. A disk drive is required.

Have you ever needed to repair a garbled Apple II disk? Without a good disk-editor program, the task can become a nightmare. "Diskcheck" allows you to examine and modify the contents of any sector on an Apple II disk formatted with DOS 3.3. If you're just learning about how DOS 3.3 disks are organized, the program is an invaluable educational tool. Other important uses include repairing damaged disk data and concealing files from unauthorized users.

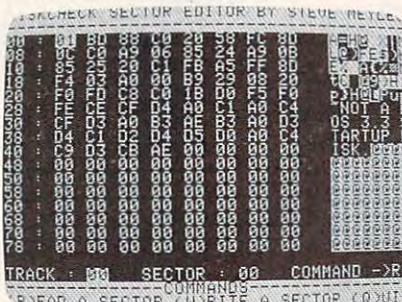
Since this program is written in machine language, you must type it in with "MLX," the machine language entry program published elsewhere in this issue. Here are the starting and ending addresses for MLX:

Starting address: 1000
Ending address: 197A

After you type in and save the program, run it with BRUN. The photograph illustrates the Diskcheck work screen. The central area of the screen displays the contents of one disk sector. The rest of the screen contains prompts and other information.

One-Key Operation

Diskcheck offers several different functions; each of them is invoked by pressing a single key. The simplest commands are Q, which exits



"Diskcheck" provides several different functions for examining and editing the contents of Apple II disks.

the program, and C, which clears the screen and displays a catalog of the current disk. Press Return when you have finished looking at the catalog display.

Each byte in the current sector is displayed as a two-digit hexadecimal value. The inverse (reverse) cursor marks your position in the byte display. Use the I, J, K, and M keys to move the cursor up, left, right, and down, respectively. The current track and sector are displayed at the bottom of the byte display area.

The Read command (R) lets you read and display the contents of any sector on the disk. After you press R, the inverse cursor moves to the track and sector area below the byte field. To choose a new track, enter a two-digit hexadecimal value and press Return. The program then moves the cursor to the sector display and waits for you to enter a new sector value. After you press Return, the new sector is read from disk and shown on the screen.

For instance, sector \$0F of track \$11 contains directory information. To view this, enter 11 for the track and 0F for the sector.

Diskcheck reads the first sector of the directory and displays it on the screen.

The plus (+) and minus (-) keys allow you to move forward and backward through the disk without having to enter new track and sector values for each sector. The plus key reads the next sector and the minus key backs up to read the previous sector.

The Write command (W) writes the current sector back to disk. This is normally done after you have made some changes to the sector's contents.

The Fill command (F) fills all or part of the sector with the specified byte value. This function operates from the current cursor position to the end of the sector, so you should position the cursor at the desired byte before selecting the function. After you press F, Diskcheck asks you to enter the byte value and press Return. The designated area is then filled with that value.

Editing

The Text (T) and Hex (H) functions are used to edit the current sector. In Hex mode, you can change a byte by entering a new two-digit hexadecimal value. Text mode is useful in cases where you need to enter a number of ASCII characters. When you press T, Diskcheck asks you to choose between flashing, inverse, and normal characters. After you respond to this prompt, you can enter new values simply by pressing one key for each byte. To exit editing mode, press Return.

If you're not familiar with Apple DOS 3.3, it's best to practice on an unimportant disk rather than one which contains important

information. A disk editor is a very powerful tool; in the worst case, you might scramble an entire disk by changing only a few bytes.

Note that this program is designed for use with normal DOS 3.3 disks. (Don't try to edit ProDOS disks with this program.) If a disk has been specially formatted with extra sectors, those sectors may not be accessible to Diskcheck. Similarly, you may get peculiar results if you try to edit a commercially copy-protected disk which contains deliberate errors or nonconventional file structures.

Diskcheck

Please refer to the "MLX" article in this issue before entering the following program.

START ADDRESS: 1000
END ADDRESS: 197A

```
1000: A9 00 8D 03 03 8D 04 03 30
1008: 8D 05 03 8D 06 03 8D 0A CA
1010: 03 8D 0B 03 20 2D 13 20 A2
1018: 3D 11 20 48 12 A9 1D 8D A2
1020: 08 03 A9 10 8D 09 03 A9 7B
1028: 13 20 5B FB A9 25 85 24 16
1030: 20 BE 18 20 38 15 C9 D2 91
1038: D0 09 20 72 16 20 2D 13 CC
1040: 4C 1A 10 C9 D7 D0 0C 20 E5
1048: DD 18 20 72 16 20 65 13 97
1050: 4C 1A 10 C9 C3 D0 06 20 49
1058: 88 13 4C 1A 10 C9 C9 D0 B8
1060: 06 20 8C 15 4C 1A 10 C9 23
1068: CD D0 06 20 AE 15 4C 1A E2
1070: 10 C9 CB D0 06 20 DE 15 15
1078: 4C 1A 10 C9 CA D0 06 20 A9
1080: FF 15 4C 1A 10 C9 AB D0 E0
1088: 11 20 41 19 20 22 16 A9 52
1090: 00 8D 05 03 8D 06 03 4C BB
1098: 1A 10 C9 AD D0 11 20 41 2A
10A0: 19 20 4C 16 A9 00 8D 05 AD
10A8: 03 8D 06 03 4C 1A 10 C9 53
10B0: CB D0 06 20 B8 16 4C 1A FC
10B8: 10 C9 DA D0 06 20 66 17 8F
10C0: 4C 1D 10 C9 C6 D0 06 20 92
10C8: 19 19 4C 1A 10 C9 D1 F0 23
10D0: 03 4C 1A 10 A9 00 85 22 44
10D8: A9 18 85 23 20 5B FC 20 33
10E0: 5C 15 C7 CF CF C4 C2 D9 5B
10E8: C5 A0 AD A0 D9 CF D5 A0 2E
10F0: C3 C1 CE A0 D2 C5 C5 CE 4F
10F8: D4 C5 D2 A0 D4 C8 D2 CF 98
1100: D5 C7 C8 A0 C1 8D B1 B0 7A
1108: B0 B0 C7 A0 C6 D2 CF CD A0
1110: A0 D4 C8 C5 A0 CD CF CE D7
1118: C9 D4 CF D2 A0 CF D2 A0 06
1120: C1 A0 C3 C1 CC CC A0 B4 6F
1128: B0 B9 B6 8D C6 D2 CF CD AF
1130: A0 C2 C1 D3 C9 C3 8D 8D CE
1138: 8D 00 4C D0 03 20 5B FC FD
1140: 20 5C 15 04 09 13 0B 03 1A
1148: 08 05 03 0B 20 13 05 03 1B
1150: 14 0F 12 20 02 19 20 13 14 2E
1158: 0F 12 20 02 19 20 13 14 2E
1160: 05 16 05 20 0D 05 19 0C E7
1168: 05 13 8D 20 2D 20 2D 2D FB
1170: 2D 2D 2D 2D 2D 2D 2D 2D 92
1178: 2D 2D 2D 2D 2D 2D 2D 2D 9A
1180: 2D 2D 2D 2D 2D 2D 2D 2D 92
1188: 2D 2D 2D 2D 2D 2D 2D 2D AA
1190: 2D 2D 8D 00 A9 14 20 5B 7F
1198: FB 20 5C 15 20 2D 2D 2D 43
11A0: 2D 2D 2D 2D 2D 2D 2D 2D C2
```

```
11A8: 2D 2D 2D 03 0F 0D 0D 01 4A
11B0: 0E 04 13 2D 2D 2D 2D 2D B5
11B8: 2D 2D 2D 2D 2D 2D 2D 2D DA
11C0: 2D 2D 2D 8D 28 12 29 05 24
11C8: 01 04 20 01 20 13 05 03 DA
11D0: 14 0F 12 20 28 17 29 12 07
11D8: 09 14 05 20 01 20 13 05 DA
11E0: 03 14 0F 12 20 28 11 29 79
11E8: 15 09 14 8D 20 20 20 20 15
11F0: 28 14 29 05 18 14 20 20 13
11F8: 28 08 29 05 18 20 20 28 50
1200: 2B 29 20 20 28 2D 29 20 72
1208: 20 28 06 29 09 0C 0C 20 4A
1210: 20 20 20 8D 20 20 20 20 0B
1218: 20 28 03 29 01 14 01 0C AF
1220: 0F 07 20 20 20 28 09 2C 73
1228: 0A 2C 0B 2C 0D 29 0D 0F B6
1230: 16 05 0D 05 0E 14 20 20 B3
1238: 20 20 20 00 A9 02 85 22 FA
1240: A9 14 85 23 20 5B FC 60 DD
1248: AD 55 C0 20 5B FC A9 00 BC
1250: 8D 01 03 AD 06 03 8D 02 10
1258: 03 AE 02 03 20 44 F9 20 40
1260: 5C 15 A0 BA A0 00 A0 00 FD
1268: 98 18 6D 02 03 AA BD 00 EA
1270: 1F AA 20 44 F9 A9 A0 20 EE
1278: F0 FD C8 C0 08 D0 E9 20 31
1280: 5C 15 A0 A0 00 A0 00 98 51
1288: 18 6D 02 03 AA BD 00 1F EF
1290: C9 80 90 06 C9 A0 B0 02 60
1298: 29 7F 20 F0 FD C8 C0 08 E0
12A0: D0 E5 A9 8D 20 ED FD 98 02
12A8: 18 6D 02 03 8D 02 03 EE 0E
12B0: 01 03 AD 01 03 C9 10 D0 0C
12B8: A0 AD 02 03 8D 07 03 20 B7
12C0: 5C 15 8D D4 D2 C1 C3 CB 48
12C8: A0 BA A0 00 AE 03 03 20 A7
12D0: 44 F9 20 5C 15 A0 A0 A0 6C
12D8: D3 C5 C3 D4 CF D2 A0 BA E3
12E0: A0 00 AE 04 03 20 44 F9 86
12E8: 20 5C 15 A0 A0 A0 C3 CF BF
12F0: CD C1 CE C4 A0 AD BE 57
12F8: 00 20 03 13 20 4E 19 AD D0
1300: 54 C0 60 AD 05 03 38 ED F9
1308: 06 03 8D 02 03 4A 4A E3
1310: 18 69 02 20 5B FB AD 02 07
1318: 03 29 07 BD 02 03 A0 6D 61
1320: 02 03 69 05 85 24 AC 05 A0
1328: 03 B9 00 1F 60 AD 03 03 F2
1330: 8D 54 13 AD 04 03 8D 55 0C
1338: 13 A9 01 8D 5C 13 A9 13 E0
1340: A0 50 20 D9 03 AD 5D 13 09
1348: 20 4E 14 A9 00 85 48 60 36
1350: 01 60 01 00 00 00 61 13 05
1358: 00 1F 00 00 01 00 00 60 AE
1360: 01 00 01 EF D8 AD 03 03 AC
1368: 8D 54 13 AD 04 03 8D 55 44
1370: 13 A9 02 8D 5C 13 A9 13 39
1378: A0 50 20 D9 03 AD 5D 13 41
1380: 20 4E 14 A9 00 85 48 60 36
1388: 20 DC 03 85 FD 84 FC 20 CA
1390: 5B FC A0 00 B9 A8 13 91 5E
1398: FC C8 C0 12 D0 F6 20 D6 22
13A0: 03 20 15 14 20 38 15 60 A0
13A8: 06 00 00 00 00 01 06 00 E1
13B0: 00 00 00 00 00 93 00 00 25
13B8: 00 00 85 FE 20 4E 19 A9 95
13C0: 88 20 ED FD 20 ED FD 2C B1
13C8: 00 C0 10 FB AD 00 C0 2C FB
13D0: 10 C0 C9 8D D0 0F A6 FE 50
13D8: 20 44 F9 A9 A0 20 F0 FD 5F
13E0: A5 FE A2 00 60 C9 A0 D0 2A
13E8: 0F A6 FE 20 44 F9 A9 A0 20
13F0: 20 F0 FD A5 FE A2 01 60 62
13F8: 49 B0 C9 0A 90 08 69 88 C9
1400: C9 FA 90 C3 29 0F 48 A5 D5
1408: FE 0A 0A 0A 0A 85 FE 68 E0
1410: 05 FE 4C BA 13 A0 0A B1 90
1418: FC C9 00 D0 01 60 20 5B 60
1420: FC 20 5C 15 87 49 6F 4F 3B
1428: 60 45 52 52 4F 52 8D 8D AD
1430: CB C9 D4 A0 C1 CE D9 A0 71
1438: CB C5 D9 A0 D4 CF A0 C3 E7
1440: CF CE D4 C9 CE D5 C5 00 94
```

```
1448: 20 38 15 68 68 60 B0 01 DE
1450: 60 8D 00 03 AD 0A 03 8D 65
1458: 03 03 AD 0B 03 8D 04 03 82
1460: A9 87 20 ED FD 20 5B FC 40
1468: AD 00 03 C9 10 D0 39 20 BA
1470: 5C 15 44 49 53 4B 60 49 FA
1478: 53 60 57 52 49 54 45 60 F8
1480: 50 52 4F 54 45 43 54 45 B9
1488: 44 61 8D 8D C8 C9 D4 A0 6D
1490: C1 CE D9 A0 CB C5 D9 A0 5C
1498: D4 CF A0 C3 CF CE D4 C9 9C
14A0: CE D5 C5 00 20 38 15 60 CA
14A8: C9 40 D0 60 20 5C 15 44 C6
14B0: 52 49 56 45 60 45 52 52 82
14B8: 4F 52 6C 60 43 41 4E 67 D3
14C0: 54 60 52 45 41 44 6F 57 1B
14C8: 52 49 54 45 8D A0 A0 01 C
14D0: A0 A0 A0 A0 A0 A0 A0 F8
14D8: A0 A0 A0 A0 A0 A0 A0 01
14E0: A0 A0 54 52 41 43 4B 6F 4E
14E8: 53 45 43 54 4F 52 8D 8D 26
14F0: C8 C9 D4 A0 C1 CE D9 A0 32
14F8: CB C5 D9 A0 D4 CF A0 C3 A8
1500: CF CE D4 C9 CE D5 C5 00 56
1508: 20 38 15 60 20 5C 15 42 D7
1510: 41 44 60 53 45 43 54 4F 5C
1518: 52 61 8D 8D C8 C9 D4 A0 06
1520: C1 CE D9 A0 CB C5 D9 A0 ED
1528: D4 CF A0 C3 CF CE D4 C9 2E
1530: CE D5 C5 00 20 38 15 60 5C
1538: 20 5C 15 60 88 00 2C 00 CE
1540: C0 10 FB 20 5C 15 A0 88 51
1548: 00 AD 00 C0 2C 10 C0 C9 D6
1550: 80 90 05 C9 A0 B0 01 60 46
1558: 20 F0 FD 60 85 45 84 47 26
1560: 86 46 68 95 FA 68 85 FB 45
1568: E6 FA D0 02 E6 FB A0 00 67
1570: B1 FA F0 0B 20 ED FD E6 9C
1578: FA D0 F5 E6 FB D0 F1 A5 2E
1580: FB 48 A5 FA 48 A5 45 A6 29
1588: 46 A4 47 60 AD 05 03 38 AD
1590: E9 08 B0 05 68 68 6C 08 DD
1598: 03 CD 06 03 90 03 4C 9F 71
15A0: 16 8D 05 03 AD 06 03 38 CD
15A8: E9 08 8D 06 03 60 AD 05 D5
15B0: 03 18 69 08 90 05 68 68 E1
15B8: 6C 08 03 38 E9 01 CE 07 F6
15C0: 03 CD 07 03 08 EE 07 03 FD
15C8: 18 69 01 28 90 0D 8D 05 D4
15D0: 03 AD 06 03 18 69 08 8D DC
15D8: 06 03 60 4C 9F 16 AC 05 4E
15E0: 03 C8 D0 05 68 68 6C 08 AB
15E8: 03 CC 07 03 F0 04 98 4C ED
15F0: 9F 16 8C 05 03 AD 06 03 30
15F8: 18 69 08 8D 06 03 60 AC D0
1600: 05 03 88 C0 FF D0 05 68 42
1608: 68 6C 08 03 CC 06 03 90 C7
1610: 04 98 4C 9F 16 8C 05 03 D9
1618: AD 06 03 38 E9 08 8D 06 11
1620: 03 60 AD 04 03 C9 0F F0 2A
1628: 07 EE 04 03 20 D0 13 60 80
1630: A9 00 8D 04 03 AD 03 03 FA
1638: C9 22 F0 07 EE 03 03 20 0A
1640: 2D 13 60 A9 00 8D 03 03 AD
1648: 20 2D 13 60 AD 04 03 F0 AC
1650: 07 CE 04 03 20 2D 13 60 A0
1658: A9 0F 8D 04 03 AD 03 03 E6
1660: F0 07 CE 03 03 20 2D 13 D6
1668: 60 A9 22 8D 03 03 20 2D DD
1670: 13 60 20 41 19 A9 13 20 0C
1678: 5B FB A9 08 85 24 AD 03 22
1680: 03 20 BA 13 C9 23 B0 EA E5
1688: 8D 03 03 A9 16 85 24 AD F3
1690: 04 03 20 BA 13 29 0F C9 54
1698: 10 B0 F0 8D 04 03 60 85 62
16A0: FA 20 03 13 A0 24 F9 3C
16A8: AA FA 8C 05 03 20 43 13 79
16B0: 20 4E 19 68 68 6C 08 03 32
16B8: A9 01 20 5B FB A9 00 85 BF
16C0: 24 20 5C 15 6D 60 48 45 DA
16C8: 58 6D 53 50 41 43 45 60 ED
16D0: 42 45 54 57 45 45 45 4E 87
16D8: 60 45 41 43 48 7B 60 43 17
16E0: 6F 52 60 54 4F 60 45 4E 7F
```

```

16E8: 44 6D 6D 00 20 03 13 20 93
16F0: BE 18 AC 05 03 B9 00 1F 86
16F8: 20 BA 13 AC 05 03 99 00 78
1700: 1F C9 80 90 06 C9 A0 B0 92
1708: 02 29 7F 48 AD 05 03 29 A6
1710: 07 18 69 1F 85 24 68 20 94
1718: F0 FD E0 00 F0 13 A9 EC 6E
1720: 8D 08 03 A9 16 8D 09 03 0E
1728: 20 DE 15 20 48 12 4C EC D2
1730: 16 A9 01 20 5B FB A9 00 14
1738: 85 24 20 5C 15 2D 2D 2D E0
1740: 2D 2D 2D 2D 2D 2D 2D 2D 6E
1748: 2D 2D 2D 2D 2D 2D 2D 2D 76
1750: 2D 2D 2D 2D 2D 2D 2D 2D 7E
1758: 2D 2D 2D 2D 2D 2D 2D 2D 86
1760: 2D 2D 2D 2D 00 60 A9 12 CF
1768: 20 5B FB A9 00 85 24 20 16
1770: 5C 15 68 46 69 4C 41 53 D5
1778: 48 49 4E 47 60 28 09 29 3A
1780: 0E 16 05 12 13 05 20 A8 92
1788: CE A9 CF D2 CD C1 CC A0 5F
1790: A8 C6 AF C9 AF CE A9 00 63
1798: 20 38 15 C9 C6 D0 08 A9 57
17A0: 7F 8D 0C 03 4C D2 17 C9 49
17A8: C9 D0 08 A9 3F 8D 0C 03 D6
17B0: 4C D2 17 C9 CE D0 08 A9 AC
17B8: FF 8D 0C 03 4C D2 17 C9 A1
17C0: 80 90 07 C9 A0 B0 03 4C EA
17C8: 98 17 A9 88 20 ED FD 4C C7
17D0: 98 17 A9 12 20 5B FB A9 77
17D8: 00 85 24 20 5C 15 A0 A0 08
17E0: A0 A0 A0 A0 A0 A0 A0 A0 0F
17E8: A0 A0 A0 A0 A0 A0 A0 A0 17
17F0: A0 A0 A0 A0 A0 A0 A0 A0 1F
17F8: A0 A0 A0 A0 A0 A0 A0 A0 27
1800: A0 A0 A0 A0 00 A9 01 20 8F
1808: 5B FB A9 00 85 24 20 5C 73
1810: 15 6D 6D 6D 54 45 58 54 67
1818: 6D 54 59 50 45 60 49 4E D0
1820: 60 44 41 54 41 60 57 49 82
1828: 54 48 60 43 6F 52 60 54 AE
1830: 4F 60 45 4E 44 6D 6D 6D CD
1838: 00 A9 13 20 5B FB A9 25 7A
1840: 85 24 20 BE 18 20 38 15 F2
1848: C9 8D F0 3D 2D 0C 03 AC FE
1850: 05 03 99 00 1F 48 20 03 54
1858: 13 20 4E 19 68 C9 80 90 71
1860: 06 C9 A0 B0 02 29 7F 48 21
1868: AD 05 03 29 07 18 69 1F 2E
1870: 85 24 68 20 F0 FD A9 39 87
1878: 8D 08 03 A9 18 8D 09 03 78
1880: 20 DE 15 20 48 12 4C 39 79
1888: 18 A9 01 20 5B FB A9 00 6F
1890: 85 24 20 5C 15 2D 2D 2D 3B
1898: 2D 2D 2D 2D 2D 2D 2D 2D C8
18A0: 2D 2D 2D 2D 2D 2D 2D 2D D0
18A8: 2D 2D 2D 2D 2D 2D 2D 2D D8
18B0: 2D 2D 2D 2D 2D 2D 2D 2D E0
18B8: 2D 2D 2D 2D 00 60 A2 04 0E
18C0: 8E CC 18 A9 08 8D CF 18 37
18C8: A0 00 B9 00 04 99 00 08 0F
18D0: C8 D0 F7 EE CC 18 EE CF FB
18D8: 18 CA D0 EE 60 A9 12 20 BE
18E0: 5B FB A9 00 85 24 20 5C 4C
18E8: 15 57 52 49 54 45 60 68 39
18F0: 59 6F 4E 69 00 20 38 15 10
18F8: C9 CE D0 05 68 68 4C 1A C3
1900: 10 C9 D9 F0 13 C9 80 90 48
1908: 07 C9 A0 B0 03 4C F5 18 9C
1910: A9 88 20 ED FD 4C F5 18 41
1918: 60 A9 12 20 5B FB A9 00 47
1920: 85 24 20 5C 15 C8 C5 D8 18
1928: A0 C2 D9 D4 C5 A0 AD AD 9D
1930: BE 00 A9 00 20 BA 13 AC B5
1938: 05 03 99 00 1F C8 D0 FA 99
1940: 60 AD 03 03 8D 0A 03 AD E6
1948: 04 03 8D 08 03 60 85 FE 43
1950: 4A 4A 4A 4A C9 0A B0 08 08
1958: 69 30 20 F0 FD 4C 65 19 63
1960: E9 09 20 F0 FD A5 FE 29 8A
1968: 0F C9 0A B0 08 69 30 20 47
1970: F0 FD 4C 7A 19 E9 09 20 6E
1978: F0 FD 60 D8 D4 A0 A0 A0 47

```

128 Editing Functions For Commodore 64

Jim Allen

This powerful programming aid makes all of the important Commodore 128 screen-editing functions available on the Commodore 64.

