80×50 Display

Joseph Heaverin

DOUBLE THE VERTICAL RESOLUTION OF YOUR 80-COLUMN SCREEN WITH THIS POWERFUL UTILITY FOR THE 128. INCLUDED IS A ROUTINE THAT ADDS THIS CAPABILITY TO *SPEEDSCRIPT 128*. AN RGB MONITOR IS REQUIRED.

Computer users are on a continuing quest for higher text resolution. One of the first affordable home computers, the VIC-20, had a 22 × 25 text screen. The 64 soon followed with a 40 × 25 screen. The 128, with its 80 × 25 screen, offered even greater text resolution. Now comes "80 × 50 Display," a program that lets you display 50 rows of 80-column text on your 128.

And 80 × 50 Display isn't limited to the PRINT statement—any program that uses the Kernal BSOUT routine will work without modification. This includes BASIC programs, the built-in machine language (ML) monitor, and many ML programs. With the addition of a short wedge, 80 × 50 Display is even compatible with *SpeedScript 128*.

Getting Started

Four programs comprise 80 × 50 Display: Program 1, 80 × 50 Display; Program 2, "Speed Routines"; Program 3, "Patch"; and Program 4, "Speed Boot." Program 1 is a general 80 × 50 display routine, while the others give *SpeedScript 128* this higher text resolution.

Programs 1 and 2 are written entirely in machine language. Type them in using the 128 version of "MLX," the machine language entry program, found elsewhere in this issue. When MLX prompts you, respond with the values given below.

Program 1:

Starting address: 1300
Ending address: 158F

Program 2:

Starting address: 0800
Ending address: 08D7

Before you exit MLX, be sure to save a copy of each program to disk. Save Program 1 as 80X50 DISPLAY and Program 2 as SPEED ROUTINES. It's important that you use these names because Program 4 expects to load these files.

Programs 3 and 4 are BASIC programs. To prevent typing errors while entering these programs, use "The Automatic Proofreader," also found elsewhere in this issue. When you've finished typing, be sure to save a copy of each program to disk. Save Program 3 as PATCH and Program 4 as SPEED BOOT.

Next, copy *SpeedScript 128* to your program disk and run Patch. This program loads *SpeedScript 128*, modifies it to display text in 80 × 50 mode, and then saves the modified version as SPEED80X50.

Using 80 × 50 Display

To load the program and activate the 80 × 50 display mode, type

```
BLOAD"80X50 DISPLAY":SYS 4864
```

Your programs will run as before, only now they'll display twice as much text. To return to the 80 × 25 screen, press ESC ↑. To switch back

to 80 × 50 mode, press ESC ← or press RUN/STOP-RESTORE.

Note that if you use BASIC's WINDOW command, you can't create a window that extends beyond the 25th row even though there are 50 lines of text; attempting to do so will trigger an ILLEGAL QUANTITY ERROR. Instead, you must directly POKE the row and column parameters into the registers at 228–231. For example, to establish a 10 × 10 window in the lower left corner of the screen (at row 40), you'd type POKE 228,49: POKE 229, 40:POKE 230,0:POKE 231,9.

To use the 80 × 50 version of *SpeedScript 128*, load and run Program 4, Speed Boot. (To avoid disk swapping, the files SPEED BOOT, 80X50 DISPLAY, SPEED ROUTINES, and SPEED80X50 should be on the same disk.) All of *SpeedScript*'s commands work normally, but now you'll notice twice as much text appears on the screen as before. (Note that you can't toggle between 80 × 25 and 80 × 50 mode while in the modified version of *SpeedScript*.)

How It Works

80 × 50 Display first copies the ROM routines at $C000–$FFFF to bank 0 RAM, and then it modifies these routines. The Kernal routine BSOUT at $FFD2 is diverted to the modified routines in bank 0. After a character has been printed, the program returns to bank 15.

To speed printing to the screen, the 128 is operated at 2 MHz while in 80-column mode and is switched to 1 MHz when the 40-column screen is used. The top-of-BASIC text storage is moved to $C000, screen memory is stored at $0000–$0FFF in 80-column RAM, and attribute memory is moved to $1000–$1FFF in 80-column RAM (which, for owners of a 128, means all 80-column memory is used; 128D users have 48K of free memory).

Several changes were made to *SpeedScript 128* that greatly increase its response time. Instead of using a loop, the VDC's fill routine pads the end of each text line with spaces. Also, the bottom of RAM to $1000 is made common and the text read and write routines are moved to $0800; this eliminates the need to switch banks when accessing each character.

BEFORE TYPING . . .

Before typing in programs, please refer to "How to Type In COMPUTE!'s Gazette Programs," elsewhere in this issue.

Program 1: 80 × 50 Display

```
1300:AD 27 03 C9 14 D0 06 A9 5D
1308:30 8D 00 FF 60 A0 00 8C BB
1310:12 12 84 16 A9 C0 8D 13 34
1318:12 85 17 A9 01 8D 00 FF 64
1320:B1 16 91 16 C8 D0 F9 E6 9C
1328:17 A5 17 F0 09 C9 FF D0 75
1330:EF A0 05 4C 20 13 A9 30 AC
1338:8D 00 FF A2 08 A9 03 20 5C
1340:CC CD A2 04 A9 40 20 CC 30
1348:CD A2 06 A9 32 20 CC CD D2
1350:E8 A9 3A 20 CC CD A2 00 81
1358:A9 80 20 CC CD A2 14 A9 0F
1360:10 8D 2F 0A 8D 3A CA 20 83
1368:CC CD E8 A9 00 20 CC CD 08
1370:AD 24 03 8D 16 15 AD 25 35
1378:03 8D 17 15 A9 5B 8D 24 B1
1380:03 A9 14 8D 25 03 AD 26 A4
1388:03 8D 14 15 AD 27 03 8D 05
1390:15 15 A9 E6 8D 26 03 A9 DE
1398:14 8D 27 03 AD 39 03 C9 63
13A0:14 F0 29 8D 19 15 AD 38 BB
13A8:03 8D 18 15 A9 6C 8D 38 5A
13B0:03 A9 14 8D 39 03 AD 18 67
13B8:03 8D 12 15 AD 19 03 8D BC
13C0:13 15 A9 41 8D 18 03 A9 7B
13C8:14 8D 19 03 A9 31 85 E4 B1
13D0:85 ED A9 0F 8D 6C C1 8D 8A
13D8:56 C4 8D 87 C1 8D 89 C4 A1
13E0:8D 22 C5 A9 07 8D 35 CA 4D
13E8:A9 1C 8D 5F C1 8D 4D C4 36
13F0:A9 4E 8D 69 C1 8