If you own a Commodore 64, you may wish that you had the extra screen-editing functions available on the Commodore 128. With just two quick keystrokes, you can turn auto-insert mode on or off, clear selected portions of the screen, set and manipulate screen windows, move the cursor instantly to any location, and more. In the 128, these functions are called ESC (escape) functions because they are activated with the ESC key. "128 Screen Editor" adds 14 ESC functions to the 64, plus a NO SCROLL key to prevent screen scrolling, and three new functions that aren't even available on the 128. It also allows all keys to repeat.

Type in and save the program with the "MLX" machine language entry program found elsewhere in this issue. Be sure to read and understand the instructions for using

MLX before you begin entering the data for 128 Screen Editor. When you run MLX, you'll be asked for a starting address and an ending address for the data you'll be entering. Use the following values:

Starting address: C000
Ending address: C3D7

After you've finished entering all the data for 128 Screen Editor, be sure to save a copy to disk or tape before you leave MLX.

This program loads with the command LOAD "filename",8,1 (replace *filename* with the name you used when entering the program with MLX, and replace the ,8,1 with ,1,1 if you are using tape instead of disk). After the program has loaded into memory, type these commands in direct mode (without line numbers):

NEW
SYS 49152

Don't forget to press RETURN at the end of each line. After you've entered the second command, the program sets the screen border to the same color as the background as a signal that it is active.

New ESCape Functions

Since the 64 lacks an ESC key, this program uses the back-arrow key (←) in its place. In the remainder of this article, the term ESC refers to the back-arrow key at the upper left corner of the keyboard. ESC functions require two keypresses: First you press and release the ESC key; then you press a second key. For instance, to delete a line, you press ESC and then D.

If you change your mind after pressing ESC and decide *not* to perform an ESC function, simply press a key that has no special ESC function (a CRSR key, for instance). If you need to type the back-arrow character itself, press ESC twice in succession. A description of the ESC functions follows.

Erase/Delete Functions

ESC @ Clears the screen from the current cursor position to the bottom.

ESC D Deletes the line the cursor is on, scrolls the screen up to fill that line, and places the cursor on the left screen margin.

ESC P Erases from the start of the current line up to and including the current cursor position.

ESC Q Erases from the current cursor position to the end of the current line.

f1 Erases from the line the cursor is on, including the current line, to the top of the screen or window.

f3 Deletes the character under the cursor and moves the remainder of the line one space to the left.

Insert Functions

ESC A Turns on autoinsert mode, which allows you to insert printable characters without using the INST key. The cursor and other editing keys work as usual. RETURN cancels autoinsert mode (this is not true on the 128).

ESC C Cancels autoinsert mode. This works the same as ESC C on the 128. (Note, however, that some of Commodore's documenta-

tion for the 128 confuses ESC C with the ESC O function.)

ESC O Cancels insert and quote modes, allowing you to use cursor keys and other editing keys after pressing INST or typing a quotation mark.

ESC I Inserts a blank line at the current cursor position, scrolling the remainder of the screen down and placing the cursor on the left edge of the display.

Cursor Movement Functions

ESC J Moves the cursor to the beginning of the line.

ESC K Moves the cursor to the end of the text on the line.

f7 Moves the cursor to the lower left corner of the screen.

Miscellaneous Functions

ESC T Sets the top of the window. Blocks the top portion of the screen from being erased or scrolled. The cursor position determines the top row of the new window. To reset the window to the full screen size, press the HOME key twice.

ESC V Scrolls the contents of the screen or window up one line. A new blank line will be scrolled in at the bottom, and the previous contents of the top line will be lost.

ESC W Scrolls the contents of the screen or window down one line. A new blank line will be scrolled in at the top, and the previous contents of the bottom line will be lost.

SHIFT Enables the NO SCROLL feature. The NO SCROLL key on the 128 lets you pause printing to the screen display (for instance, when you are listing a program). To pause a scrolling display, press SHIFT or SHIFT LOCK. When you release the key, printing resumes.

ESC Z Disables all 128 Screen Editor functions. You can reenables the ESC functions

at any time with SYS 49152. (On the 128, ESC Z clears all TAB stops, a function not available in this program.)

You should be aware that these functions affect logical lines, not physical screen lines. On the Commodore 64, a physical line is always 40 characters long, but a logical line can overlap two physical screen lines. Thus, for example, the ESC V function may scroll the screen upward two lines if the topmost logical line in the current window is more than 40 characters long.

The window function (ESC T) does not prevent you from moving the cursor above the window boundary with the cursor keys. If you venture above the boundary, strange results will occur. (If this happens accidentally, perform a warm start by pressing RUN/STOP-RESTORE; then restart the program with SYS 49152.)

The delete-line function (ESC D) is intended for deleting a line which lies between two other lines. If you simply want to erase a line (particularly a line on the bottom screen line), use the ESC Q function.

This program works by copying BASIC and the Kernal ROM into underlying RAM and modifying them. It also modifies two important vectors: IQPLOT at \$306 and IBASIN at \$324. Locations \$02, \$B6, and \$334-\$338 are also used for various purposes.

Quick Reference

Function	Keys
Erase from cursor to end of screen	ESC @
Delete current line	ESC D
Erase from start of line to cursor	ESC P
Erase from cursor to end of line	ESC Q
Erase from cursor to top of window	† f1
Delete to right of cursor	† f3
Enable auto-insert mode	ESC A
Cancel auto-insert mode	ESC C
Cancel insert and quote modes	ESC O
Insert a line	ESC I
Move cursor to start of line	ESC J
Move cursor to end of line	ESC K
Move cursor to lower left corner	† f7
Set top of window	ESC T
Scroll screen/window up	ESC V
Scroll screen/window down	ESC W
Pause scrolling	† SHIFT
Disable Editor	† ESC Z
Enable Editor	† SYS 49152

† different from Commodore 128

LOTTO CIPHER..

GET THE BEST ODDS ON ANY LOTTERY
SIX NUMBER - PICK FOUR - DAILY GAME

- PRODUCES FOUR COMBINATIONS OF NUMBERS TO CHOOSE FROM.
- ANY AMOUNT OF BALLS AND NUMBERS CAN BE PROGRAMMED.
- PRINTS OUT PAST LOTTO NUMBERS DRAWN, PAST COMPUTER PICKS, AND NUMBER DRAW FREQUENCY LIST.
- RANDOM NUMBER GENERATOR INCLUDED.



\$29.95 C-84/128

Window Magic

SUPER HI-RESOLUTION DRAWING IN MULTI OR MONO COLOR

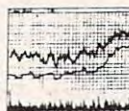
- COPY
- FILL
- LINES
- DRAW
- COLOR SQUARES
- TYPES LETTERS AND GRAPHICS
- POLYGON SHAPES-EXPAND, SHRINK AND ROTATE, THEN STAMP ANYWHERE
- ZOOM-FIT-DRAW ON AN EXPANDED WINDOW AND YOUR DRAWING AT THE SAME TIME
- MIRROR, FLIP, AND SCROLLING WINDOWS
- ZOOM-EXPANDS A WINDOW TO DOUBLE SIZE
- SAVE AND LOAD YOUR WINDOWS ON DISK
- PRINTS ON STANDARD DOT MATRIX PRINTER
- CLONE COLOR ATTRIBUTES

\$24.95 C-84/128

STOCK BROKER

PROFITS GUARANTEED OR
YOUR MONEY BACK

- BUYING GOOD QUALITY, VOLATILE ISSUES AND USING THIS TRADING SYSTEM WILL HAVE YOU FULLY INVESTED AT THE LOWEST PRICES AND CONVERTING TO CASH AS THE STOCK NEARS ITS PEAK.
- TECHNICAL TRADING THAT WORKS.
- BAR GRAPH PRINT-OUTS.
- RECORD UP TO 144 STOCKS ON A DISK.



\$29.95 C-84/128

ACORN OF INDIANA, INC.

2721 OHIO STREET
MICHIGAN CITY, IN 46360



219-879-2284

SHIPPING AND HANDLING, ADD \$1.50 - C.O.D.'S ACCEPTED
VISA AND MASTER CARD ORDERS ADD 4%
INDIANA RESIDENTS ADD 5% SALES TAX

AN OPPORTUNITY UNLIKE ANY OTHER



The nation's largest computer camp offers all traditional camping activities and camaraderie in a beautiful setting. Your child will have exclusive use of a major brand microcomputer for at least 6 hours every day.

An experienced staff ratio of one to three and a computer ratio of one to one uncaps the creativity of young people. Students receive hands-on experience in robotics, graphics, lasers, sound, languages, telecommunications and more.

Complete details on the 1, 2, 4, and 8 week coed sessions for ages 8-18 are available in a free brochure. Call or write for yours!

Call (317) 297-2700 or write to
MIDWEST COMPUTER CAMP
9392 Lafayette Road, Unit C4, Indianapolis, IN 46278

To receive additional
information from advertisers
in this issue, use the
handy reader service cards
in the back of the
magazine.

128 Screen Editor For Commodore 64

Please refer to the "MLX" article in this issue
before entering the following program.

```

C000:20 90 C3 78 AD 24 03 8D 47
C008:34 03 A9 7B 8D 24 03 AD 02
C010:25 03 8D 35 03 A9 C0 8D B7
C018:25 03 AD 06 03 C9 49 D0 A5
C020:0A A9 C3 CD 07 03 F0 16 A2
C028:AD 06 03 8D 37 03 A9 49 9D
C030:8D 06 03 AD 07 03 8D 38 CC
C038:03 A9 C3 8D 07 03 A2 A0 21
C040:A0 00 20 AD C3 A2 E0 20 7B
C048:AD C3 A2 FF 86 B6 84 02 FF
C050:A9 A6 8D F6 E8 8D 16 E9 05
C058:A9 B6 8D F7 E8 A9 02 8D 0D
C060:17 E9 A9 FD 25 01 85 01 36
C068:58 AD 8A 02 8D 36 03 A9 E7
C070:80 8D 8A 02 4C 80 A4 00 B4
C078:00 00 00 A5 99 D0 13 A5 30
C080:D3 85 CA A5 D6 85 C9 98 F9
C088:48 8A 48 A5 D0 F0 09 4C DC
C090:3A E6 4C 57 F1 20 16 E7 0C
C098:20 1A C3 C9 5F D0 0C EE 0B
C0A0:78 C0 20 A3 C1 C9 0D F0 0D
C0A8:4B D0 ED C9 13 D0 18 AE 19
C0B0:78 C0 10 05 A2 00 20 5B A1
C0B8:C1 A2 FF 8E 78 C0 A6 02 C2
C0C0:F0 D3 20 14 C3 10 D1 EE E5
C0C8:78 C0 C9 93 D0 0B A6 02 2B
C0D0:F0 C3 86 19 A2 18 4C FB 28
C0D8:C0 C9 88 D0 03 4C 37 C3 C6
C0E0:C9 85 D0 03 4C F7 C0 C9 80
C0E8:86 D0 05 20 3F C3 10 A8 56
C0F0:C9 0D D0 A1 4C EA E5 A6 4F
C0F8:D6 86 19 20 FF E9 CA 30 1A
C100:04 E4 02 B0 F6 A6 19 20 AE
C108:14 C3 4C 98 C0 20 76 C3 D0
C110:A6 02 20 F0 E9 A0 27 B5 50
C118:DA 30 02 A0 4F 8C 7A C0 C1
C120:B1 D1 99 00 01 88 10 F8 67
C128:86 D6 20 65 E9 A6 02 20 0D
C130:FF E9 E8 20 F0 E9 AC 7A 50
C138:C0 B9 00 01 91 D1 88 10 8F
C140:F8 A6 02 B5 DA 09 80 95 17
C148:DA AC 7A C0 C0 27 F0 04 48
C150:A9 7F 35 DB 95 DB 4C 88 2A
C158:C3 A6 D6 86 02 CA 86 B6 A9
C160:60 A4 D3 84 19 A0 00 84 CF
C168:D3 88 C8 A9 20 20 D2 FF D2
C170:C4 19 90 F6 C6 D3 60 20 84
C178:76 C3 20 EA E8 4C 88 C3 28
C180:F0 8B A0 00 84 D3 60 20 C7
C188:C0 C3 B1 D1 C9 20 D0 05 26
C190:88 10 F7 F0 01 C8 84 D3 72
C198:60 20 C0 C3 20 3F C3 88 B6
C1A0:D0 FA 60 20 1A C3 C9 54 21
C1A8:F0 AF C9 50 F0 B3 C9 40 F8
C1B0:F0 32 C9 57 F0 CA C9 56 84
C1B8:F0 BD C9 44 F0 32 C9 41 C6
C1C0:F0 76 C9 4F F0 6E C9 49 A6
C1C8:F0 6C C9 4A F0 B4 C9 4B F6
C1D0:F0 B5 C9 51 F0 C3 C9 5A 0D
C1D8:D0 03 4C B6 C2 A8 68 68 6C
C1E0:98 4C 95 C0 20 99 C1 20 8D
C1E8:76 C3 20 E7 C2 4C 88 C3 37
C1F0:A5 B6 85 19 A5 02 85 1A 91
C1F8:A6 D6 8E 77 C0 B5 DA 10 71
C200:0D B5 D9 30 15 09 80 95 1B
C208:D9 CA 8E 77 C0 CA 09 80 3A
C210:95 DA 86 B6 E8 86 02 20 D8
C218:EA E8 AE 77 C0 CA 86 B6 8F
C220:E8 86 02 20 EA E8 A6 19 5F
C228:86 B6 A6 1A 86 02 AE 77 26
C230:C0 4C 14 C3 F0 27 F0 3E 2C
C238:20 1A C3 C9 14 F0 73 C9 7E
C240:86 D0 05 20 3F C3 10 F0 F9
C248:C9 11 F0 0C C9 91 F0 08 54
C250:C9 1D F0 04 C9 9D D0 0E D4
C258:20 D2 FF D0 DB A2 00 86 9F
C260:D8 86 D4 86 C7 60 C9 5F A9
    
```

```

C268:D0 27 20 1A C3 C9 43 F0 82
C270:EC 20 CA C1 10 C2 20 82 38
C278:C1 A9 9D 20 D2 FF A9 94 7D
C280:20 D2 FF A9 20 D2 FF 8C
C288:A6 D6 B5 D9 09 80 95 D9 BA
C290:60 C9 0D D0 03 4C 5D C2 2E
C298:48 A5 D4 8D 77 C0 A9 00 31
C2A0:85 D4 A9 94 20 D2 FF AD 96
C2A8:77 C0 85 D4 68 20 D2 FF 81
C2B0:D0 86 A9 14 D0 F7 78 AD BB
C2B8:34 03 8D 24 03 AD 35 03 49
C2C0:8D 25 03 AD 37 03 8D 06 78
C2C8:03 AD 38 03 8D 07 03 A9 AA
C2D0:02 05 01 85 01 58 AD 36 0C
C2D8:03 8D 8A 02 AD 79 C0 30 B9
C2E0:03 8D 20 D0 4C 80 A4 AE B8
C2E8:77 C0 B5 DA 8D 7A C0 30 C6
C2F0:03 EE 77 C0 EE 77 C0 A2 28
C2F8:18 20 FF E9 B5 D9 09 80 D8
C300:95 D9 CA EC 77 C0 B0 F1 03
C308:CE 77 C0 2C 7A C0 30 03 E9
C310:CE 77 C0 60 A0 00 18 4C 7C
C318:F0 FF A5 C6 85 CC 8D 92 46
C320:02 F0 F7 78 A5 CF F0 0C C5
C328:A5 CE AE 87 02 A0 00 84 9B
C330:CF 20 13 EA 4C B4 E5 A2 5C
C338:18 20 14 C3 4C 98 C0 A9 82
C340:20 20 D2 FF A9 14 4C D2 43
C348:FF 48 98 48 20 87 EA 20 8E
C350:EF 4F C9 83 D0 0F 20 F3 B2
C358:F6 68 A8 68 20 77 C1 20 93
C360:82 C1 4C 80 A4 AD 8D 02 24
C368:C9 01 F0 E0 A9 00 85 C6 60
C370:68 A8 68 6C 37 03 38 20 80
C378:F0 FF 8E 77 C0 8C 78 C0 AB
C380:B5 D9 30 03 CE 77 C0 60 C5
C388:AE 77 C0 AC 78 C0 10 86 95
C390:AD 20 D0 29 0F 8D 79 C0 06
C398:AD 21 D0 29 0F CD 79 C0 4F
C3A0:F0 04 8D 20 D0 60 AD 86 3F
C3A8:02 8D 20 D0 60 86 1A 84 7B
C3B0:19 A2 20 B1 19 91 19 C8 96
C3B8:D0 F9 E6 1A CA D0 F4 60 89
C3C0:A0 27 A6 D6 B5 DA 10 07 E4
C3C8:C4 D3 F0 05 90 01 60 A0 FF
C3D0:4F 60 00 00 00 00 00 18
    
```

©

All the programs in
this issue are
available on the
ready-to-load
COMPUTE! Disk. To
order a one-year
(four-disk)
subscription,
call toll free
800-247-5470
(in IA 800-532-1272).
Please specify which
computer you are
using.

Amiga Banner Printer

Walter Bulawa

Here's a banner-printing program with an unusual twist. In addition to the usual Amiga characters, you can use any of the Amiga's disk-based custom character fonts. A dot-matrix or laser printer is required.

This Amiga BASIC program allows you to construct and print a banner of enlarged letters using any of the 13 fonts present on the Workbench disk. You can use any combination of fonts on the same banner. Program 1, the banner printing program, requires that two special files called *graphics.bmap* and *diskfont.bmap* be present on the same disk as the program itself. The *graphics.bmap* file is included in the Basic-Demos folder on the Amiga Extras disk supplied with the computer (the disk which contains BASIC). The *diskfont.bmap* file is not included on disk; however, you can create a copy by running Program 2. (This program is taken from *Advanced Amiga BASIC* by Tom Halfhill and Charles Brannon, available from COMPUTE! Books.) Once you have both *graphics.bmap* and *diskfont.bmap*, you can copy them either to the same folder as Program 1 or to the LIBS folder of the same disk. (Once you've used Program 2 to create *diskfont.bmap*, you won't need Program 2 again except to create additional copies of that file.)

Banner Construction

When you run Program 1, it opens a window where you can construct a banner. The white area near the bottom of the window represents the printer paper, with the left edge of the display corresponding to the top edge of the printer paper. The

small vertical line is the cursor.

Letters that you type on the keyboard appear in the work area with the current character font. You can move the cursor to any position in the white work area by dragging it with the mouse pointer. The Mouse menu allows you to use the mouse for two other purposes as well—drawing and erasing pixels in the work area. This facility lets you add graphics to text or erase text that you wish to eliminate.

The upper portion of the window indicates which font is currently in use. To change fonts, simply choose the desired font from the Font Selection menu. Except for the Topaz fonts, which are contained in ROM, new fonts will be loaded from the Workbench disk. Once the program has found the font, it identifies and displays the font on the screen. You may then type in the work area with that font.

It is important to remember that the white area represents the banner as it will be produced on the printer. So, should you wish larger letters for the banner, use the Box Height menu to select a narrower height for the white area. The more you shrink the work area, the larger the characters appear on the paper. Changing the work area's height always erases the work area completely.

This program ordinarily uses the X character to form the banner characters. However, you can select a different printing character with the Change Printer Char option of the Action menu. Simply type the new character when prompted.

Printing

Once you have finished writing on the work area, choose the Print

Banner option from the Action menu to print the banner. Make sure that the printer is connected and turned on before you take this action. It's also important that you use the correct printer driver for your printer. To check or change the printer driver, click on the Preferences tool from the Workbench and choose the Change Printer option.

You can abort the printing at any time by pressing the ESC key. Printing always begins at the left margin of the work area; to avoid wasting paper, it's usually best to locate the first character close to the left margin. If your banner message doesn't completely fill the text window, the printer will print blank lines representing the unused portion. To avoid wasting paper, you may want to press ESC to halt printing as soon as all of your message has been printed.

When the banner is printed, the characters tend to look somewhat stretched compared to their appearance on the screen. The Printer lines/display column option allows you to correct for the stretching effect, depending on what height is selected under the Box Height menu. Good results can often be obtained by using a value about half as large as the default value.

For instructions on entering these programs, please refer to "COMPUTE!'s Guide to Typing in Programs" elsewhere in this issue.

Program 1: Amiga Banner Printer

```
'4
DEFINT a-z4
LIBRARY "graphics.library"4
LIBRARY "diskfont.library"4
DECLARE FUNCTION OpenDiskFont( )
```

```

LIBRARY4
DECLARE FUNCTION OpenFont&() LIB
RARY4
DIM TextAttr&(1),FontName$(13),F
ontSize(13)4
DIM choice(4)4
xmin=0:xmax=615:ymin=107:ymax=18
4:LpC%=1:CpL%=14
false=0:true=-1:kwit=false4
pen=1:choice(2)=1:choice(3)=44
xe=70:ye=204
PChar$="X"4
4
MENU 1,0,1,"Action"4
MENU 1,1,1," Print Banner"4
MENU 1,2,1," Erase Box"4
MENU 1,3,1," Change Printer Cha
r"4
MENU 1,4,1," Change Ptr lines/c
ol"4
MENU 1,5,1," Quit"4
4
MENU 2,0,1,"Mouse"4
MENU 2,1,2," Draw"4
MENU 2,2,1," Erase"4
MENU 2,3,1," Position cursor"4
4
MENU 3,0,1,"Box Height"4
MENU 3,1,1," 10 pixels"4
MENU 3,2,1," 20 pixels"4
MENU 3,3,1," 40 pixels"4
MENU 3,4,2," 80 pixels"4
4
MENU 4,0,1,"Font Selection"4
FOR i%=1 TO 134
READ FontName$(i%),FontSize(i%)4
MENU 4,i%,1," "+FontName$(i%)+S
TR$(FontSize(i%))4
NEXT i% 4
4
'Set-up the Screen Display4
COLOR 3,04
LOCATE 3,1:PRINT "Font:"4
LOCATE 7,1:PRINT "Printer charac
ter:"4
PRINT "Printer lines/display col
umn:"4
COLOR 1,04
LOCATE 7,20:PRINT PChar$4
LOCATE 8,30:PRINT LpC%4
'Get Info on current font4
Rp&=WINDOW(8)4
CALL AskFont&(Rp&,VARPTR(TextAtt
r&(&)))4
FontSize=TextAttr&(1)\65536&4
FontName$="topaz"4
Font.Default&=OpenFont&(VARPTR(T
extAttr&(&)))4
Font.Active&=Font.Default&4
BaseLine=PEEKW(Rp&+62)4
GOSUB Show.Example4
GOSUB Erase.Box 'Display Banner
Box4
4
'Main Loop4
ON MOUSE GOSUB Mouse.Action:MOUS
E ON4
ON MENU GOSUB Menu.Request:MENU
ON4
4
WHILE NOT kwit4
c$=INKEY$4
IF c$<>" " THEN4
IF ASC(c$)=13 THEN4
GOSUB Erase.Cursor4
xc=xmin:yc=yc+FontSize+34
GOSUB Yc.Check4
GOSUB Move.Cursor4
ELSE4
GOSUB Erase.Cursor4
COLOR 2,14
PRINT RIGHT$(c$,1);4
xc=PEEKW(Rp&+36):yc=PEEKW(Rp&+38
)4
GOSUB Show.Cursor 'display new c

```

```

ursor4
END IF4
END IF 4
WEND4
4
Done:4
COLOR 1,04
MENU RESET4
CALL CloseFont&(Rp&,Font.Active&
)4
CALL SetFont&(Rp&,Font.Default&)
4
LIBRARY CLOSE4
END4
4
Mouse.Action:4
WHILE MOUSE(0)<>04
mx=MOUSE(1):my=MOUSE(2)4
IF mx<xmin THEN mx=xmin4
IF mx>xmax THEN mx=xmax4
IF cursor.mode THEN 'Move
cursor w/mouse4
GOSUB Erase.Cursor4
yc=my:xc=mx4
GOSUB Yc.Check4
GOSUB Move.Cursor4
ELSE 'Draw
w/mouse 4
IF my<ymin THEN my=ymin4
IF my>ymax THEN my=ymax4
PSET (mx,my),pen4
END IF4
WEND4
RETURN4
4
Menu.Request:4
mnu=MENU(0):item=MENU(1)4
IF choice(mnu)<>0 THEN MENU mnu,
choice(mnu),14
choice(mnu)=item4
MENU mnu,choice(mnu),2 4
ON mnu GOSUB Menu.1,Menu.2,Menu.
3,Menu.44
c$=""4
RETURN4
4
Menu.1:4
ON item GOSUB Print.it,Erase.Box
,Choose.PChar,Choose.LpC,Quit4
RETURN4
4
Menu.2:4
IF item=1 THEN pen=24
IF item=2 THEN pen=14
cursor.mode=false4
IF item=3 THEN cursor.mode=true4
RETURN4
4
Menu.3:4
LINE (xmin,ymin)-(xmax,ymax),0,b
f 'erase old box4
BoxHeight%=10*2^(item-1)4
ymax=ymin+BoxHeight%-14
GOSUB Erase.Box 'display new bo
x4
LpC%=80/BoxHeight%:CpL%=LpC%4
CALL SetFont&(Rp&,Font.Default&)
4
COLOR 1,0:LOCATE 8,30:PRINT LpC%
4
CALL SetFont&(Rp&,Font.Active&)4
GOSUB Move.Cursor4
RETURN4
4
Menu.4:4
F=04
TextAttr&(0)=SADD(FontName$(item
)+"font"+CHR$(0))4
TextAttr&(1)=FontSize(item)*6553
6&4
IF item<3 THEN4
F&=OpenFont&(VARPTR(TextAttr&(0)
)) 'ROM fonts4
ELSE4
F&=OpenDiskFont&(VARPTR(TextAttr

```

```

&(0))) 'Disk fonts4
END IF4
IF F&=0 THEN RETURN4
GOSUB Erase.Example:GOSUB Erase.
Cursor4
IF Font.Active&<>0 AND Font.Acti
ve&<>Font.Default& THEN CALL Clo
seFont&(Rp&,Font.Active&)4
Font.Active&=F&4
FontSize=FontSize(item)4
FontName$=FontName$(item)4
CALL SetFont&(Rp&,Font.Active&)4
BaseLine=PEEKW(Rp&+62)4
GOSUB Show.Example4
GOSUB Yc.Check4
GOSUB Move.Cursor4
RETURN4
4
Choose.PChar:4
CALL SetFont&(Rp&,Font.Default&)
4
COLOR 1,0:LOCATE 10,1:PRINT "Ent
er new printer character: "4
c$="":WHILE c$="" :c$=INKEY$:WEND
4
IF ASC(c$)>32 THEN4
PChar$=c$4
LOCATE 7,20:PRINT PChar$4
END IF4
LOCATE 10,1:PRINT STRING$(30,32)
4
CALL SetFont&(Rp&,Font.Active&)4
GOSUB Move.Cursor4
GOSUB Delay4
RETURN4
4
Choose.LpC:4
CALL SetFont&(Rp&,Font.Default&)
4
COLOR 1,0:LOCATE 10,1:PRINT "Ent
er lines/col: "4
c$="":WHILE c$="" :c$=INKEY$:WEND
4
IF VAL(c$)<LpC% AND VAL(c$)>0 TH
EN4
LpC%=VAL(c$)4
LOCATE 8,30:PRINT LpC%4
END IF4
LOCATE 10,1:PRINT STRING$(30,32)
4
CALL SetFont&(Rp&,Font.Active&)4
GOSUB Move.Cursor4
GOSUB Delay4
RETURN4
4
Choose.LpC:4
CALL SetFont&(Rp&,Font.Default&)
4
COLOR 1,0:LOCATE 10,1:PRINT "Ent
er lines/col: "4
c$="":WHILE c$="" :c$=INKEY$:WEND
4
IF VAL(c$)<LpC% AND VAL(c$)>0 TH
EN4
LpC%=VAL(c$)4
LOCATE 8,30:PRINT LpC%4
END IF4
LOCATE 10,1:PRINT STRING$(30,32)
4
CALL SetFont&(Rp&,Font.Active&)4
GOSUB Move.Cursor4
GOSUB Delay4
RETURN4
4
Print.it:4
Prt.Stop=false4
OPEN "PRT:" FOR OUTPUT AS #14
PRINT #1,CHR$(27);"[0z";CHR$(27)
;"#3";4
CALL SetFont&(Rp&,Font.Default&)
4
COLOR 1,0:LOCATE 10,1:PRINT "Pres
s ESC to Abort Print"4
FOR x=xmin TO xmax4
p$=""4

```

```

IF INKEY$=CHR$(27) THEN GOTO Print.Done
FOR y=ymin TO ymax STEP -1
IF POINT(x,y)<>1 THEN
c$=PChar$
ELSE
c$=" "
END IF
FOR i=1 TO CpL: p$=p$+c$:NEXT
NEXT y
FOR i=1 TO LpC%
PRINT #1,p$
NEXT i
NEXT x
Print.Done:
CLOSE #1
LOCATE 10,1:PRINT STRING$(30," ")
CALL SetFont(Rp&,Font.Active)
GOSUB Move.Cursor
RETURN
Erase.Box:
LINE (xmin,ymin)-(xmax,ymax),1,B
f
xc=xmin:yc=ymin+BaseLine
GOSUB Move.Cursor
RETURN
Quit:
kwit=true
RETURN
Move.Cursor:
CALL move(Rp&,xc,yc)
Show.Cursor:
ytemp=yc-BaseLine
LINE (xc,ytemp)-(xc,ytemp+FontSize-1),3
RETURN

```

```

Yc.Check:
ymn=ymin+BaseLine
IF yc<ymn THEN yc=ymn
ymx=ymax-FontSize+BaseLine+1
IF yc>ymx THEN yc=ymx
RETURN
Erase.Example:
COLOR 0,0
CALL move(Rp&,xe,ye)
CALL ClearEOL(Rp&)
RETURN
Show.Example:
COLOR 2,1
CALL move(Rp&,xe,ye)
c$=FontName$+STR$(FontSize)
CALL Text(Rp&,SADD(c$),LEN(c$))
RETURN
Erase.Cursor:
ytemp=yc-BaseLine
LINE (xc,ytemp)-(xc,ytemp+FontSize-1),1
RETURN
Delay:
FOR j=1 TO 1000:NEXT
RETURN
FontTypes:
DATA topaz,8,topaz,9
DATA diamond,12
DATA garnet,9,garnet,16
DATA ruby,8,ruby,12
DATA emerald,20
DATA opal,11
DATA sapphire,14,sapphire,15,sapphire,18,sapphire,19

```

Program 2: Diskfont.bmap Filemaker

```

'DiskfontMaker
file$="BMAPS/Diskfont.bmap"
READ filesize,checksum
PRINT "Checking DATA statements.
..":PRINT
FOR i=1 TO filesize
READ a$:a=VAL("&h"+a$)
check=check+a
NEXT i
RESTORE DiskFontData
IF check<>checksum THEN PRINT "C
checksum mismatch -- error in typ
ing.":END
PRINT "DATA ok, creating the fil
e."
ON ERROR GOTO CreationError
OPEN file$ FOR OUTPUT AS #1
FOR i=1 TO filesize
READ a$:a=VAL("&h"+a$)
PRINT#1,CHR$(a)
NEXT i
CLOSE#1
PRINT "Finished."
END
CreationError:
PRINT "ERROR #";ERR:END
DATA 34,3196
DiskFontData:
DATA 4F,70,65,6E,44,69,73,6B,46,
6F,6E,74,00,FF,E2,09
DATA 00,41,76,61,69,6C,46,6F,6E,
74,73,00,FF,DC,09,01
DATA 02,00

```

©

CASINO CRAPS

A computer program for the beginning or experienced shooter.

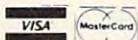
C 64/128 APPLE II IBM PC DISK

COMPUTER GAMBLING — "It's a MUST HAVE..." for aspiring or experienced craps players"

Completely simulates various casinos' house rules and game variations. Allows every betting option available on a casino craps table.

Place • Buy • Hardway • Pass • Don't Pass
Come • Don't Come • Odds • And Much More!

As advertised in Gambling Times!
It's the same as the real thing!
Reviews: "Excellent" "Highly Recommended!"
The 1st Complete Craps Program



Only \$19.95 plus \$3.00 shipping and handling
Credit cards, 1-800-443-2751 ext. 58 24 Hour Service
Check/Money Order to Casino Software
505 Lewis, Rock Springs, WY 82901



Public Domain & User Supported Software

NEW TOP TEN FOR COMMODORE 64 \$5.00/DISK

- The 64 GOLD Library
- ☐ 105 ARTIST SKETCHBOOK drawing programs
 - ☐ 106 GREAT AMERICAN NOVELISTS word processing
 - ☐ 107 PHONE CONNECTIONS communications
 - ☐ 108 SPACE WARS space games
 - ☐ 109 DUNGEONS & DRAGONS text adventures
 - ☐ 110 HOME ORCHESTRA instrument simulation
 - ☐ 111 JUKE BOX prerecorded songs
 - ☐ 112 EINSTEIN'S FAVORITES advanced math
 - ☐ 113 PONZO'S TUTOR programming from BASIC to machine
 - ☐ 114 ELECTRONIC SECRETARY filehandling utilities

NEW TOP TEN FOR IBM \$6.00/DISK

- PC-SIG Authorized Dealer
- ☐ 005 PC-FILE III, V4 labels, forms, and more
 - ☐ 078 PC-WRITE v2.165 popular and powerful search, more
 - ☐ 274 BEST GAMES packman, breakout, wizard, more
 - ☐ 293 ARCADE GAMES (color graphics required)
 - ☐ 405 DESKMATE more than a sidekick
 - ☐ 457 GREATEST ARCADE the best of the best games
 - ☐ 528 NEW YORK WORD sophisticated word processing, 1 of 2
 - ☐ 529 NEW YORK WORD 2 of 2
 - ☐ 557 PINBALL ALLEY from simple to complex pinball games

NEW TOP TEN FOR APPLE \$5.00/DISK

- ☐ 037 FREEWRITER wordprocessor (Apple II + needs paddles)
- ☐ 038 BUSINESS/HOME MANAGEMENT checkbook, calculator, more
- ☐ 039 BEST OF BUSINESS general ledger, payroll, much more
- ☐ 056 BANK'n SYSTEM check balancer, write & print checks
- ☐ 057 OMNI FILE data base with instructions
- ☐ 064 BEST OF EDUCATION math drills, spelling, typing, etc.
- ☐ 085 BASIC MATH DRILLS fractions, multiple choice, work problems
- ☐ 118 GAMES fast action space arcade games
- ☐ 195 PASSIONS nude women, game
- ☐ 213 BEST UTILITIES diskcat, crunch, diskcheck, diskmap, etc.

NEW TOP TEN FOR MAC \$9.00/DISK

- ☐ 005 CODE CRACKING, FEDIT edit file blocks in ASCII or hex
- ☐ 006 ReSD and ReED edit menu bars, icons and I.D. numbers
- ☐ 007 SWITCHER edit multiple Microsoft BASIC files
- ☐ 029 COMMUNICATIONS Red Ryder, MacRep
- ☐ 037 SLIDE SHOW
- ☐ 039 FONTS Font catalog
- ☐ 045 DESK ACCESSORIES Minifinder, timer
- ☐ 062 GAMES Dungeons of doom, baseball
- ☐ 067 GAMES Billiards, volleyball, juggling
- ☐ 086 BEST OF MAC MacWorld 86

PUBLIC DOMAIN SOFTWARE EXCHANGE
Authorized Dealer

Add \$4 shipping & handling per order. CA residents add 6.5% sales tax
Amount enclosed \$ _____ ☐ Check ☐ VISA ☐ MasterCard

Call toll free 800-431-6249
in Calif. 415-952-1994

Card No. _____
Signature _____ Exp. Date _____
Phone (_____) _____
Name _____
Address _____
City _____ State _____ Zip _____



BLACKSHIP
COMPUTER SUPPLY
PO: Box 883362
San Francisco, CA 94188

Using PUT And GET On The IBM PC/PCjr

Rafael Gonzalez

This tutorial for beginning BASIC programmers explains how to use PUT and GET for graphics on the IBM PC/PCjr and compatibles. The program requires a color/graphics card, color monitor, and BASIC on the PC and a color monitor and cartridge BASIC on the PCjr.

IBM BASIC has two commands—PUT and GET—that make it easy for you to animate figures. These powerful commands appear frequently in games, but they have many other uses, as well. (PUT and GET are also used for random file operations, but with a different syntax.)

Bit Pumping

You might think of PUT and GET as "bit pump" operations which move bits from memory onto the screen (PUT) and from the screen into memory (GET). GET reads the colors of the points within a rectangular screen area and stores that information in an array. Here is the basic syntax for the command:

GET (x1,y1)-(x2,y2), array

Each GET command includes two pairs of screen coordinates and an array name. The coordinates define the area to be captured and the array name tells BASIC where to store the image. The first pair of coordinates (x1 and y1 in this example) defines the upper left corner of the rectangle. The second pair (x2 and y2 in this instance) defines the rectangle's lower right corner. (This is identical to the method used to define a rectangle in a LINE command with the B option.)

The array used with GET must be of the numeric type. It can be any precision, although integer arrays are commonly used. Except for very small shapes, you must

DIMension the array before using it. This task, in turn, requires that you calculate how big the array should be. Here is the BASIC formula for calculating the array size:

$$4 + \text{INT}((x * \text{bits per pixel} + 7) / 8) * y$$

In this case, x and y are the lengths of the horizontal and vertical sides of the rectangle, respectively. The *bits per pixel* value is equal to 4 in low resolution, 2 or 4 in medium resolution, and 1 or 2 in high resolution, depending upon the current screen mode.

For example, suppose you want to capture a 10 × 12-pixel image in medium resolution with GET. The number of bytes required is $4 + \text{INT}((10 * 2 + 7) / 8) * 12$, or 40 bytes. Next, you must consider how many bytes each element of the array contains. This factor depends on the array's precision. The following table shows how many bytes are contained in each element of an integer, single-precision, or double-precision array.

Number Of Bytes In Each Array Element

Bytes	Array type
2	integer
4	single precision
8	double precision

Since the example shape requires 40 bytes, it can be stored in an integer array containing 20 elements, a single precision array containing 8 elements, or a double precision array containing 5 elements. It's important to dimension an array of the proper size, since BASIC stops with the error message *Illegal function call* if the array is too small. Using an overly large array doesn't do any harm. However, grossly overlarge arrays waste memory.

The PUT command is the opposite of GET: Once you have stored a shape with GET, PUT can

place the shape anywhere on the screen. Here is the basic syntax:

PUT (x,y), array, action

In this example, x and y set the coordinates where the upper left corner of the image will be placed and the variable *array* identifies the array which contains the shape.

The optional parameter *action* lets you select different modes for a PUT operation. This part of the statement may consist of the word PSET, PRESET, XOR, OR, or AND. If you omit the action parameter, PUT defaults to XOR mode (see below).

Modes For PUT

The PUT mode determines how the placed shape interacts with graphics data that's already present in the same screen area. Type in and save the example program, then run it to see how the mode affects PUT. The program draws a multicolored background and PUTs the same shape on the screen in five different places, using all of the different modes. Here is an explanation of what each mode does.

PSET. In this mode, PUT simply stores the captured data on the screen, overwriting any graphics data that previously existed in the same area. In the example program, the transferred image completely replaces the contents of that screen area.

PRESET. This mode replaces all existing data, just as in PSET mode, but the image is reversed. That is, a value of 0 in the array causes the corresponding point on the screen to have attribute number 3, and vice versa. A value of 1 in the array causes the corresponding point on the screen to have attribute 2, and forth. In the program,

this mode causes the image to have a different color.

AND. The AND mode sets pixels only at points that already contain data matching corresponding data in the transferred image. In the example program, only pixels that are originally cyan remain in the final image.

OR. This mode superimposes an image onto existing data.

XOR. The XOR mode is most often used for animation. When a pixel in the PUT image overlays a point on the screen that contains data, the point is inverted. This feature allows you to move a shape nondestructively over a complex background: When an image is PUT against a background twice, it restores the original data unchanged.

The following table shows how AND, XOR, and OR modes affect screen attributes in medium-resolution mode (SCREEN 1 or SCREEN 4).

Screen Attributes In AND, XOR, And OR Modes

AND

screen	array value			
	0	1	2	3
0	0	0	0	0
1	0	1	0	1
2	0	0	2	2
3	0	1	2	3

OR

screen	array value			
	0	1	2	3
0	0	1	2	3
1	1	1	3	3
2	2	3	2	3
3	3	3	3	3

XOR

screen	array value			
	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

Animation With PUT

The example program also demonstrates simple animation with PUT in XOR mode. After the five large

shapes are drawn, it sends a small shape bouncing around the screen. The process of animation involves four basic steps:

1. Calculate a new position for the shape.
2. PUT the shape on the screen at its previous location (to erase the old image).
3. PUT the shape in its new position.
4. Return to step 1.

Before you enter the loop, you must have PUT the shape on the screen once, so that the PUT in step 2 will erase it. This preliminary step is performed in line 440 of the program. Line 450 saves the old position of the shape in OLDX and OLDY before a new position is calculated in lines 460-490.

BASIC animation with PUT always involves a certain amount of flickering, which results from the delay between the time the old shape is erased and the new one is drawn. To minimize flicker, you should perform the two PUTs as close together as possible. This reduces the amount of time that the shape is invisible. The example program accomplishes this by putting both PUT statements on the same line. The first statement in line 510 erases the old image, and the second statement draws the new one. The do-nothing loop in line 520 holds the new image on the screen for a short interval to alleviate flicker even further. Most programs won't need an explicit delay, since the program will be doing other time-consuming tasks between each redraw.

Once you understand the basics of GET and PUT, you may find many uses for these commands. A drawing program, for instance, may include a feature allowing you to copy one screen area to another. If you capture the indicated area with GET, it is effectively saved in an offscreen buffer, and can be replaced at any time with a simple PUT command. In fact, if sufficient memory is available, you can even GET an entire screen.

To see the effect of a full-screen GET, press any key while the small box is moving on the screen. The program saves the current screen in the array SCRNI%, then PUTs on the screen an image

previously stored in the array SCRNI%. Immediately thereafter, it restores the current image by PUTting the SCRNI% image back on the screen. As you can see, significant delays result from manipulating images of this size.

PUT And GET Demo

For instructions on entering this program, please refer to "COMPUTE!'s Guide to Typing in Programs" in this issue of COMPUTE!.

```

61 100 SCREEN 1:KEY OFF:RANDOMIZ
    E TIMER
CE 110 LINE (60,60)-(120,120),1,
    BF
PF 120 X=61:Y=61
DE 130 BITSPERPIXEL=2
CA 140 NUM=4+INT((X*BITSPERPIXEL
    +7)/8)*Y
FB 150 DIM LARGE%(NUM/2),SMALL%(
    NUM/4)
GF 160 X=320:Y=200
CG 170 NUM=4+INT((X*BITSPERPIXEL
    +7)/8)*Y
MD 180 DIM SCRNI%(NUM/2),SCRNI%(
    NUM/2)
MJ 190 GET (60,60)-(120,120),LA
    RGE%
BC 200 GET (60,60)-(90,90),SM
    ALL%
AA 210 CLS
DB 220 FOR COL=1 TO 3:FOR J=1 TO
    50
JI 230 X=INT(RND*319):Y=INT(RND*
    199)
BI 240 PSET (X,Y),COL:NEXT NEXT
DP 250 GET (0,0)-(319,199),SCRNI
    %
FC 260 FOR J=40 TO 140 STEP 10
OF 270 LINE (0,J)-(320,J+1),1,
    BF
PP 280 LINE (0,J+2)-(320,J+3),2,
    BF
GN 290 LINE (0,J+4)-(320,J+5),3,
    BF
NP 300 NEXT
CO 310 PUT (30,20),LARGE%,PS
    ET
GN 320 PUT (120,20),LARGE%,PR
    ESET
KH 330 PUT (210,20),LARGE%,AN
    D
NH 340 PUT (70,110),LARGE%,OR
FJ 350 PUT (180,110),LARGE%,XO
    R
NB 360 LOCATE 2,6:PRINT "PSET"
KN 370 LOCATE 2,17:PRINT "PRESE
    T"
LO 380 LOCATE 2,29:PRINT "AND"
OH 390 LOCATE 23,13:PRINT "OR"
HA 400 LOCATE 23,26:PRINT "XOR"
JC 410 X=10:Y=50:DX=2:DY=2
JP 420 RLIM=320-32:LLIM=0
KD 430 ULIM=0:DLIM=200-32
EH 440 PUT (X,Y),SMALL%
PI 450 OLDX=X:OLDY=Y
KE 460 IF INKEY$<>" " THEN GET (0
    ,0)-(319,199),SCRNI%:PUT
    (0,0),SCRNI%,AND:PUT (0,0
    ),SCRNI%,PSET
HM 470 X=X+DX
KN 480 IF X=>RLIM OR X<=LLIM THE
    N DX=-DX
IG 490 Y=Y+DY
HF 500 IF Y<=ULIM OR Y>=DLIM THE
    N DY=-DY
FJ 510 PUT (OLDX,OLDY),SMALL%:PU
    T (X,Y),SMALL%
ID 520 FOR J=0 TO 80:NEXT
GL 530 GOTO 450

```

Superplotter

Greg Perkins and Derry Bryson

This 3-D plotting program for the Commodore 64 can handle the difficult problem of removing hidden lines from two-dimensional representations of solid objects. A disk drive is required.

As shown by the popularity of games using 3-D, people are interested in and entertained by three-dimensional displays. "Superplotter" is a program that plots 3-D equations with hidden-line removal and real perspective. Despite the complexity of those tasks—which require a lot of calculations—the program runs relatively quickly.

Type in and save Programs 1–3. Program 1 is the main program, Program 2 is a machine language routine used by the main program, and Program 3 is a BASIC program that displays picture files created by Program 1. Since Program 2 is written in machine language, it must be typed in using the "MLX" machine language entry program listed elsewhere in this issue. Read the MLX instructions carefully before you begin to enter the program. When you run MLX, you'll be asked for a starting address and an ending address for the data you'll be entering. Here are the addresses you need for Program 2:

Starting address: C500
Ending address: C83F

Using Superplotter

To draw a picture with Superplotter, Programs 1 and 2 must both be

in memory. First, load Program 2 with a command like this:

LOAD"ML",8,1

Replace ML with the filename you used when you saved the Program 2 data with MLX. Type NEW and press RETURN; then load and run Program 1. Superplotter then asks you to enter the information it needs to draw the display. If the prompt following the request for information includes a blinking cursor, you'll need to press RETURN after your answer. If no blinking cursor appears, just press the appropriate key. After you've entered all the necessary information, the program waits for you to press any key to start the drawing process. When you've finished viewing the picture, press any key. You then have the option to run the program again, save the picture to disk, redraw the picture (perhaps in a different format), or quit.

Program 3 allows you to display pictures saved on disk without having to wait for them to be redrawn. When you run this program, it asks for the name of the picture file you wish to display. If you enter a dollar sign (\$) at the filename prompt, the program prints a directory of the current disk. Once the picture is on the screen, the program waits for you to press any key.

Plotting New Shapes

The shape drawn by Superplotter is controlled by a function definition in the first line of the program (line

0 of Program 1). To change the picture, use a different function definition. Here's a simple example:

DEF FN F(X)=(X*X+Y*Y)

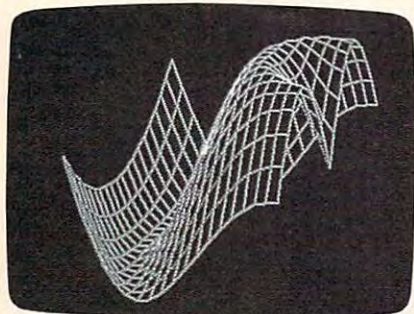
The DEF FN statement creates a user-defined function which is used later in the program. The expression to the right of the equal sign (=) is the part which you can change to modify the resulting shape. The equation must contain several items: a vantage point, x and y domains, a sampling resolution, the method of screen scaling, and whether or not to draw with hidden-line removal. Here's an explanation of those terms.

The *vantage point* is the point in space from which you wish to view the picture. Superplotter uses THETA, PHI, and RHO to define a point in a spherical coordinate system. In fact, they describe a line connecting that point and the origin (0,0,0). THETA is the angle of rotation about the z -axis using the x -axis as a starting point. PHI is the angle between the z -axis and the line. RHO is the distance from the origin to the point. RHO also determines the amount of perspective. The closer the object seems to be, the more pronounced the perspective effect. If your vantage point is inside the object, the picture usually becomes distorted beyond recognition.

The x and y domains are the limits for x and y in this drawing. These values determine how much of the equation is actually plotted on the screen.

The *sampling resolution* controls the number of segments or slices in each domain (*x* and *y*). This value tells Superplotter how many times it must sample the equation in order to draw the picture.

Superplotter can scale (proportion) the picture on the screen in two different ways. Full-screen scaling stretches or squeezes the picture horizontally and vertically to fit exactly on the screen. One-to-one scaling plots the picture exactly in one dimension and adjusts the other dimension to be in an exact one-to-one ratio with the first dimension. This results in truly representational plots for those concerned about mathematical accuracy.



"Superplotter" for the Commodore 64 can draw thousands of different 3-D shapes such as the one shown here.

Hidden-Line Removal

Superplotter also allows two methods for drawing the picture: with or without *hidden-line removal*. This factor determines whether the picture includes lines from surfaces on the opposite side of the object from your vantage point. If the hidden lines are removed, the object looks solid. If they are not, the representation of the object lacks solidity, as if the object were constructed of wire. In most cases, you will want hidden-line removal so that objects look solid. If you wish to see hidden features, however, this feature can be defeated.

For those who are interested, here is a description of how this program accomplishes hidden-line removal. The machine language line-draw routines constantly update a moving border to show the outside edges of the picture being drawn. Before plotting a point, the line-draw routine checks each point against this boundary. If the point is inside the boundary, the point is

not plotted. If the point is outside the border, it is plotted and the moving border is updated to accommodate the new point. By drawing the picture from front to back, the program insures that only visible lines are drawn. Rather than remove lines that have been drawn, the program actually prevents hidden points from being plotted in the first place.

Sample Equations

Listed above are a few equations with parameters that produce interesting pictures. In each case you should substitute the first formula for the statement inside parentheses in the first line of the program. In the Bowl example, for instance, replace the original function definition with this statement:

```
DEF FN F(X)=(X*X+Y*Y)
```

The remainder of the information for each shape tells you how to respond to the program's prompts.

Bowl

Function: $X^2 + Y^2$
Eye: 30,50,200
Domains: -4,4,-4,-4
Segments: 10,10
Scaling: Full screen

SIN of distance

Function: $\text{SIN}(\text{SQR}(X^2 + Y^2))$
Eye: 30,75,200
Domains: -11,11,-11,11
Segments: 25,25
Scaling: Full screen

Thumbtack

Function: $\text{COS}(\text{SQR}(X^2 + Y^2)) / (\text{SQR}(X^2 + Y^2) + 1)$
Eye: 25,89.5,200
Domains: -8,8,-8,8
Segments: 26,26
Scaling: Full screen

Spike

Function: $\text{COS}(\text{SQR}(X^2 + Y^2)) / (10 * \text{SQR}(X^2 + Y^2) + 1)$
Eye: 30,89,200
Domains: -2,2,-2,2
Segments: 16,16
Scaling: Full screen

Modulation

Function: $10 / \text{SQR}(X^2 + Y^2) * \text{SIN}(\text{SQR}(X^2 + Y^2)) * \text{COS}(X/2)$
Eye: 60,70,200
Domains: -15,15,-15,15
Segments: 35,35
Scaling: Full screen

Cross section of SIN of distance

Function: $\text{SIN}(\text{SQR}(X^2 + Y^2))$
Eye: 15,80,200
Domains: -3,3,-3,0
Segments: 20,10
Scaling: Full screen

Program 1: Superplotter

For instructions on entering this program, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```

DE 0 DEF FN F(X)=X*X+Y*Y
AH 10 POKE53280,0:POKE53281,0:
PRINTCHR$(14)" {CLR} {WHT}
{10 SPACES}<<< SUPERPLOT
TER >>>"
JD 20 PRINT "{WHT} {DOWN}
{10 SPACES}3-D FUNCTION
{SPACE}PLOTTER{4}"
AX 30 MM=2304:PRINT:POKE631,14
4
KC 40 POKE632,71:POKE633,111:P
OKE634,53:POKE635,48:POK
E636,13:POKE198,6:LIST-1
JF 50 PRINT"{5}{2 UP}
{6 SPACES}{DOWN}"
CH 60 PRINT"{UP}ENTER EYE COOR
DINATES (THETA DEGREES,"
HS 70 PRINT"{2 SPACES}PHI DEGR
EES, RHO UNITS)"?:INPUTT
H,PH,RH
QH 80 IF TH>=360THENTH=TH-360:
GOTO80
BM 90 IF TH<0THENTH=TH+360:GOT
O90
EE 100 IF PH>=360THENPH=PH-360
:GOTO100
XX 110 IF PH<0THENPH=PH+360:GO
TO110
PX 120 SH=SIN(TH*.0174532778):
CT=COS(TH*.0174532778):
SP=SIN(PH*.0174532778)

```

```

SC 130 CP=COS(PH*.0174532778):
DI=RH
XP 140 PRINT"[DOWN]ENTER DOMAI
NS OF X AND Y (XMIN, XM
AX,"
FR 150 PRINT"[2 SPACES]YMIN, Y
MAX)";:INPUTXS,XE,YS,YE
MD 160 IF XS>XE OR YS>YE THE
N PRINT"[3 UP]";:GOTO14
0
MP 170 PRINT"[DOWN]ENTER THE N
UMBER OF SEGMENTS FOR E
ACH"
EK 180 PRINT"[2 SPACES]DOMAIN
[SPACE]{XSEGS, YSEGS}";
:INPUTSX,SY:SY=INT(SX):
SY=INT(SY)
CQ 190 IF SX<=0 OR SY<=0ORSX*S
Y>MM THENPRINT"[3 UP]";
:GOTO170
FC 200 DIM PT(SY,SX,2)
CM 210 PRINT"[DOWN]SCALE <O>NE
-TO-ONE, OR <F>ULL SCRE
EN? ";
GE 220 GETSC$:IFSC$<>"O"ANDSC$
<>"F"THEN220
HG 230 PRINTSC$
FH 240 PRINT"[DOWN]DRAW WITH
{RVS}HIDDEN-LINE[OFF] R
EMOVAL? (Y/N) ";
QF 250 GETHL$:IFHL$<>"Y"ANDHL$
<>"N"THEN250
CJ 260 PRINTHL$
FF 270 PRINT"[2 DOWN]{8}SAMPLI
NG FUNCTION..."
EB 280 IFPH<180THEN310
JP 290 PH=360-PH:IFTH>180THENT
H=TH-180:GOTO310
FA 300 TH=TH+180
QR 310 X1=XS:X2=XE:Y1=YS:Y2=YE
:DX=1:DY=1
CP 320 IFTH<0ANDTH<180 THENY1=
YE:Y2=YS:DY=-1
DK 330 IFTH<0ORTH>270 THENX1=
XE:X2=XS:DX=-1
GE 340 AY=-1:FORLX=X1TOX2STEPD
X*(X2-X1)/DX:AY=AY+1:
AX=-1
RE 350 PRINT"[UP]{20 RIGHT}"SX
-AY"[LEFT]{2 SPACES}"
SM 360 FORLY=Y1TOY2STEPDY*((Y
-YS)/SY):AX=AX+1:X=LY:Y
=LY:Z=FN(X):EX=-X*SH+Y
*CT
FP 370 EY=-X*CT*CP-Y*SH*CP+Z*S
P:EZ=-X*CT*SP-Y*SH*SP-Z
*CP+RH:X=DI*(EX/EZ)
AG 380 Y=-DI*(EY/EZ):PT(AX,AY,
1)=X:PT(AX,AY,2)=Y
JX 390 IF AX<0ANDAY<0THENM1=X:
M2=Y:M3=Y:M4=Y
DE 400 IFX>M1THENM1=X
FF 410 IFX<M2THENM2=X
QS 420 IFY>M3THENM3=Y
GX 430 IFY<M4THENM4=Y
PB 440 NEXTLY:NEXTLX
KC 450 GOSUB940:PRINT"[UP]SCAL
ING[SHIFT-SPACE]RESULTS
..."
AD 460 IFSC$="F"THEN510
CR 470 IFM1>M3THENM3=M1
SR 480 IFM3>M1THENM1=M3
XC 490 IFM2<M4THENM4=M2
GC 500 IFM4<M2THENM2=M4
MB 510 S1=319/(M1-M2):S2=-S1*M
2:S3=199/(M3-M4):S4=-S3
*M4
MP 520 FORLY=0TOAY:PRINT"[UP]
{18 RIGHT}"AY-LY"[LEFT]
{2 SPACES}";:FORLX=0TOAX
EJ 530 PT(LX,LY,1)=INT((S1*PT(
LX,LY,1)+S2)+.5)

```

```

CK 540 PT(LX,LY,2)=INT((S3*PT(
LX,LY,2)+S4)+.5):NEXTLX
:NEXTLY
BA 550 GOSUB940:PRINT"[UP]
{RVS} PRESS ANY KEY TO
[SPACE]DRAW "
HK 560 GETK$:IFK$=" "THEN560
JB 570 GOSUB940:IFHL$="Y"THENS
YS50905:GOTO590
MX 580 SYS50977
FG 590 IFTH>180THENTH=TH-180:G
OTO590
EC 600 IFTH>90THEN630
PR 610 IFTH>45THEN700
RQ 620 GOTO640
KQ 630 IFTH<135THEN700
ER 640 FORLY=0TOAY-1:FORLX=0TO
AX-1
EH 650 SYS50630,PT(LX,LY,1),PT
(LX,LY,2),PT(LX+1,LY,1)
,PT(LX+1,LY,2)
RS 660 SYS50630,PT(LX,LY,1),PT
(LX,LY,2),PT(LX,LY+1,1)
,PT(LX,LY+1,2):NEXTLX
XP 670 SYS50630,PT(LX,LY,1),PT
(LX,LY,2),PT(LX+1,LY,1)
,PT(LX,LY+1,2):NEXTLY
GK 680 FORLX=0TOAX-1:SYS50630,
PT(LX,LY,1),PT(LX,LY,2)
,PT(LX+1,LY,1),PT(LX+1,
LY,2)
JK 690 NEXTLX:GOTO760
FJ 700 FORLX=0TOAX-1:FORLY=0TO
AY-1
CH 710 SYS50630,PT(LX,LY,1),PT
(LX,LY,2),PT(LX,LY+1,1)
,PT(LX,LY+1,2)
JH 720 SYS50630,PT(LX,LY,1),PT
(LX,LY,2),PT(LX+1,LY,1)
,PT(LX+1,LY,2):NEXTLY
DP 730 SYS50630,PT(LX,LY,1),PT
(LX,LY,2),PT(LX+1,LY,1)
,PT(LX+1,LY,2):NEXTLX
ER 740 FORLY=0TOAY-1:SYS50630,
PT(AX,LY,1),PT(AX,LY,2)
,PT(AX,LY+1,1),PT(AX,LY
+1,2)
RK 750 NEXTLY
MJ 760 GETK$:IFK$=" "THEN760
SR 770 SYS50448:PRINTCHR$(14);
KA 780 PRINT"[UP]<R>UN, <S>AVE
, <RE>DRAW, OR <Q>UIT?
[SPACE]"
MC 790 GETCH$:IFCH$<>"R"ANDCH$
<>"S"ANDCH$<>"D"ANDCH$<
>"Q"THEN790
KH 800 IFCH$="S"THEN860
MF 810 IFCH$="R"THENRUN
SM 820 IFCH$="Q"THENPRINT"[7]
[CLR]";CHR$(142);:POKE5
3280,14:POKE53281,6:LIS
T-1
MR 830 GOSUB940:PRINT"[UP]DRAW
WITH HIDDEN-LINE REMOV
AL? (Y/N) "
GE 840 GETHL$:IFHL$<>"Y"ANDHL$
<>"N"THEN840
SD 850 GOSUB940:GOTO570
FA 860 GOSUB940:FI$=" ":PRINT"
[UP]PICTURE FILENAME";:
INPUTFI$
QM 870 IF FI$=" "THENGOSUB940:P
RINT"[UP]";:GOTO780
GG 880 GOSUB940:PRINT"[UP]SAVI
NG "FI$
ES 890 OPEN2,8,2,"0:"FI$+",P,
W":SYS51197:CLOSE2:OPEN
15,8,15:INPUT#15,A,B$:C
LOSE15
XS 900 GOSUB940:PRINT"[UP]"B$:
IFB$="OK"THENPRINT"[UP]
":GOTO780

```

```

DK 910 PRINT"[UP]{23 RIGHT} (P
RESS ANY KEY)
BH 920 GETK$:IFK$=" "THEN920
BK 930 GOSUB940:PRINT"[UP]";:GO
TO780
CQ 940 PRINT"[UP]";:FORC=1TO38
:PRINT"[RIGHT]{DEL}";:N
EXTC:PRINT:RETURN

```

Program 2: Machine Language Routine

Please refer to the "MLX" article in this issue before entering the following listing.

```

C500:A9 3B 8D 11 D0 A9 08 8D BC
C508:18 D0 A9 00 8D 00 DD 60 91
C510:A9 1B 8D 11 D0 A9 15 8D DE
C518:18 D0 A9 03 8D 00 DD 60 D1
C520:A9 00 85 FA A9 E0 85 FB B8
C528:A9 00 A2 00 A0 00 91 FA FF
C530:C8 D0 FB E6 FB E8 E0 20 A7
C538:D0 F2 A9 10 A0 00 99 00 57
C540:C0 99 00 C1 99 00 C2 99 9A
C548:00 C3 C8 D0 F1 60 A4 02 47
C550:98 29 F8 85 FE 85 FC A9 9B
C558:00 85 FD 06 FC 26 FD 06 E7
C560:FC 26 FD 18 A5 FC 65 FE 20
C568:85 FC A5 FD 69 00 85 FD DE
C570:06 FC 26 FD 06 FC 26 FD 51
C578:06 FC 26 FD 98 29 07 18 7A
C580:65 FC 85 FC A5 FD 69 00 76
C588:85 FD 18 A5 FA 29 F8 65 87
C590:FC 85 FC A5 FB 65 FD 85 EC
C598:FD 18 A9 00 65 FC 85 FC 85
C5A0:A9 E0 65 FD 85 FD A5 FA 30
C5A8:29 07 49 07 AA A9 01 CA EC
C5B0:30 03 0A D0 FA A0 00 A2 60
C5B8:34 78 86 01 11 FC 91 FC F9
C5C0:AE 37 86 01 58 60 20 FD CE
C5C8:AE 20 EB B7 8E 02 00 AD D6
C5D0:14 D0 8D FA 00 AD 15 00 A8
C5D8:8D FB 00 20 FD AE 20 EB 03
C5E0:B7 8E A9 02 AD 14 00 8D 8C
C5E8:A7 02 AD 15 00 8D A8 02 59
C5F0:A9 00 8D AD 02 8D AE 02 83
C5F8:A9 EE 8D 8A C6 8D B7 C6 12
C600:AD A7 02 38 ED FA 00 8D FA
C608:AA 02 AD A8 02 ED FB 00 6B
C610:8D AB 02 10 19 A9 00 8D 8D
C618:B7 C6 38 A9 00 ED AA 02 E3
C620:8D AA 02 A9 00 ED AB 02 0B
C628:8D AB 02 4C 40 C6 AD AA 8F
C630:02 D0 D0 AD AB 02 D0 08 7E
C638:A9 FF 8D AD 02 8D AE 02 CC
C640:AD A9 02 38 ED 02 00 8D D7
C648:AC 02 A9 00 E9 00 8D AF FB
C650:02 10 16 A9 CE 8D 8A C6 C8
C658:A9 00 38 ED AC 02 8D AC D5
C660:02 A9 00 ED AF 02 8D AF 88
C668:02 20 32 C7 AD A9 02 CD A7
C670:02 00 D0 11 AD A7 02 CD 08
C678:FA 00 D0 09 AD A8 02 CD 10
C680:FB 00 D0 01 60 AD AE 02 4F
C688:10 19 EE 02 00 AD AA 02 70
C690:18 6D AD 02 8D AD 02 AD 30
C698:AB 02 6D AE 02 8D AE 02 BA
C6A0:4C 69 C6 38 AD AD 02 ED 21
C6A8:AC 02 8D AD 02 AD AE 02 BF
C6B0:ED AF 02 8D AE 02 A9 EE F9
C6B8:F0 0B EE FA 00 D0 03 EE 47
C6C0:FB 00 4C 69 C6 38 AD FA D9
C6C8:00 E9 01 8D FA 00 AD FB F8
C6D0:00 E9 00 8D FB 00 4C 69 93
C6D8:C6 A9 C7 A0 00 99 00 CA 68
C6E0:99 00 CB C8 D0 F7 A9 00 FA
C6E8:99 41 CB 99 00 CC C8 D0 3C
C6F0:F7 A9 3F 99 81 CC 99 81 5A
C6F8:CD C8 A9 01 99 81 CC 99 EA
C700:81 CD C8 D0 EC A9 00 99 91
C708:11 CE 99 A1 CE C8 D0 F7 54
C710:20 20 C5 20 00 C5 A9 32 0F
C718:8D 6A C6 A9 C7 8D 6B C6 8E

```


COMPUTE!'s Author's Guide

Most of the following suggestions serve to improve the speed and accuracy of publication. COMPUTE! is primarily interested in new and timely articles on the Commodore 64/128, Atari, Apple, IBM PC/PCjr, Amiga, and Atari ST. We are much more concerned with the content of an article than with its style, but articles should be clear and well-explained.

The guidelines below will permit your good ideas and programs to be more easily edited and published:

1. The upper left corner of the first page should contain your name, address, telephone number, and the date of submission.

2. The following information should appear in the upper right corner of the first page. If your article is specifically directed to one make of computer, please state the brand name and, if applicable, the BASIC or ROM or DOS version(s) involved. In addition, *please indicate the memory requirements of programs.*

3. The underlined title of the article should start about 2/3 of the way down the first page.

4. Following pages should be typed normally, except that in the upper right corner there should be an abbreviation of the title, your last name, and the page number. For example: Memory Map/Smith/2.

5. All lines within the text of the article must be double- or triple-spaced. A one-inch margin should be left at the right, left, top, and bottom of each page. No words should be divided at the ends of lines. And please do not justify. Leave the lines ragged.

6. Standard typing paper should be used (no erasable, onionskin, or other thin paper) and typing should be on one side of the paper only (upper- and lowercase).

7. Sheets should be attached together with a paper clip. Staples should not be used.

8. If you are submitting more than one article, send each one in a separate mailer with its own tape or disk.

9. Short programs (under 20 lines) can easily be included within the text. Longer programs should be separate listings. *It is essential that we have a copy of the program, recorded twice, on a tape or disk.* If your article was written with a word processor, we also appreciate a copy of the text file on the tape or disk. Please use high-quality 10 or 30 minute tapes with the program recorded on both sides. The tape or disk should be labeled with the author's name, the title of the article, and, if applicable, the BASIC/ROM/DOS version(s). Atari tapes should specify whether they are to be LOADED or ENTERED. We prefer to receive Apple programs on disk rather than tape. Tapes are fairly sturdy, but disks need to be enclosed within plastic or

cardboard mailers (available at photography, stationery, or computer supply stores).

10. A good general rule is to spell out the numbers zero through ten in your article and write higher numbers as numerals (1024). The exceptions to this are: Figure 5, Table 3, TAB(4), etc. Within ordinary text, however, the zero through ten should appear as words, not numbers. Also, symbols and abbreviations should not be used within text: use "and" (not &), "reference" (not ref.), "through" (not thru).

11. For greater clarity, use all capitals when referring to keys (RETURN, TAB, ESC, SHIFT), BASIC words (LIST, RND, GOTO), and three languages (BASIC, APL, PILOT). Headlines and subheads should, however, be initial caps only, and emphasized words are not capitalized. If you wish to emphasize, underline the word and it will be italicized during typesetting.

12. Articles can be of any length—from a single-line routine to a multi-issue series. The average article is about four to eight double-spaced, typed pages.

13. If you want to include photographs, they should be either 5×7 black and white glossies or color slides.

14. We do not consider articles which are submitted simultaneously to other publishers. If you wish to send an article to another magazine for consideration, please do not submit it to us.

15. COMPUTE! pays between \$70 and \$800 for published articles. In general, the rate reflects the length and quality of the article. Payment is made upon acceptance. Following submission (Editorial Department, COMPUTE! Magazine, P.O. Box 5406, Greensboro, NC 27403) it will take from four to eight weeks for us to reply. If your work is accepted, you will be notified by a letter which will include a contract for you to sign and return. *Rejected manuscripts are returned to authors who enclose a self-addressed, stamped envelope.*

16. If your article is accepted and you have since made improvements to the program, please submit an entirely new tape or disk and a new copy of the article reflecting the update. We cannot easily make revisions to programs and articles. It is necessary that you send the revised version as if it were a new submission entirely, but be sure to indicate that your submission is a revised version by writing, "Revision" on the envelope and the article.

17. COMPUTE! does not accept unsolicited product reviews. If you are interested in serving on our panel of reviewers, contact the Review Coordinator for details.

COMPUTE!'s Guide To Typing In Programs

Computers are precise—type the program *exactly* as listed, including necessary punctuation and symbols, except for special characters noted below. We have provided a special listing convention as well as a program to check your typing—"The Automatic Proofreader."

Programs for the IBM, TI-99/4A, and Atari ST models should be typed exactly as listed; no special characters are used. Programs for Commodore, Apple, and Atari 400/800/XL/XE computers may contain some hard-to-read special characters, so we have a listing system that indicates these control characters. You will find these Commodore and Atari characters in curly braces; do not type the braces. For example, {CLEAR} or {CLR} instructs you to insert the symbol which clears the screen on the Atari or Commodore machines. A complete list of these symbols is shown in the tables below. For Commodore, Apple, and Atari, a single symbol by itself within curly braces is usually a control key or graphics key. If you see {A}, hold down the CONTROL key and press A. This will produce a reverse video character on the Commodore (in quote mode), a graphics character on the Atari, and an invisible control character on the Apple.

Graphics characters entered with the Commodore logo key are enclosed in a special bracket: [<A>]. In this case, you would hold down the Commodore logo key as you type A. Our Commodore listings are in uppercase, so shifted symbols are underlined>. A graphics heart symbol (SHIFT-S) would be listed as S. One exception is {SHIFT-SPACE}. When you see this, hold down SHIFT and press the space bar. If a number precedes a symbol, such as {5 RIGHT}, {6 S}, or {8 Q>}, you would enter five cursor rights, six shifted S's, or eight Commodore-Q's. On the Atari, inverse characters (white on black) should be entered with the inverse video

Atari 400/800/XL/XE

When you see	Type	See
{CLEAR}	ESC SHIFT <	↵ Clear Screen
{UP}	ESC CTRL -	↑ Cursor Up
{DOWN}	ESC CTRL =	↓ Cursor Down
{LEFT}	ESC CTRL +	← Cursor Left
{RIGHT}	ESC CTRL *	→ Cursor Right
{BACK S}	ESC DELETE	⌫ Backspace
{DELETE}	ESC CTRL DELETE	⌫ Delete character
{INSERT}	ESC CTRL INSERT	⌫ Insert character
{DEL LINE}	ESC SHIFT DELETE	⌫ Delete line
{INS LINE}	ESC SHIFT INSERT	⌫ Insert line
{TAB}	ESC TAB	␣ TAB key
{CLR TAB}	ESC CTRL TAB	⌫ Clear tab
{SET TAB}	ESC SHIFT TAB	⌫ Set tab stop
{BELL}	ESC CTRL 2	🔔 Ring buzzer
{ESC}	ESC ESC	⌨ ESCape key

Commodore PET/CBM/VIC/64/128/16/+4

When You Read:	Press:	See:	When You Read:	Press:	See:
{CLR}	SHIFT CLR/HOME	⌫	[1]	COMMODORE	1
{HOME}	CLR/HOME	⌫	[2]	COMMODORE	2
{UP}	SHIFT ↑ CRSR ↓	⬆	[3]	COMMODORE	3
{DOWN}	↑ CRSR ↓	⬇	[4]	COMMODORE	4
{LEFT}	SHIFT ← CRSR →	⬅	[5]	COMMODORE	5
{RIGHT}	← CRSR →	➡	[6]	COMMODORE	6
{RVS}	CTRL 9	⌛	[7]	COMMODORE	7
{OFF}	CTRL 0	⬛	[8]	COMMODORE	8
{BLK}	CTRL 1	⬛	{ F1 }	f1	⌨
{WHT}	CTRL 2	⬜	{ F2 }	SHIFT f1	⌨
{RED}	CTRL 3	⬜	{ F3 }	f3	⌨
{CYN}	CTRL 4	⬜	{ F4 }	SHIFT f3	⌨
{PUR}	CTRL 5	⬜	{ F5 }	f5	⌨
{GRN}	CTRL 6	⬜	{ F6 }	SHIFT f5	⌨
{BLU}	CTRL 7	⬜	{ F7 }	f7	⌨
{YEL}	CTRL 8	⬜	{ F8 }	SHIFT f7	⌨
			←	⬅	⌨

key (Atari logo key on 400/800 models).

Whenever more than two spaces appear in a row, they are listed in a special format. For example, {6 SPACES} means press the space bar six times. Our Commodore listings never leave a single space at the end of a line, instead moving it to the next printed line as {SPACE}.

Amiga program listings contain only one special character, the left arrow (-) symbol. This character marks the end of each program line. Wherever you see a left arrow, press RETURN or move the cursor off the line to enter that line into memory. Don't try to type in the left arrow symbol; it's there only as a marker to indicate where each program line ends.

The Automatic Proofreader

Type in the appropriate program listed below, then save it for future use. The Commodore Proofreader works on the Commodore 128, 64, Plus/4, 16, and VIC-20. Don't omit any lines, even if they contain unfamiliar commands or you think they don't apply to your computer. When you run the program, it installs a machine language program in memory and erases its BASIC portion automatically (so be sure to save several copies before running the program for the first time). If you're using a Commodore 128, Plus/4 or 16, do not use any GRAPHIC commands while the Proofreader is active. You should disable the Commodore Proofreader before running any other program. To do this, either turn the computer off and on or enter SYS 64738 (for the 64), SYS 65341 (128), SYS 64802 (VIC-20), or SYS 65526 (Plus/4 or 16). To reenact the Proofreader, reload the program and run it as usual. Unlike the original VIC/64 Proofreader, this version works the same with disk or tape.

On the Atari, run the Proofreader to activate it (the Proofreader remains active in memory as a machine language program); you must then enter NEW to erase the BASIC loader. Pressing SYSTEM RESET deactivates the Atari Proofreader; enter PRINT USR(1536) to reenact it.

The Apple Proofreader erases the BASIC portion of itself after you run it, leaving only the machine language portion in memory. It works with either DOS 3.3 or ProDOS. Disable the Apple Proofreader by pressing CTRL-RESET before running another BASIC program.

The IBM Proofreader is a BASIC program that simulates the IBM BASIC line editor, letting you enter, edit, list, save, and load programs that you type. Type RUN to activate. Be sure to leave Caps Lock on, except when typing lowercase characters.

Once the Proofreader is active, try typing in a line. As soon as you press RETURN, either a hexadecimal number (on the Apple) or a pair of letters (on the Commodore, Atari, or IBM) appears. The number or pair of letters is called a *checksum*.

Compare the value displayed on the screen by the Proofreader with the checksum printed in the program listing in the magazine. The checksum is given to the left of each line number. Just type in the program a line at a time (without the printed checksum), press RETURN or Enter, and compare the checksums. If they match, go on to the next line. If not, check your typing; you've made a mistake. Because of the checksum method used, do not type abbreviations, such as ? for PRINT. On the Atari and Apple Proofreaders, spaces are not counted as part of the checksum, so be sure you type the right number of spaces between quote marks. The Atari Proofreader does not check to see that you've typed the characters in the right order, so if characters are transposed, the checksum still matches the listing. The Commodore Proofreader catches transposition errors and ignores spaces unless they're enclosed in quotation marks. The IBM Proofreader detects errors in spacing and transposition.

IBM Proofreader Commands

Since the IBM Proofreader replaces the computer's normal BASIC line editor, it has to include many of the direct-mode IBM BASIC commands. The syntax is identical to IBM BASIC. Commands simulated are LIST, LLIST, NEW, FILES, SAVE, and LOAD. When listing your program, press any key (except Ctrl-Break) to stop the listing. If you enter NEW, the Proofreader prompts you to press Y to be especially sure you mean yes.

Two new commands are BASIC and CHECK. BASIC exits the Proofreader back to IBM BASIC, leaving the Proofreader in memory. CHECK works just like LIST, but shows the checksums along with the listing. After you have typed in a program, save it to disk. Then exit the Proofreader with the BASIC command, and load the program as usual (this replaces the Proofreader in memory). You can now run the program, but you may want to re-save it to disk. This will shorten it on disk and make it load faster, but it can no longer be edited with the Proofreader. If you want to convert an existing BASIC program to Proofreader format, save it to disk with SAVE "filename", A.

Program 1: Atari Proofreader

By Charles Brannon, Program Editor

```
100 GRAPHICS 0
110 FOR I=1536 TO 1700:READ A:POKE I,A:CK=CK+A:NEXT I
120 IF CK<>19072 THEN ? "Error in DATA Statement s. Check Typing.":END

130 A=USR(1536)
140 ? :? "Automatic Proofreader Now Activated."
150 END
160 DATA 104,160,0,185,26,3,201,69,240,7
170 DATA 200,200,192,34,208,243,96,200,169,74
180 DATA 153,26,3,200,169,6,153,26,3,162
190 DATA 0,189,0,228,157,74,6,232,224,16
200 DATA 208,245,169,93,141,78,6,169,6,141
210 DATA 79,6,24,173,4,228,105,1,141,95
220 DATA 6,173,5,228,105,0,141,96,6,169
230 DATA 0,133,203,96,247,238,125,241,93,6
240 DATA 244,241,115,241,124,241,76,205,238
250 DATA 0,0,0,0,0,32,62,246,8,201
260 DATA 155,240,13,201,32,240,7,72,24,101
270 DATA 203,133,203,104,40,96,72,152,72,138
280 DATA 72,160,0,169,128,145,88,200,192,40
290 DATA 208,249,165,203,74,74,74,74,24,105
300 DATA 161,160,3,145,88,165,203,41,15,24
310 DATA 105,161,200,145,88,169,0,133,203,104
320 DATA 170,104,168,104,40,96
```

Program 2: IBM Proofreader

By Charles Brannon, Program Editor

```
10 'Automatic Proofreader Version 3.0 (Lines 205,206 added/190 deleted/470,490 changed from V2.0)
100 DIM L$(500),LNUM(500):COLOR 0,7,7:KEY OFF:CLS:MAX=0:LNUM(0)=65536!
110 ON ERROR GOTO 120:KEY 15,CHR$(4)+CHR$(70):ON KEY(15)GOSUB 640:KEY (15) ON:GOTO 130
120 RESUME 130
130 DEF SEG=&H40:W=PEEK(&H4A)
140 ON ERROR GOTO 650:PRINT:PRINT "Proofreader Ready."
150 LINE INPUT L$:Y=CSRLIN-INT(LEN(L$)/W)-1:LOCATE Y,1
160 DEF SEG=0:POKE 1050,30:POKE 1052,34:POKE 1054,0:POKE 1055,79:POKE 1056,13:POKE 1057,28:LINE INPUT L$:DEF SEG:IF L$="" THEN 150
170 IF LEFT$(L$,1)="" THEN L$=MID$(L$,2):GOTO 170
```

```

180 IF VAL(LEFT$(L$,2))=0 AND
MID$(L$,3,1)=" " THEN L$=M
ID$(L$,4)
200 IF ASC(L$)>57 THEN 260 'no
line number, therefore co
mmand
205 BL=INSTR(L$," "):IF BL=0 T
HEN BL=L$:GOTO 206 ELSE B
L$=LEFT$(L$,BL-1)
206 LNUM=VAL(BL$):TEXT$=MID$(L
$,LEN(STR$(LNUM))+1)
210 IF TEXT$="" THEN GOSUB 540
:IF LNUM=LNUM(P) THEN GOSU
B 560:GOTO 150 ELSE 150
220 CKSUM=0:FOR I=1 TO LEN(L$)
:CKSUM=(CKSUM+ASC(MID$(L$,
I)))AND 255:NEXT:LOCATE
Y,1:PRINT CHR$(65+CKSUM/1
6)+CHR$(65+(CKSUM AND 15))
+" "+L$
230 GOSUB 540:IF LNUM(P)=LNUM
THEN L$(P)=TEXT$:GOTO 150
'replace line
240 GOSUB 580:GOTO 150 'insert
the line
260 TEXT$="":FOR I=1 TO LEN(L$)
:A=ASC(MID$(L$,I)):TEXT$=
TEXT$+CHR$(A+32*(A>96 AND
A<123)):NEXT
270 DELIMITER=INSTR(TEXT$," ")
:COMMAND$=TEXT$:ARG$="":IF
DELIMITER THEN COMMAND$=L
EFT$(TEXT$,DELIMITER-1):AR
G$=MID$(TEXT$,DELIMITER+1)
ELSE DELIMITER=INSTR(TEXT
$,CHR$(34)):IF DELIMITER T
HEN COMMAND$=LEFT$(TEXT$,D
ELIMITER-1):ARG$=MID$(TEXT
$,DELIMITER)
280 IF COMMAND$<>"LIST" THEN 4
10
290 OPEN "scrn:" FOR OUTPUT AS
#1
300 IF ARG$="" THEN FIRST=0:P=
MAX-1:GOTO 340
310 DELIMITER=INSTR(ARG$,"-"):
IF DELIMITER=0 THEN LNUM=V
AL(ARG$):GOSUB 540:FIRST=P
:GOTO 340
320 FIRST=VAL(LEFT$(ARG$,DELIM
ITER)):LAST=VAL(MID$(ARG$,
DELIMITER+1))
330 LNUM=FIRST:GOSUB 540:FIRST
=P:LNUM=LAST:GOSUB 540:IF
P=0 THEN P=MAX-1
340 FOR X=FIRST TO P:N$=MID$(S
TR$(LNUM(X)),2)+" "
350 IF CKFLAG=0 THEN A$="":GOT
O 370
360 CKSUM=0:A$=N$+L$(X):FOR I=
1 TO LEN(A$):CKSUM=(CKSUM+
ASC(MID$(A$,I)))AND 255
:NEXT:A$=CHR$(65+CKSUM/16)
+CHR$(65+(CKSUM AND 15))+
"
370 PRINT #1,A$+N$+L$(X)
380 IF INKEY$<>" " THEN X=P
390 NEXT :CLOSE #1:CKFLAG=0
400 GOTO 130
410 IF COMMAND$="LLIST" THEN O
PEN "lpt1:" FOR OUTPUT AS
#1:GOTO 300
420 IF COMMAND$="CHECK" THEN C
KFLAG=1:GOTO 290
430 IF COMMAND$<>"SAVE" THEN 4
50
440 GOSUB 600:OPEN ARG$ FOR OU
TPUT AS #1:ARG$="":GOTO 30
0
450 IF COMMAND$<>"LOAD" THEN 4
90

```

```

460 GOSUB 600:OPEN ARG$ FOR IN
PUT AS #1:MAX=0:P=0
470 WHILE NOT EOF(1):LINE INPU
T #1,L$:BL=INSTR(L$," "):B
L$=LEFT$(L$,BL-1):LNUM(P)=
VAL(BL$):L$(P)=MID$(L$,LEN
(STR$(VAL(BL$)))+1):P=P+1:
WEND
480 MAX=P:CLOSE #1:GOTO 130
490 IF COMMAND$="NEW" THEN INP
UT "Erase program - Are yo
u sure?";L$:IF LEFT$(L$,1)=
"y" OR LEFT$(L$,1)="Y" THE
N MAX=0:LNUM(0)=65536:GOT
O 130 ELSE 130
500 IF COMMAND$="BASIC" THEN C
OLOR 7,0,0:ON ERROR GOTO 0
:CLS:END
510 IF COMMAND$<>"FILES" THEN
520
515 IF ARG$="" THEN ARG$="A:"
ELSE SEL=1:GOSUB 600
517 FILES ARG$:GOTO 130
520 PRINT "Syntax error":GOTO 1
30
540 P=0:WHILE LNUM>LNUM(P) AND
P<MAX:P=P+1:WEND:RETURN
560 MAX=MAX-1:FOR X=P TO MAX:L
NUM(X)=LNUM(X+1):L$(X)=L$(
X+1):NEXT:RETURN
580 MAX=MAX+1:FOR X=MAX TO P+1
STEP -1:LNUM(X)=LNUM(X-1)
:L$(X)=L$(X-1):NEXT:L$(P)=
TEXT$:LNUM(P)=LNUM:RETURN
600 IF LEFT$(ARG$,1)<>CHR$(34)
THEN 520 ELSE ARG$=MID$(A
RG$,2)
610 IF RIGHT$(ARG$,1)=CHR$(34)
THEN ARG$=LEFT$(ARG$,LEN(
ARG$)-1)
620 IF SEL=0 AND INSTR(ARG$,".
")=0 THEN ARG$=ARG$+".BAS"
630 SEL=0:RETURN
640 CLOSE #1:CKFLAG=0:PRINT "St
opped.":RETURN 150
650 PRINT "Error #";ERR:RESUME
150

```

Program 3: Commodore Proofreader

By Philip Nelson, Assistant Editor

```

10 VEC=PEEK(772)+256*PEEK(773)
:LO=43:HI=44
20 PRINT "AUTOMATIC PROOFREADER
FOR ";IF VEC=42364 THEN
{SPACE}PRINT "C-64"
30 IF VEC=50556 THEN PRINT "VI
C-20"
40 IF VEC=35158 THEN GRAPHIC C
LR:PRINT "PLUS/4 & 16"
50 IF VEC=17165 THEN LO=45:HI=
46:GRAPHIC CLR:PRINT"128"
60 SA=(PEEK(LO)+256*PEEK(HI))+
6:ADR=SA
70 FOR J=0 TO 166:READ BYT:POK
E ADR,BYT:ADR=ADR+1:CHK=CHK
+BYT:NEXT
80 IF CHK<>20570 THEN PRINT "**
ERROR* CHECK TYPING IN DATA
STATEMENTS":END
90 FOR J=1 TO 5:READ RF,LF,HF:
RS=SA+RF:HB=INT(RS/256):LB=
RS-(256*HB)
100 CHK=CHK+RF+LF+HF:POKE SA+L
F,LB:POKE SA+HF,HB:NEXT
110 IF CHK<>22054 THEN PRINT "
*ERROR* RELOAD PROGRAM AND

```

```

{SPACE}CHECK FINAL LINE":EN
D
120 POKE SA+149,PEEK(772):POKE
SA+150,PEEK(773)
130 IF VEC=17165 THEN POKE SA+
14,22:POKE SA+18,23:POKESA+
29,224:POKESA+139,224
140 PRINT CHR$(147);CHR$(17);"
PROOFREADER ACTIVE":SYS SA
150 POKE HI,PEEK(HI)+1:POKE (P
EEK(LO)+256*PEEK(HI))-1,0:N
EW
160 DATA 120,169,73,141,4,3,16
9,3,141,5,3
170 DATA 88,96,165,20,133,167,
165,21,133,168,169
180 DATA 0,141,0,255,162,31,18
1,199,157,227,3
190 DATA 202,16,248,169,19,32,
210,255,169,18,32
200 DATA 210,255,160,0,132,180
,132,176,136,230,180
210 DATA 200,185,0,2,240,46,20
1,34,208,8,72
220 DATA 165,176,73,255,133,17
6,104,72,201,32,208
230 DATA 7,165,176,208,3,104,2
08,226,104,166,180
240 DATA 24,165,167,121,0,2,13
3,167,165,168,105
250 DATA 0,133,168,202,208,239
,240,202,165,167,69
260 DATA 168,72,41,15,168,185,
211,3,32,210,255
270 DATA 104,74,74,74,168,1
85,211,3,32,210
280 DATA 255,162,31,189,227,3,
149,199,202,16,248
290 DATA 169,146,32,210,255,76
,86,137,65,66,67
300 DATA 68,69,70,71,72,74,75,
77,80,81,82,83,88
310 DATA 13,2,7,167,31,32,151,
116,117,151,128,129,167,136
,137

```

Program 4: Apple Proofreader

By Tim Victor, Editorial Programmer

```

10 C = 0: FOR I = 768 TO 768 +
68: READ A:C = C + A: POKE I
,A: NEXT
20 IF C < > 7258 THEN PRINT "ER
ROR IN PROOFREADER DATA STAT
EMENTS": END
30 IF PEEK(190 * 256) < > 76 T
HEN POKE 56,0: POKE 57,3: CA
LL 1002: GOTO 50
40 PRINT CHR$(4); "IN#A$300"
50 POKE 34,0: HOME : POKE 34,1:
VTAB 2: PRINT "PROOFREADER
INSTALLED"
60 NEW
100 DATA 216,32,27,253,201,141
110 DATA 208,60,138,72,169,0
120 DATA 72,189,255,1,201,160
130 DATA 240,8,104,10,125,255
140 DATA 1,105,0,72,202,208
150 DATA 238,104,170,41,15,9
160 DATA 48,201,58,144,2,233
170 DATA 57,141,1,4,138,74
180 DATA 74,74,74,41,15,9
190 DATA 48,201,58,144,2,233
200 DATA 57,141,0,4,104,170
210 DATA 169,141,96

```

MLX Machine Language Entry Program For Commodore 64 And 128

Ottis Cowper, Technical Editor

"MLX" is a labor-saving utility that allows almost fail-safe entry of machine language programs. Included are versions for the Commodore 64 and 128.

Type in and save some copies of whichever version of MLX is appropriate for your computer (you'll want to use it to enter future ML programs from COMPUTE!). Program 1 is for the Commodore 64, and Program 2 is for the 128 (128 MLX can also be used to enter Commodore 64 ML programs for use in 64 mode). When you're ready to enter an ML program, load and run MLX. It asks you for a starting address and an ending address. These addresses appear in the article accompanying the MLX-format program listing you're typing.

If you're unfamiliar with machine language, the addresses (and all other values you enter in MLX) may appear strange. Instead of the usual decimal numbers you're accustomed to, these numbers are in *hexadecimal*—a base 16 numbering system commonly used by ML programmers. Hexadecimal—hex for short—includes the numerals 0-9 and the letters A-F. But don't worry—even if you know nothing about ML or hex, you should have no trouble using MLX.

After you enter the starting and ending addresses, you'll be offered the option of clearing the workspace. Choose this option if you're starting to enter a new listing. If you're continuing a listing that's partially typed from a previous session, don't choose this option.

A functions menu will appear. The first option in the menu is ENTER DATA. If you're just starting to type in a program, pick this. Press the E key, and type the first number in the first line of the program listing. If you've already typed in part of a program, type the line number where you left off typing at the end of the previous session (be sure to load the partially completed program before you resume entry). In any case, make sure the address you enter corresponds to the address of a line in the listing you are entering. Otherwise, you'll be unable to enter the data correctly. If you pressed E by mistake, you can return to the command menu by pressing RETURN alone when asked for the address. (You can get back to the menu from most options by pressing RETURN with no other input.)

Entering A Listing

Once you're in Enter mode, MLX prints the address for each program line for you. You then type in all nine numbers on that line, beginning with the first two-digit number after the colon (:). Each line represents eight data bytes and a checksum. Although an MLX-format listing appears similar to the "hex dump" listings from a machine language monitor program, the extra checksum number on the end allows MLX to check your typing. (Commodore 128 users can enter the data from an MLX listing using the built-in monitor if the rightmost column of data is omitted, but we recommend against it. It's much easier to let MLX do the proofreading and error checking for you.)

Figure 1: 64 MLX Keypad

7	8	9	0
4 U	5 I	6 O	F P
1 J	2 K	3 L	E :
A M	B ,	C .	D /
0 Space			

Figure 2: 128 MLX Keypad

A (F1)	B (F3)	C (F5)	D (F7)
7	8	9	E (+)
4	5	6	F (-)
1	2	3	E N T E R
0	.		

When you enter a line, MLX recalculates the checksum from the eight bytes and the address and compares this value to the number from the ninth column. If the values match, you'll hear a bell tone, the data will be added to the workspace area, and the prompt for the next line of data will appear. But if MLX detects a typing error, you'll hear a low buzz and see an error message. The line will then be redisplayed for editing.

Invalid Characters Banned

Only a few keys are active while you're entering data, so you may have to unlearn some habits. You *do not* type spaces between the columns; MLX automatically inserts these for you. You *do not* press RETURN after typing the last number in a line; MLX automatically enters and checks the line after you type the last digit.

Only the numerals 0-9 and the letters A-F can be typed in. If you press any other key (with some exceptions noted below), you'll hear a warning buzz. To simplify typing, 128 MLX redefines the function keys and + and - keys on the numeric keypad so that you can enter data one-handed. In either case, the keypad is active only while entering data. Addresses must be entered with the normal letter and number keys. The figures below show the keypad configurations for each version.

MLX checks for transposed characters. If you're supposed to type in A0 and instead enter 0A, MLX will catch your mistake. There is one error that can slip past MLX: Because of the checksum formula used, MLX won't notice if you accidentally type FF in place of 00, and vice versa. And there's a very slim chance that you could garble a line and still end up with a combination of characters that adds up to the proper checksum. However, these mistakes should not occur if you take reasonable care while entering data.

Editing Features

To correct typing mistakes before finishing a line, use the INST/DEL key to delete the character to the left of the cursor. (The cursor-left key also deletes.) If you mess up a line really badly, press CLR/HOME to start the line over. The RETURN key is also active, but only before any data is typed on a line. Pressing RETURN at this point returns you to the command menu. After you

type a character of data, MLX disables RETURN until the cursor returns to the start of a line. Remember, you can press CLR/HOME to quickly get to a line number prompt.

More editing features are available when correcting lines in which MLX has detected an error. To make corrections in a line that MLX has redisplayed for editing, compare the line on the screen with the one printed in the listing, then move the cursor to the mistake and type the correct key. The cursor left and right keys provide the normal cursor controls. (The INST/DEL key now works as an alternative cursor-left key.) You cannot move left beyond the first character in the line. If you try to move beyond the rightmost character, you'll reenter the line. During editing, RETURN is active; pressing it tells MLX to recheck the line. You can press the CLR/HOME key to clear the entire line if you want to start from scratch, or if you want to get to a line number prompt to use RETURN to get back to the menu.

Display Data

The second menu choice, DISPLAY DATA, examines memory and shows the contents in the same format as the program listing (including the checksum). When you press D, MLX asks you for a starting address. Be sure that the starting address you give corresponds to a line number in the listing. Otherwise, the checksum display will be meaningless. MLX displays program lines until it reaches the end of the program, at which point the menu is redisplayed. You can pause the display by pressing the space bar. (MLX finishes printing the current line before halting.) Press space again to restart the display. To break out of the display and get back to the menu before the ending address is reached, press RETURN.

Other Menu Options

Two more menu selections let you save programs and load them back into the computer. These are SAVE FILE and LOAD FILE; their operation is quite straightforward. When you press S or L, MLX asks you for the filename. You'll then be asked to press either D or T to select disk or tape.

You'll notice the disk drive starting and stopping several times during a load or save (save only for the 128 version). Don't panic; this is normal behavior. MLX opens and reads from or writes to the file instead of using the usual LOAD and SAVE commands (128 MLX makes use of BLOAD). Disk users should also note that the drive prefix 0: is automatically added to the filename (line 750 in 64 MLX), so this should *not* be included when entering

the name. This also precludes the use of @ for Save-with-Replace, so remember to give each version you save a different name. The 128 version makes up for this by giving you the option of scratching the existing file if you want to reuse a filename.

Remember that MLX saves the entire workspace area from the starting address to the ending address, so the save or load may take longer than you might expect if you've entered only a small amount of data from a long listing. When saving a partially completed listing, make sure to note the address where you stopped typing so you'll know where to resume entry when you reload.

MLX reports the standard disk or tape error messages if any problems are detected during the save or load. (Tape users should bear in mind that Commodore computers are never able to detect errors during a save to tape.) MLX also has three special load error messages: INCORRECT STARTING ADDRESS, which means the file you're trying to load does not have the starting address you specified when you ran MLX; LOAD ENDED AT address, which means the file you're trying to load ends before the ending address you specified when you started MLX; and TRUNCATED AT ENDING ADDRESS, which means the file you're trying to load extends beyond the ending address you specified when you started MLX. If you see one of these messages and feel certain that you've loaded the right file, exit and rerun MLX, being careful to enter the correct starting and ending addresses.

The 128 version also has a CATALOG DISK option so you can view the contents of the disk directory before saving or loading.

The QUIT menu option has the obvious effect—it stops MLX and enters BASIC. The RUN/STOP key is disabled, so the Q option lets you exit the program without turning off the computer. (Of course, RUN/STOP-RESET also gets you out.) You'll be asked for verification; press Y to exit to BASIC, or any other key to return to the menu. After quitting, you can type RUN again and reenter MLX without losing your data, as long as you don't use the clear workspace option.

The Finished Product

When you've finished typing all the data for an ML program and saved your work, you're ready to see the results. The instructions for loading and using the finished product vary from program to program. Some ML programs are designed to be loaded and run like BASIC programs, so all you need to type is LOAD "filename",8 for disk

(DLOAD "filename" on the 128) or LOAD "filename" for tape, and then RUN. Such programs will usually have a starting address of 0801 for the 64 or 1C01 for the 128. Other programs must be reloaded to specific addresses with a command such as LOAD "filename",8,1 for disk (BLOAD "filename" on the 128) or LOAD "filename",1,1 for tape, then started with a SYS to a particular memory address. On the Commodore 64, the most common starting address for such programs is 49152, which corresponds to MLX address C000. In either case, you should always refer to the article which accompanies the ML listing for information on loading and running the program.

An Ounce Of Prevention

By the time you finish typing in the data for a long ML program, you may have several hours invested in the project. Don't take chances—use our "Automatic Proofreader" to type the new MLX, and then test your copy *thoroughly* before first using it to enter any significant amount of data. Make sure all the menu options work as they should. Enter fragments of the program starting at several different addresses, then use the Display option to verify that the data has been entered correctly. And be sure to test the Save and Load options several times to insure that you can recall your work from disk or tape. Don't let a simple typing error in the new MLX cost you several nights of hard work.

Program 1: MLX For Commodore 64

```
SS 10 REM VERSION 1.1: LINES 8
    30,950 MODIFIED, LINES 4
    85-487 ADDED
EK 100 POKE 56,50:CLR:DIM IN$,
    I,J,A,B,A$,B$,A(7),N$
DM 110 C4=48:C6=16:C7=7:Z2=2:Z
    4=254:Z5=255:Z6=256:Z7=
    127
CJ 120 FA=PEEK(45)+Z6*PEEK(46)
    :BS=PEEK(55)+Z6*PEEK(56)
    :H$="0123456789ABCDEF"
SB 130 R$=CHR$(13):L$="{LEFT}"
    :S$=" " :D$=CHR$(20):Z$=
    CHR$(0):T$="{13 RIGHT}"
CQ 140 SD=54272:FOR I=SD TO SD
    +23:POKE I,0:NEXT:POKE
    [SPACE]SD+24,15:POKE 78
    8,52
FC 150 PRINT "{CLR}"CHR$(142)CH
    R$(8):POKE 53280,15:POK
    E 53281,15
EJ 160 PRINT T$ " {RED}{RVS}
    {2 SPACES}E8 @}
    {2 SPACES}"SPC(28)"
    {2 SPACES}{OFF}{BLU} ML
    X II {RED}{RVS}
    {2 SPACES}"SPC(28)"
    {12 SPACES}{BLU}"
FR 170 PRINT "{3 DOWN}
    {3 SPACES}COMPUTE! 'S MA
```

```

CHINE LANGUAGE EDITOR
[3 DOWN]"
JB 180 PRINT"[BLK]STARTING ADD
RESS[4]";:GOSUB300:SA=A
D:GOSUB1040:IF F THEN18
0
GF 190 PRINT"[BLK]{2 SPACES}EN
DING ADDRESS[4]";:GOSUB
300:EA=AD:GOSUB1030:IF
[SPACE]F THEN190
KR 200 INPUT"[3 DOWN]{BLK}CLEA
R WORKSPACE [Y/N][4]";A
$:IF LEFT$(A$,1)<>"Y"TH
EN220
PG 210 PRINT"[2 DOWN]{BLU}WORK
ING...";:FOR=BS TO BS+
EA-SA+7:POKE I,0:NEXT:P
RINT"DONE"
DR 220 PRINTTAB(10)"[2 DOWN]
{BLK}{RVS} MLX COMMAND
[SPACE]MENU [DOWN][4]";
PRINT T$"{RVS}E[OFF]NTE
R DATA"
BD 230 PRINT T$"{RVS}D[OFF]ISP
LAY DATA":PRINT T$
{RVS}L[OFF]OAD FILE"
JS 240 PRINT T$"{RVS}S[OFF]AVE
FILE":PRINT T$"{RVS}Q
[OFF]UIT[2 DOWN]{BLK}"
JH 250 GET A$:IF A$=N$ THEN250
HK 260 A=0:FOR I=1 TO 5:IF A$=
MID$( "EDLSQ",I,1)THEN A
=I:I=5
FD 270 NEXT:ON A GOTO420,610,6
90,700,280:GOSUB1060:GO
TO250
EJ 280 PRINT"[RVS] QUIT ":INPU
T"[DOWN][4]ARE YOU SURE
[Y/N]";A$:IF LEFT$(A$,
1)<>"Y"THEN220
EM 290 POKE SD+24,0:END
JX 300 IN$=N$:AD=0:INPUTIN$:IF
LEN(IN$)<>4THENRETURN
KF 310 B$=IN$:GOSUB320:AD=A:B$
=MID$(IN$,3):GOSUB320:A
D=AD*256+A:RETURN
PP 320 A=0:FOR J=1 TO 2:A$=MID
$(B$,J,1):B=ASC(A$)-C4+
(A$>"e")*C7:A=A*C6+B
JA 330 IF B<0 OR B>15 THEN AD=
0:A=-1:J=2
GX 340 NEXT:RETURN
CH 350 B=INT(A/C6):PRINT MID$(
H$,B+1,1);:B=A-B*C6:PRI
NT MID$(H$,B+1,1);:RETU
RN
RR 360 A=INT(AD/Z6):GOSUB350:A
=AD-A*Z6:GOSUB350:PRINT
":
BE 370 CK=INT(AD/Z6):CK=AD-Z4*
CK+Z5*(CK>Z7):GOTO390
PX 380 CK=CK*Z2+Z5*(CK>Z7)+A
JC 390 CK=CK+Z5*(CK>Z5):RETURN
QS 400 PRINT"[DOWN]STARTING AT
[4]";:GOSUB300:IF IN$<>
N$ THEN GOSUB1030:IF F
[SPACE]THEN400
EX 410 RETURN
HD 420 PRINT"[RVS] ENTER DATA
[SPACE]":GOSUB400:IF IN
$=N$ THEN220
JK 430 OPEN3,3:PRINT
SK 440 POKE198,0:GOSUB360:IF F
THEN PRINT IN$:PRINT"
{UP}[5 RIGHT]";
GC 450 FOR I=0 TO 24 STEP 3:B$
=S$:FOR J=1 TO 2:IF F T
HEN B$=MID$(IN$,I+J,1)
HA 460 PRINT"[RVS]"B$;:IF I<
24THEN PRINT"[OFF]";
HD 470 GET A$:IF A$=N$ THEN470
FK 480 IF (A$>"/"AND A$<"")OR(A
$>"@")AND A$<"G")THEN540
GS 485 A=- (A$="M")-2*(A$=",")-
3*(A$=".")-4*(A$="/")-5
*(A$="J")-6*(A$="K")
FX 486 A=A-7*(A$="L")-8*(A$="")
-9*(A$="U")-10*(A$="I")
-11*(A$="O")-12*(A$="
P")
CM 487 A=A-13*(A$=S$):IF A THE
N A$=MID$( "ABCD123E456F
0",A,1):GOTO 540
MP 490 IF A$=R$ AND((I=0)AND(J
=1)OR F)THEN PRINT B$;:
J=2:NEXT:I=24:GOTO550
KC 500 IF A$="{HOME}" THEN PRI
NT B$:J=2:NEXT:I=24:NEX
T:F=0:GOTO440
MX 510 IF A$="{RIGHT}" AND F TH
ENPRINT B$;:GOTO540
GK 520 IF A$<L$ AND A$<>D$ OR
((I=0)AND(J=1))THEN GOS
UB1060:GOTO470
HG 530 A$=L$+S$+L$:PRINT B$;:
J=2-J:IF J THEN PRINT
[SPACE]L$;:I=I-3
QS 540 PRINT A$;:NEXT J:PRINT
[SPACE]S$;
PM 550 NEXT I:PRINT:PRINT"{UP}
[5 RIGHT]";:INPUT#3,IN$
:IF IN$=N$ THEN CLOSE3:
GOTO220
QC 560 FOR I=1 TO 25 STEP3:B$=
MID$(IN$,I):GOSUB320:IF
I<25 THEN GOSUB380:A(I
/3)=A
PK 570 NEXT:I:IF A<>CK THEN GOSU
B1060:PRINT"[BLK]{RVS}
[SPACE]ERROR: REENTER L
INE [4]";F=1:GOTO440
HJ 580 GOSUB1080:B=BS+AD-SA:FO
R I=0 TO 7:POKE B+I,A(I
):NEXT
QQ 590 AD=AD+8:IF AD>EA THEN C
LOSE3:PRINT"[DOWN]{BLU}
** END OF ENTRY **[BLK]
[2 DOWN]";F=1:GOTO700
GQ 600 F=0:GOTO440
QA 610 PRINT"[CLR]{DOWN}{RVS}
[SPACE]DISPLAY DATA ":G
OSUB400:IF IN$=N$ THEN2
20
RJ 620 PRINT"[DOWN]{BLU}PRESS:
[RVS]SPACE[OFF] TO PAU
SE, [RVS]RETURN[OFF] TO
BREAK[4]{DOWN}"
KS 630 GOSUB360:B=BS+AD-SA:FOR
I=BTO B+7:A=PEEK(I):GOS
UB350:GOSUB380:PRINT S$
;
CC 640 NEXT:PRINT"[RVS]";:A=CK
:GOSUB350:PRINT
KH 650 F=1:AD=AD+8:IF AD>EA TH
ENPRINT"[DOWN]{BLU}** E
ND OF DATA **":GOTO220
KC 660 GET A$:IF A$=R$ THEN GO
SUB1080:GOTO220
EQ 670 IF A$=S$ THEN F=F+1:GOS
UB1080
AD 680 ONFGOTO630,660,630
CM 690 PRINT"[DOWN]{RVS} LOAD
[SPACE]DATA ":OP=1:GOTO
710
PC 700 PRINT"[DOWN]{RVS} SAVE
[SPACE]FILE ":OP=0
RX 710 IN$=N$:INPUT"[DOWN]FILE
NAME[4]";IN$:IF IN$=N$
[SPACE]THEN220
PR 720 F=0:PRINT"[DOWN]{BLK}
[RVS]T[OFF]APE OR [RVS]
D[OFF]ISK: [4]";
FP 730 GET A$:IF A$="T"THEN PR
INT"T{DOWN}":GOTO880
HQ 740 IF A$<>"D"THEN730
HH 750 PRINT"D{DOWN}":OPEN15,8
,15,"I0":B=EA-SA:IN$="
0":+IN$:IF OP THEN810
SQ 760 OPEN 1,8,8,IN$+"P,W":G
OSUB860:IF A THEN220
FJ 770 AH=INT(SA/256):AL=SA-(A
H*256):PRINT#1,CHR$(AL)
;CHR$(AH);
PE 780 FOR I=0 TO B:PRINT#1,CH
R$(PEEK(BS+I));:IF ST T
HEN800
FC 790 NEXT:CLOSE1:CLOSE15:GOT
O940
GS 800 GOSUB1060:PRINT"[DOWN]
{BLK}ERROR DURING SAVE:
[4]";:GOSUB860:GOTO220
MA 810 OPEN 1,8,8,IN$+"P,R":G
OSUB860:IF A THEN220
GE 820 GET#1,A$,B$:AD=ASC(A$+Z
$)+256*ASC(B$+Z$):IF AD
<>SA THEN F=1:GOTO850
RX 830 FOR I=0 TO B:GET#1,A$:P
OKE BS+I,ASC(A$+Z$):IF(
I<>B)AND ST THEN F=2:AD
=I:I=B
FA 840 NEXT:IF ST<>64 THEN F=3
FQ 850 CLOSE1:CLOSE15:ON ABS(F
>0)+1 GOTO960,970
SA 860 INPUT#15,A,A$:IF A THEN
CLOSE1:CLOSE15:GOSUB10
60:PRINT"[RVS]ERROR: "A
$
GQ 870 RETURN
EJ 880 POKE183,PEEK(FA+2):POKE
187,PEEK(FA+3):POKE188,
PEEK(FA+4):IFOP=0THEN92
0
HJ 890 SYS 63466:IF(PEEK(783)A
ND1)THEN GOSUB1060:PRIN
T"[DOWN]{RVS} FILE NOT
[SPACE]FOUND ":GOTO690
CS 900 AD=PEEK(829)+256*PEEK(8
30):IF AD<>SA THEN F=1:
GOTO970
SC 910 A=PEEK(831)+256*PEEK(83
2)-1:F=F-2*(A<EA)-3*(A>
EA):AD=A-AD:GOTO930
KM 920 A=SA:B=EA+1:GOSUB1010:P
OKE780,3:SYS 63338
JF 930 A=BS:B=BS+(EA-SA)+1:GOS
UB1010:ON OP GOTO950:SY
S 63591
AE 940 GOSUB1080:PRINT"[BLU]**
SAVE COMPLETED **":GOT
O220
XP 950 POKE147,0:SYS 63562:IF
[SPACE]ST>0 THEN970
FR 960 GOSUB1080:PRINT"[BLU]**
LOAD COMPLETED **":GOT
O220
DP 970 GOSUB1060:PRINT"[BLK]
{RVS}ERROR DURING LOAD:
[DOWN][4]":ON F GOSUB98
0,990,1000:GOTO220
PP 980 PRINT"INCORRECT STARTIN
G ADDRESS ("":GOSUB360:
PRINT"):RETURN
GR 990 PRINT"LOAD ENDED AT "":
AD=SA+AD:GOSUB360:PRINT
D$:RETURN
FD 1000 PRINT"TRUNCATED AT END
ING ADDRESS":RETURN
RX 1010 AH=INT(A/256):AL=A-(AH
*256):POKE193,AL:POKE1
94,AH
FF 1020 AH=INT(B/256):AL=B-(AH
*256):POKE174,AL:POKE1
75,AH:RETURN

```

```

FX 1030 IF AD<SA OR AD>EA THEN
1050
HA 1040 IF (AD>511 AND AD<40960
)OR(AD>49151 AND AD<53
248)THEN GOSUB1080:F=0
:RETURN
HC 1050 GOSUB1060:PRINT "{RVS}
{SPACE}INVALID ADDRESS
{DOWN}{BLK}":F=1:RETU
RN
AR 1060 POKE SD+5,31:POKE SD+6
,208:POKE SD,240:POKE
{SPACE}SD+1,4:POKE SD+
4,33
DX 1070 FOR S=1 TO 100:NEXT:GO
TO1090
PF 1080 POKE SD+5,8:POKE SD+6,
240:POKE SD,0:POKE SD+
1,90:POKE SD+4,17
AC 1090 FOR S=1 TO 100:NEXT:PO
KE SD+4,0:POKE SD,0:PO
KE SD+1,0:RETURN

```

Program 2: MLX For Commodore 128

```

AE 100 TRAP 960:POKE 4627,128:
DIM NL$,A(7)
XP 110 Z2=2:Z4=254:Z5=255:Z6=2
56:Z7=127:BS=256*PEEK(4
627):EA=65280
FB 120 BE$=CHR$(7):RT$=CHR$(13
):DL$=CHR$(20):SP$=CHR$(
32):LF$=CHR$(157)
KE 130 DEF FNHB(A)=INT(A/256):
DEF FNLB(A)=A-FNHB(A)*2
56:DEF FNAD(A)=PEEK(A)+
256*PEEK(A+1)
JB 140 KEY 1,"A":KEY 3,"B":KEY
5,"C":KEY 7,"D":VOL 15
:IF RGR(0)=5 THEN FAST
FJ 150 PRINT "{CLR}"CHR$(142):C
HR$(8):COLOR 0,15:COLOR
4,15:COLOR 6,15
GQ 160 PRINT TAB(12)"{RED}
{RVS}{2 SPACES}{9 0}
{2 SPACES}"RT$:TAB(12)"
{RVS}{2 SPACES}{OFF}
{BLU} 128 MLX {RED}
{RVS}{2 SPACES}"RT$:TAB
(12)"{RVS}{13 SPACES}
{BLU}"
FE 170 PRINT "{2 DOWN}
{3 SPACES}COMPUTE!S MA
CHINE LANGUAGE EDITOR
{2 DOWN}"
DK 180 PRINT "{BLK}STARTING ADD
RESS{4}":GOSUB 260:IF
{SPACE}AD THEN SA=AD:EL
SE 180
FH 190 PRINT "{BLK}{2 SPACES}EN
DING ADDRESS{4}":GOSUB
260:IF AD THEN EA=AD:E
LSE 190
MF 200 PRINT "{DOWN}{BLK}CLEAR
{SPACE}WORKSPACE [Y/N]?
{4}":GETKEY AS:IF AS<>"
Y" THEN 220
QH 210 PRINT "{DOWN}{BLU}WORKIN
G...":BANK 0:FOR A=BS
{SPACE}TO BS+(EA-SA)+7:
POKE A,0:NEXT A:PRINT "D
ONE"
DC 220 PRINT TAB(10)"{DOWN}
{BLK}{RVS} MLX COMMAND
{SPACE}MENU {4}{DOWN}":
PRINT TAB(13)"{RVS}E
{OFF}INTER DATA"RT$:TAB(
13)"{RVS}D{OFF}ISPLAY D
ATA"RT$:TAB(13)"{RVS}L
{OFF}OAD FILE"

```

```

HB 230 PRINT TAB(13)"{RVS}S
{OFF}AVE FILE"RT$:TAB(1
3)"{RVS}C{OFF}ATALOG DI
SK"RT$:TAB(13)"{RVS}Q
{OFF}UIT{DOWN}{BLK}"
AP 240 GETKEY AS:A=INSTR("EDLS
CQ",AS):ON A GOTO 340,5
50,640,650,930,940:GOSU
B 950:GOTO 240
SX 250 PRINT "STARTING AT":GOS
UB 260:IF (AD<>0)OR(AS=N
L$)THEN RETURN:ELSE 250
BG 260 AS=NL$:INPUT AS:IF LEN(
AS)=4 THEN AD=DEC(AS)
PP 270 IF AD=0 THEN BEGIN:IF A
S<>NL$ THEN 300:ELSE RE
TURN:BEND
MA 280 IF AD<SA OR AD>EA THEN
{SPACE}300
PM 290 IF AD>511 AND AD<65280
{SPACE}THEN PRINT BE$::
RETURN
SQ 300 GOSUB 950:PRINT "{RVS} I
NVALID ADDRESS {DOWN}
{BLK}":AD=0:RETURN
RD 310 CK=FNHB(AD):CK=AD-Z4*CK
+Z5*(CK>Z7):GOTO 330
DD 320 CK=CK*Z2+Z5*(CK>Z7)+A
AH 330 CK=CK+Z5*(CK>Z5):RETURN
QD 340 PRINT BE$:"{RVS} ENTER
{SPACE}DATA ":GOSUB 250
:IF AS=NL$ THEN 220
JA 350 BANK 0:PRINT:F=0:OPEN 3
,3
BR 360 GOSUB 310:PRINT HEX$(AD
)+":::IF F THEN PRINT
{SPACE}L$:PRINT "{UP}
{5 RIGHT}";
QA 370 FOR I=0 TO 24 STEP 3:BS
=SP$:FOR J=1 TO 2:IF F
{SPACE}THEN BS=MID$(L$,
I+J,1)
PS 380 PRINT "{RVS}"BS+LF$::IF
{SPACE}I<24 THEN PRINT "
{OFF}";
RC 390 GETKEY AS:IF (AS>"/" AN
D AS<":") OR(AS>"@ AND
AS<"G") THEN 470
AC 400 IF AS="+" THEN AS="E":G
OTO 470
QB 410 IF AS="-" THEN AS="F":G
OTO 470
FB 420 IF AS=RT$ AND ((I=0) AN
D (J=1) OR F) THEN PRIN
T BS::J=2:NEXT:I=24:GOT
O 480
RD 430 IF AS="{HOME}" THEN PRI
NT BS:J=2:NEXT:I=24:NEX
T:F=0:GOTO 360
XB 440 IF (AS="{RIGHT}") AND F
THEN PRINT BS+LF$::GOT
O 470
JP 450 IF AS<>LF$ AND AS<>DL$
{SPACE}OR ((I=0) AND (J
=1)) THEN GOSUB 950:GOT
O 390
PS 460 AS=LF$+SP$+LF$:PRINT BS
+LF$::J=2-J:IF J THEN P
RINT LF$::I=I-3
GB 470 PRINT AS::NEXT J:PRINT
{SPACE}SP$:
HA 480 NEXT I:PRINT:PRINT "{UP}
{5 RIGHT}":L$="
{27 SPACES}"
DP 490 FOR I=1 TO 25 STEP 3:GE
T#3,AS,BS:IF AS=SP$ THE
N I=25:NEXT:CLOSE 3:GOT
O 220
BA 500 AS=AS+BS:A=DEC(AS):MID$(
L$,I,2)=AS:IF I<25 THE
N GOSUB 320:A(I/3)=A:GE
T#3,AS

```

```

AR 510 NEXT I:IF A<>CK THEN GO
SUB 950:PRINT:PRINT"
{RVS} ERROR: REENTER LI
NE ":F=1:GOTO 360
DX 520 PRINT BE$:B=BS+AD-SA:FO
R I=0 TO 7:POKE B+I,A(I
):NEXT I
XB 530 F=0:AD=AD+8:IF AD<EA T
HEN 360
CA 540 CLOSE 3:PRINT "{DOWN}
{BLU}** END OF ENTRY **
{BLK}{2 DOWN}":GOTO 650
MC 550 PRINT BE$:"{CLR}{DOWN}
{RVS} DISPLAY DATA ":GO
SUB 250:IF AS=NL$ THEN
{SPACE}220
JF 560 BANK 0:PRINT "{DOWN}
{BLU}PRESS: {RVS}SPACE
{OFF} TO PAUSE, {RVS}RE
TURN{OFF} TO BREAK{4}
{DOWN}"
XA 570 PRINT HEX$(AD)+":::GOS
UB 310:B=BS+AD-SA
DJ 580 FOR I=B TO B+7:A=PEEK(I
):PRINT RIGHT$(HEX$(A),
2):SP$:GOSUB 320:NEXT
{SPACE}I
XB 590 PRINT "{RVS}";RIGHT$(HEX
$(CK),2)
GR 600 F=1:AD=AD+8:IF AD>EA TH
EN PRINT "{BLU}** END OF
DATA **":GOTO 220
EB 610 GET AS:IF AS=RT$ THEN P
RINT BE$:GOTO 220
QK 620 IF AS=SP$ THEN F=F+1:PR
INT BE$:
XS 630 ON F GOTO 570,610,570
RF 640 PRINT BE$"{DOWN}{RVS} L
OAD DATA ":OP=1:GOTO 66
0
BP 650 PRINT BE$"{DOWN}{RVS} S
AVE FILE ":OP=0
DM 660 F=0:F$=NL$:INPUT "FILENA
ME{4}":F$:IF F$=NL$ THE
N 220
RF 670 PRINT "{DOWN}{BLK}{RVS}T
{OFF}APE OR {RVS}D{OFF}
ISK: {4}";
SQ 680 GETKEY AS:IF AS="T" THE
N 850:ELSE IF AS<>"D" T
HEN 680
SP 690 PRINT "DISK{DOWN}":IF OP
THEN 760
EH 700 DOPEN#1,(F$+"P"),W:IF
{SPACE}DS THEN AS=D$:GO
TO 740
JH 710 BANK 0:POKE BS-2,FNLB(S
A):POKE BS-1,FNHB(SA):P
RINT "SAVING ":F$:PRINT
MC 720 FOR A=BS-2 TO BS+EA-SA:
PRINT#1,CHR$(PEEK(A)):
IF ST THEN AS="DISK WRI
TE ERROR":GOTO 750
GC 730 NEXT A:CLOSE 1:PRINT"
{BLU}** SAVE COMPLETED
{SPACE}WITHOUT ERRORS *
*":GOTO 220
RA 740 IF DS=63 THEN BEGIN:CLO
SE 1:INPUT "{BLK}REPLACE
EXISTING FILE [Y/N]{4}
":AS:IF AS="Y" THEN SCR
ATCH(F$):PRINT:GOTO 700
:ELSE PRINT "{BLK}":GOTO
660:BEND
GA 750 CLOSE 1:GOSUB 950:PRINT
"{BLK}{RVS} ERROR DURIN
G SAVE: {4}":PRINT AS:G
OTO 220
FD 760 DOPEN#1,(F$+"P"):IF DS
THEN AS=D$:F=4:CLOSE
{SPACE}1:GOTO 790

```

```

PX 770 GET#1,A$,B$:CLOSE 1:AD=
ASC(A$)+256*ASC(B$):IF
[SPACE]AD<>SA THEN F=1:
GOTO 790
KB 780 PRINT"LOADING ";F$:PRIN
T:BLOAD(F$),B0,P(B$):AD
=SA+FNAD(174)-BS-1:F=-2
*(AD<EA)-3*(AD>EA)
RQ 790 IF F THEN 800:ELSE PRIN
T"[BLU]** LOAD COMPLETE
D WITHOUT ERRORS ***":GO
TO 220
ER 800 GOSUB 950:PRINT"[BLK]
{RVS} ERROR DURING LOAD
: [43]"ON F GOSUB 810,8
20,830,840:GOTO220
QJ 810 PRINT"INCORRECT STARTIN
G ADDRESS (";HEX$(AD);"
)":RETURN
DP 820 PRINT"LOAD ENDED AT ";H
EX$(AD):RETURN
EB 830 PRINT"TRUNCATED AT ENDI
NG ADDRESS ("HEX$(EA)")
":RETURN
FP 840 PRINT"DISK ERROR ";A$:R
ETURN
KS 850 PRINT"TAPE":AD=POINTER(
F$):BANK 1:A=PEEK(AD):A
L=PEEK(AD+1):AH=PEEK(AD
+2)
XX 860 BANK 15:SYS DEC("FF68")
,0,1:SYS DEC("FFBA"),1,
1,0:SYS DEC("FFBD"),A,A
L,AH:SYS DEC("FF90"),12
8:IF OP THEN 890
FG 870 PRINT:A=SA:B=EA+1:GOSUB
920:SYS DEC("E919"),3:
PRINT"SAVING ";F$
AB 880 A=BS:B=BS+(EA-SA)+1:GOS
UB 920:SYS DEC("EA18"):
PRINT"[DOWN]{BLU]** TAP
E SAVE COMPLETED ***":GO
TO 220
CP 890 SYS DEC("E99A"):PRINT:I
F PEEK(2816)=5 THEN GOS
UB 950:PRINT"[DOWN]
{BLK}{RVS} FILE NOT FOU
ND ":GOTO 220
GQ 900 PRINT"LOADING ...{DOWN}
":AD=FNAD(2817):IF AD<>
SA THEN F=1:GOTO 800:EL
SE AD=FNAD(2819)-1:F=-2
*(AD<EA)-3*(AD>EA)
JD 910 A=BS:B=BS+(EA-SA)+1:GOS
UB 920:SYS DEC("E9FB"):
IF ST>0 THEN 800:ELSE 7
90
XB 920 POKE193,FNLB(A):POKE194
,FNHB(A):POKE 174,FNLB(
B):POKE 175,FNHB(B):RET
URN
CP 930 CATALOG:PRINT"[DOWN]
{BLU]** PRESS ANY KEY F
OR MENU ***":GETKEY A$:G
OTO 220
MM 940 PRINT BE$[RVS] QUIT
[43]"RT$:"ARE YOU SURE
[SPACE][Y/N]?"GETKEY A
$:IF A$<>"Y" THEN 220:EL
SE PRINT"[CLR]":BANK 1
5:END
JE 950 SOUND 1,500,10:RETURN
AF 960 IF ER=14 AND EL=260 THE
N RESUME 300
MK 970 IF ER=14 AND EL=500 THE
N RESUME NEXT
KJ 980 IF ER=4 AND EL=780 THEN
F=4:A$=DS$:RESUME 800
DQ 990 IF ER=30 THEN RESUME:EL
SE PRINT ERR$(ER):"ERR
OR IN LINE":EL

```

MLX Machine Language Entry Program For Apple

Tim Victor, Editorial Programmer

To make it easier to enter machine language programs into your computer without typos, COMPUTE! is introducing its "MLX" entry program for the Apple II series. It's our best MLX yet. It runs on the II, II+, IIe, and IIc, and with either DOS 3.3 or ProDOS.

A machine language (ML) program is usually listed as a long series of numbers. It's hard to keep your place and even harder to avoid making mistakes as you type in the listing, since an incorrect line looks almost identical to a correct one. To make error-free entry easier, COMPUTE! generally lists ML programs for Commodore and Atari computers in a format designed to be typed in with a utility called "MLX." The MLX program uses a checksum system to catch typing errors almost as soon as they happen.

Apple MLX checks your typing on a line-by-line basis. It won't let you enter invalid characters or let you continue if there's a mistake in a line. It won't even let you enter a line or digit out of sequence. Best of all, you don't have to know anything about machine language to enter ML programs with MLX. Apple MLX makes typing ML programs almost foolproof.

Using Apple MLX

Type in and save some copies of Apple MLX on disk (you'll want to use MLX to enter future ML programs in COMPUTE!). It doesn't matter whether you type it in on a disk formatted for DOS 3.3 or ProDOS. Programs entered with Apple MLX, however, must be saved to a disk formatted with the same operating system as Apple MLX itself.

If you have an Apple IIe or IIc, make sure that the key marked CAPS LOCK is in the down position. Type RUN. You'll be asked for the starting and ending addresses of the ML program. These values vary for each program, so they're given at the beginning of the ML program listing and in the program's accompanying article. Find them and type them in.

Invalid Characters Banned

Apple MLX is fairly flexible about how you type in the numbers. You can put extra spaces between numbers or leave the spaces out entirely, compressing a line into 18 keypresses. Be careful not to put a space between two digits in the middle of a number. Apple MLX will

read two single-digit numbers instead of one two-digit number (F 6 means F and 6, not F6).

You can't enter an invalid character with Apple MLX. Only the numerals 0-9 and the letters A-F can be typed in. If you press any other key (with some exceptions noted below), nothing happens. This safeguards against entering extraneous characters. Even better, Apple MLX checks for transposed characters. If you're supposed to type in A0 and instead enter 0A, Apple MLX will catch your mistake.

The next thing you'll see is a menu asking you to select a function. The first is (E)NTER DATA. If you're just starting to type in a program, pick this. Press the E key, and the program asks for the address where you want to begin entering data. Type the first number in the first line of the program listing if you're just starting, or the line number where you left off if you've already typed in part of a program. Hit the RETURN key and begin entering the data.

Once you're in Enter mode, Apple MLX prints the address for each program line for you. You then type in all nine numbers on that line, beginning with the first two-digit number after the colon (:). Each line represents eight bytes and a checksum. When you enter a line and hit RETURN, Apple MLX recalculates the checksum from the eight bytes and the address. If you enter more or less than nine numbers, or the checksum doesn't exactly match, Apple MLX erases the line you just entered and prompts you again for the same line.

Apple MLX also checks to make sure you're typing in the right line. The address (the number to the left of the colon) is part of the checksum recalculation. If you accidentally skip a line and try to enter incorrect values, Apple MLX won't let you continue. Just make sure you enter the correct starting address; if you don't, you won't be able to enter any of the following lines. Apple MLX will stop you.

Editing Features

Apple MLX also includes some editing features. The left- and right-arrow keys allow you to back up and go forward on the line that you are entering, so you can retype data. Pressing the CONTROL (CTRL) and D keys at the same time (delete) removes the character under the

cursor, shortening the line by one character. Pressing CTRL-I (*insert*) puts a space under the cursor and shifts the rest of the line to the right, making the line one character longer. If the cursor is at the right end of the line, neither CTRL-D nor CTRL-I has any effect.

When you've entered the entire listing (up to the ending address that you specified earlier), Apple MLX automatically leaves Enter mode and redisplay the functions menu. If you want to leave Enter mode before then, press the RETURN key when Apple MLX prompts you with a new line address. (For instance, you may want to leave Enter mode to enter a program listing in more than one sitting; see below.)

Display Data

The second menu choice, (D)ISPLAY DATA, examines memory and shows the contents in the same format as the program listing. You can use it to check your work or to see how far you've gotten. When you press D, Apple MLX asks you for a starting address. Type in the address of the first line you want to see and hit RETURN. Apple MLX displays program lines until you press any key or until it reaches the end of the program.

Save And Load

Two more menu selections let you save programs on disk and load them back into the computer. These are (S)AVE FILE and (L)OAD FILE. When you press S or L, Apple MLX asks you for the filename. The first time you save an ML program, the name you assign will be the program's filename on the disk. If you press L and specify a filename that doesn't exist on the disk, you'll see a disk error message.

If you're not sure why a disk error has occurred, check the drive. Make sure there's a formatted disk in the drive and that it was formatted by the same operating system you're using for Apple MLX (ProDOS or DOS 3.3). If you're trying to save a file and see an error message, the disk might be full. Either save the file on another disk or quit Apple MLX (by pressing the Q key), delete an old file or two, then run Apple MLX again. Your typing should still be safe in memory.

Apple MLX: Machine Language Entry Program

For instructions on entering this program, please refer to "COMPUTE!'s Guide to Typing In Programs" elsewhere in this issue.

```
80 100 N = 9: HOME : NORMAL : PR
INT "APPLE MLX": POKE 34,
2: ONERR GOTO 610
CC 110 VTAB 1: HTAB 20: PRINT "S
TART ADDRESS": GOSUB 530
: IF A = 0 THEN PRINT CHR
$ (7): GOTO 110
8C 120 S = A
```

```
E3 130 VTAB 2: HTAB 20: PRINT "E
ND ADDRESS ";: GOSUB 530
: IF S > A OR A = 0 THE
N PRINT CHR$ (7): GOTO 13
0
20 140 E = A
85 150 PRINT "CHOOSE: (E)
NTER DATA": HTAB 22: PRI
NT "(D)ISPLAY DATA": HTAB
8: PRINT "(L)OAD FILE (
S)AVE FILE (Q)UIT": PRIN
T
AE 160 GET A$: FOR I = 1 TO 5: I
F A$ < > MID$ ("EDLSQ",I,
1) THEN NEXT: GOTO 160
93 170 ON I GOTO 270,220,180,200
: POKE 34,0: END
AF 180 INPUT "FILENAME: ";A$: IF
A$ < > "" THEN PRINT CHR
$ (4); "BLOAD";A$;"A";S
A1 190 GOTO 150
38 200 INPUT "FILENAME: ";A$: IF
A$ < > "" THEN PRINT CHR
$ (4); "BSAVE";A$;"A";S;"
,L"; (E - S) + 1
92 210 GOTO 150
C2 220 GOSUB 590: IF B = 0 THEN
150
9E 230 FOR B = B TO E STEP 8:L =
4:A = B: GOSUB 580: PRIN
T A$;" ";L = 2
85 240 FOR F = 0 TO 7:V(F + 1) =
PEEK (B + F): NEXT: GOS
UB 560:V(9) = C
F2 250 FOR F = 1 TO N:A = V(F):
GOSUB 580: PRINT A$;" ";:
NEXT: PRINT: IF PEEK (4
9152) < 128 THEN NEXT
94 260 POKE 49168,0: GOTO 150
CC 270 GOSUB 590: IF B = 0 THEN
150
48 280 FOR B = B TO E STEP 8
A6 290 HTAB 1:A = B:L = 4: GOSUB
580: PRINT A$;" ";: CAL
L 64668:A$ = "":P = 0: GO
SUB 330: IF L = 0 THEN 15
0
F9 300 GOSUB 470: IF F < > N THE
N PRINT CHR$ (7): GOTO 2
90
27 310 IF N = 9 THEN GOSUB 560:
IF C < > V(9) THEN PRINT
CHR$ (7): GOTO 290
72 320 FOR F = 1 TO 8: POKE B +
F - 1,V(F): NEXT: PRINT
: NEXT: GOTO 150
8E 330 IF LEN (A$) = 33 THEN A$
= 0:P = 0: PRINT CHR$ (7
);
22 340 L = LEN (A$):O$ = A$:O =
P:L$ = "": IF P > 0 THEN
L$ = LEFT$ (A$,P)
E8 350 R$ = "": IF P < L - 1 THE
N R$ = RIGHT$ (A$,L - P -
1)
55 360 HTAB 7: PRINT L$;: FLASH
: IF P < L THEN PRINT MID
$ (A$,P + 1,1): NORMAL:
PRINT R$;
7B 370 PRINT " ";: NORMAL
E6 380 K = PEEK (49152): IF K <
128 THEN 380
C1 390 POKE 49168,0:K = K - 128
58 400 IF K = 13 THEN HTAB 7: PR
INT A$;" ";: RETURN
8A 410 IF K = 32 OR K > 47 AND K
< 58 OR K > 64 AND K < 7
1 THEN A$ = L$ + CHR$ (K)
+ R$:P = P + 1
C1 420 IF K = 4 THEN A$ = L$ + R
$
5F 430 IF K = 9 THEN A$ = L$ + "
" + MID$ (A$,P + 1,1) +
R$
8A 440 IF K = 8 THEN P = P - (P
> 0)
```

```
93 450 IF K = 21 THEN P = P + (P
< L)
9D 460 GOTO 330
37 470 F = 1:D = 0: FOR P = 1 TO
LEN (A$):C$ = MID$ (A$,P
,1): IF F > N AND C$ < >
" " THEN RETURN
88 480 IF C$ < > " " THEN GOSUB
520:V(F) = J + 16 * (D =
1) * V(F):D = D + 1
5F 490 IF D > 0 AND C$ = " " OR
D = 2 THEN D = 0:F = F +
1
88 500 NEXT: IF D = 0 THEN F =
F - 1
17 510 RETURN
85 520 J = ASC (C$):J = J - 48 -
7 * (J > 64): RETURN
AB 530 A = 0: INPUT A$:A$ = LEFT
$ (A$,4): IF LEN (A$) = 0
THEN RETURN
6F 540 FOR P = 1 TO LEN (A$):C$
= MID$ (A$,P,1): IF C$ <
"0" OR C$ > "9" AND C$ <
"A" OR C$ > "Z" THEN A =
0: RETURN
2D 550 GOSUB 520:A = A * 16 + J:
NEXT: RETURN
28 560 C = INT (B / 256):C = B -
254 * C - 255 * (C > 127
):C = C - 255 * (C > 255)
28 570 FOR F = 1 TO 8:C = C * 2
- 255 * (C > 127) + V(F):
C = C - 255 * (C > 255):
NEXT: RETURN
DA 580 I = FRE (0):A$ = "": FOR
I = 1 TO L:T = INT (A / 1
6):A$ = MID$ ("0123456789
ABCDEF",A - 16 * T + 1,1)
+ A$:A = T: NEXT: RETUR
N
IF 590 PRINT "FROM ADDRESS ";: G
OSUB 530: IF S > A OR E <
A OR A = 0 THEN B = 0: R
ETURN
8D 600 B = S + 8 * INT ((A - S)
/ 8): RETURN
86 610 PRINT "DISK ERROR": GOTO
150
```

All the programs in this issue are available on the ready-to-load **COMPUTE! Disk**. To order a one-year (four-disk) subscription, call toll free **800-247-5470** (in IA 800-532-1272). Please specify which computer you are using.

What
C-64* &
C-128* USERS
HAVE BEEN WAITING FOR!!

DATA CHIEF™

10 MBYTE DRIVE
Stores 60 Floppies!



- * FAST
- * EXPANDABLE
- * TRANSPARENT OPERATION
- * MULTIPLE CHAINED PARTITIONS
- * GEOS** COMPATIBLE
- * SUPPORTS SINGLE SIDED C-128 AND CPM
- * BUILT-IN FLOPPY DRIVE
- * SUPPORTS ALL 2.0 TO 7.0 (C-128) DOS COMMANDS
- * REPROGRAMMABLE "BOS PAK"™ (BUFFERED OPERATING SYSTEM)

20 Mbyte Model HFD-20
ONLY \$995.00

10 MBYTE MODEL HFD-10
ONLY \$895.00

5 MBYTE MODEL HFD-5
ONLY \$695.00

The DATA CHIEF has been designed by C-64 users for C-64 users with important features like

- Storage in original individual disk format
- Easy Access to programs via Disk Number
- Example: for disk # 60 "OPEN 15, 8, 15, "H60"

Standard Utilities Included

- Includes Wedge Commands

135 Watt Power Supply

Metal Enclosure houses up to 3 each 1/2 Ht Hard Disks

- Provides Excellent Monitor Stand "P.C. style"

- Allows room for future expansion

Complete Operating Manual

1 Year Full Warranty

The DATA CHIEF is perfect for software and hardware developers, business users, students, and serious computer experimenters as well as for bulletin boards and data base systems

TO ORDER SEND CERTIFIED CHECK OR MONEY ORDER TO

Maryland residents add 5% sales tax

For more information call

(301) 371-4000

Allow 4-6 weeks delivery (U.S. Only)

* C-64 & C-128 are trademarks of

Commodore Electronics, Ltd

** GEOS is a trademark of

Berkeley Softworks



103 BAUGHMAN'S LANE
SUITE 301
FREDERICK, MD 21701

Advertisers Index

Reader Service Number/Advertiser

Page

102	Abacus	29,31
103	Abby's Discount Software	75
104	Acorn of Indiana	112
105	Activision	5
106	Activision	7
	Batteries Included	15
107	Blackship Computer Supply	115
108	Casino Software	115
	C.O.M.B. Direct Marketing Corp.	71
109	CompuServe	IFC
110	ComputAbility	66-67
111	Computer Mail Order	10-11
112	Computer Stationery Printers	74
113	Covox, Inc.	74
114	Davidson & Associates, Inc.	13
115	Dresselhaus	72
116	EPYX	1
117	EPYX	IBC
	Halix Institute	103
118	In Control	132
119	Lycos Computer	32-35
120	Midwest Computer Camp	112
	NRI Schools	52i
121	Omnitronix	103
122	PC Network	20-21
123	Precision Images	72
124	Pro-Tech-Tronics	30
125	Protecto	26-27
126	Silicon Express	25
127	subLOGIC Corporation	BC
128	Sun Remarketing	77
129	Tenex Computer Express	73

Save Your Copies of COMPUTE!



Protect your back issues of *COMPUTE!* in durable binders or library cases. Each binder or case is custom-made in flag-blue binding with embossed white lettering. Each holds a year of *COMPUTE!*. Order several and keep your issues of

COMPUTE! neatly organized for quick reference.

(These binders make great gifts, too!)

Cases:

\$6.95 each;
3 for \$20.00;
6 for \$36.00

Binders

\$8.50 each;
3 for \$24.75;
6 for \$48.00

(Please add \$2.50 per unit for orders outside the U.S.)

Send in your prepaid order with the attached coupon

Mail to: Jesse Jones Industries, P.O. Box 5120,
Dept. Code COTE, Philadelphia, PA 19141

Please send me _____ *COMPUTE!* ☐ cases ☐ binders.
Enclosed is my check or money order for \$ _____.
(U.S. funds only.)

Name _____

Address _____

City _____

State _____ Zip _____

Satisfaction guaranteed or money refunded.
Please allow 4-6 weeks for delivery.

Classified Ads	121
COMPUTE! Books' ST Collection	2
COMPUTE! Disk Subscription	39
COMPUTE! Subscription	17

~~\$36.00~~
COVER
PRICE

NOW
AVAILABLE AT
\$24⁰⁰



TRY COMPUTE! EVERY MONTH—12 ISSUES—AT
33% OFF THE COVER PRICE.

Mr/Ms _____

Street _____

City _____ State _____ Zip _____

- ☐ I prefer 24 issues—2 years—at **\$45.00**
☐ Bill Me ☐ Check Enclosed

J5362

Foreign and Canadian, please add \$6 (U.S.) per year postage.
Offer subject to change without notice.

www.commodore.ca



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS

PERMIT NO. 7478

DES MOINES, IOWA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE!

PO BOX 10954
DES MOINES, IOWA 50347



Train for the Fastest Growing Job Skill in America

Only NRI teaches you to service all computers as you build your own, fully IBM-PC compatible microcomputer

The biggest growth in jobs between now and 1995, according to Department of Labor estimates, will occur in the computer service and repair business, where demand for trained technicians will actually double.

You can cash in on this opportunity—either as a full-time corporate technician or an independent service person—once you've learned all the basics of computers the NRI way.

Get inside the newest, fully IBM-PC compatible Sanyo Microcomputer

As an NRI student, you'll get total hands-on training as you actually build the latest model Sanyo 880 Series computer from the keyboard up. It's fully IBM PC compatible, and, best of all, it runs programs almost twice as fast as an IBM PC. As you assemble the Sanyo 880, you'll perform demonstrations and experiments that will give you a total mastery of computer operation and servicing techniques.

Understanding you get only through experience

You need no previous knowledge to succeed with NRI. You start with the basics, rapidly building on the fundamentals of electronics with bite-size lessons. You perform hands-on experiments with your NRI Discovery Lab® and then move on to master advanced concepts like digital logic, microprocessors, and computer memories.

Learn at home in your spare time

You train in your own home at your own convenience, backed at all times by your own NRI instructor and the entire NRI staff of educators and student service support people. They're always ready to give you guidance, follow your progress, and help you over the rough spots to keep you moving toward your goal.

Get all the facts from NRI's 100-page catalog.



Your incomparable total systems training includes all this:

NRI's unique Discovery Lab® to let you design and modify circuits, diagnose and repair faults • NRI's digital multimeter featuring LCD and four key functions, complete with "talk-you-through" instruction on audio tape to teach you how pros use meters • A digital logic probe that lets you visually examine computer circuits • The latest Sanyo 880 dual speed computer with "intelligent" keyboard • 360K double-density, double-sided disk drive • 8K ROM, 256K RAM • Bundled software, including GW BASIC, MS-DOS, WordStar, CalcStar • Sanyo reference manuals to give you programming guidelines and schematics.

See other side for highlights of NRI's "hands-on" computer training →

SEND CARD TODAY FOR FREE NRI CATALOG

☐ Send me your free catalog on NRI computer electronics training

COMPUTER ELECTRONICS training prepares you to service all computers as you build your own 16-bit, IBM PC-compatible computer. Total systems training includes disk drive, test equipment, bundled software, and NRI Discovery Lab®.

OTHER CAREER OPPORTUNITIES

- | | |
|---|--|
| ☐ TV, Audio, and Video Servicing | ☐ Electronic Design Technology |
| ☐ Satellite Electronics | ☐ Digital Electronics Servicing |
| ☐ Industrial Electronics | ☐ Communications Electronics |
| ☐ Data Communications | ☐ Basic Electronics |
| ☐ Robotics & Industrial Control | |

For career courses approved under G.I. Bill: ☐ Check for details.

Name _____ (Please Print) Age _____

Street _____

City _____ State _____ Zip _____

Accredited by the National Home Study Council.

230-037

 www.commodore.ca

Learn Computer Servicing Skills with NRI's "Hands-On" Training



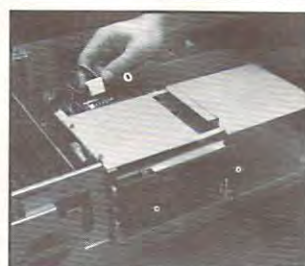
Using NRI's unique Action Audio Cassette, you are talked through the operation and practical application of your hand-held digital multimeter—the basic, indispensable tool for the computer specialist.



You'll set up and perform electronics experiments and demonstrations using your NRI Discovery Lab®. You'll even interface the lab with your computer to "see" keyboard-generated data.



After you build this digital logic probe, you'll explore the operation of the Sanyo detached "intelligent" keyboard and its dedicated microprocessor.

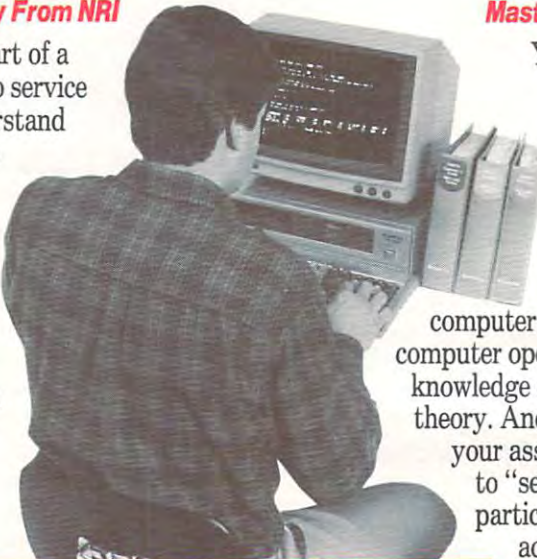


Next, you install the disk drive. You learn disk drive operation and adjustment, make a copy of the MS-DOS operating disk, and begin your exploration of the 8088 CPU.

Total Computer Systems Training, Only From NRI

No computer stands alone... it's part of a total system. And if you want to learn to service and repair computers, you have to understand computer systems. Only NRI includes a powerful computer system as part of your training, centered around the new, fully IBM-PC compatible Sanyo 880 Series computer.

As part of your training, you'll actually build this powerful Sanyo 880 Series IBM-PC compatible computer. It has two operating speeds: the standard IBM speed of 4.77 MHz and a remarkable turbo speed of 8 MHz.



Mastery is "built-in"

You'll assemble the Sanyo "intelligent" keyboard, install the power supply and disk drive, and attach the high resolution monitor.

The demonstrations and experiments you perform as you build your Sanyo computer will give you a total mastery of computer operation, based on a thorough knowledge of the intricacies of computer theory. And, most importantly, during your assembly process you'll be able to "see" for yourself how each particular section of your computer actually works.

SEND TODAY FOR FREE CATALOG

100-page, free catalog tells more... send today



BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 10008 WASHINGTON, D.C.

POSTAGE WILL BE PAID BY ADDRESSEE

NRI Schools

McGraw-Hill Continuing
Education Center
3939 Wisconsin Avenue
Washington, D.C. 20077-9265

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



Send the postage-paid reply card today for NRI's 100-page catalog that gives all the facts about computer training, plus career training in robotics, data communications, TV/audio/video servicing, and many other fields. If the card is missing, write to NRI at the address below.

NRI SCHOOLS

McGraw-Hill Continuing
Education Center
3939 Wisconsin Avenue
Washington, D.C. 20016
We'll give you tomorrow.



www.commodore.ca

COMPUTE!'s FREE Reader Information Service

Use these cards to request FREE information about the products advertised in this issue. Clearly print or type your full name and address. Only one card should be used per person. Circle the numbers that correspond to the key number appearing in the advertisers index.

Send in the card and the advertisers will receive your inquiry. Although every effort is made to insure that only advertisers wishing to provide product information have reader service numbers, COMPUTE! cannot be responsible if advertisers do not provide literature to readers.

Please use these cards *only* for subscribing or for requesting product information. Editorial and customer service inquiries should be addressed to: COMPUTE!, P.O. Box 5406, Greensboro, NC 27403. Check the expiration date on the card to insure proper handling.

Use these cards and this address only for COMPUTE!'s Reader Information Service. Do not send with payment in any form.

COMPUTE!

101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117
118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134
135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151
152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168
169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185
186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202
203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219
220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236
237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253

Circle 181 for more information. New U.S. subscription to COMPUTE! you will be billed for \$24.

Please let us know. Do you own: _____ plan to buy: _____

- | | |
|--|------------------------------|
| ☐ 270 Apple _____ | ☐ 271 |
| ☐ 272 Atari _____ | ☐ 273 |
| ☐ 274 Commodore _____ | ☐ 275 |
| ☐ 276 IBM _____ | ☐ 277 |
| ☐ 278 TI-99/4A _____ | ☐ 279 |
| ☐ 280 Other _____ | ☐ 281 |
| (specify model) | |

Please print or type name and address.
Limit one card per person.

Name _____
Address _____
City _____
State/Province _____ Zip _____
Country _____
Phone _____

Please include ZIP Code

Expiration Date 4/30/87

CO387

SUBSCRIBE TO COMPUTE!

For Fastest Service,
Call Our Toll-Free
US Order Line
800-247-5470
In IA call 1-800-532-1272

- ☐ \$24.00 One Year US Subscription
☐ \$45.00 Two Year US Subscription

Name _____

Address _____

City _____ State _____ Zip _____

- ☐ Payment Enclosed ☐ Bill me
Charge my: ☐ VISA ☐ MasterCard ☐ American Express
Account No. _____

Expires _____

www.commodore.ca

Your subscription will begin with the next available issue. Please allow 4-6 weeks for delivery of first issue. Subscription prices subject to change at any time. Outside the U.S.A. please add \$6 for each subscription.

Place
Stamp
Here

COMPUTE! Reader Service

P.O. Box 2141
Radnor, PA 19089



BUSINESS REPLY MAIL

FIRST CLASS

PERMIT NO. 7478

DES MOINES, IOWA

POSTAGE WILL BE PAID BY ADDRESSEE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



COMPUTE!

PO BOX 10954
DES MOINES, IOWA 50347



The following programs are brought to you by an incredible series of events.

ming, diving, gymnastics and skeet shooting, just to name a few.

There's its equally acclaimed sequel, *Sum-*



Sumo wrestling. A sport of ritual and tradition.



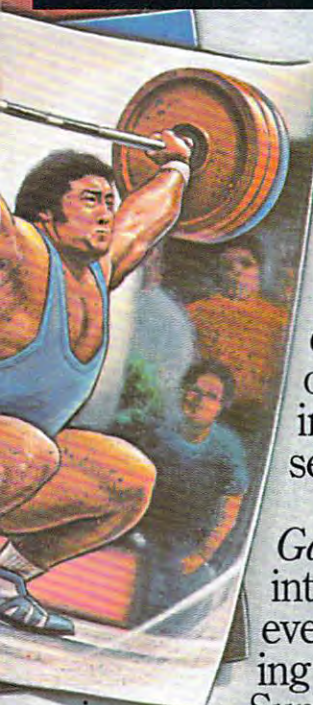
Gymnastics. A graceful display of balance, poise and concentration.



The Bobsled. One wrong move and it's right down the tube.

the hot dog aerials. And beg for mercy in the Biathlon. And coming

STAY ON TOP OF WORLD EVENTS.



Over 30 of them, to be specific.

They're the unforgettable events of our best-selling Games series.

First, *World Games*.™ Eight international events ranging from Sumo wrestling to cliff diving. Bull riding to weightlifting. Even skiing the Chamonix slalom.

There's our enormously popular *Summer Games*.® Break records in track, swim-

mer Games II.™ Go for the gold in rowing, cycling, equestrian, fencing, the high jump, the triple jump, kayaking, and more.

Finally, there's the icy challenges of *Winter Games*.™ Wait'll you careen the tube in a bobsled. Fly the ski jump. Or choreograph an entire figure skating routine. You'll flip out over



Cliff Diving. In Acapulco, everyone falls for it.

next summer, our newest Games program. (Better get ready to hit the



Try your hand at Fencing.

beach.) Now, it may seem like a lot to handle.

But don't let world events get the best of you.



Ride a bucking bull.

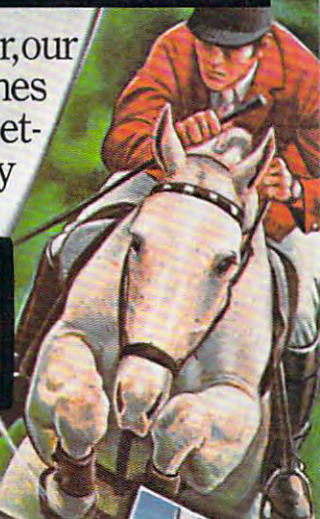


1 To 8 players. Amiga, Apple II & compatibles, Atari ST, C64/128, IBM & compatibles. Coming for the Apple IIGS.

1 To 8 players. Apple II & compatibles, Atari, C64/128.

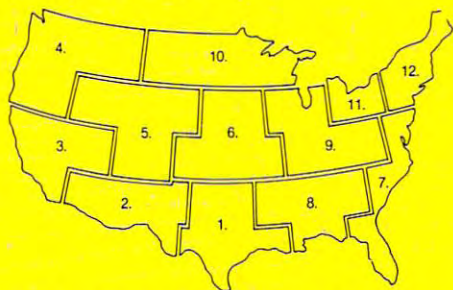
1 To 8 players. Apple II & compatibles, C64/128, IBM & compatibles.

1 To 8 players. Amiga, Apple II & compatibles, Atari ST, C64/128, IBM & compatibles, Macintosh.



Fly to Florida!

Scenery Disk # 7 covers the entire East Coast area from Philadelphia to Miami. The Florida coastline, from Cape Canaveral to Miami, is perfect for concentrated sight-seeing. Or fly to Washington DC, where scenery details include the Capitol Building, Pentagon, and Washington Monument. Whether seeking the intellectual challenge of Flight Simulator or the brute-force fun of Jet, you'll find this latest evolution of SubLOGIC scenery absolutely breath-taking!



Scenery Disks now available: Areas 1-7
San Francisco 'STAR'
Central Japan

See your dealer. SubLOGIC Scenery Disks are available individually for \$19.95. The six-disk Western U.S. set is available for \$99.95. For additional product ordering information or the name of the dealer nearest you, call

subLOGIC
Corporation
713 Edgebrook Drive
Champaign IL 61820
(217) 359-8482 Telex: 206995

ORDER LINE: (800) 637-4983
(except in Illinois, Alaska and Hawaii)
Open 7 AM to 9 PM Central Time

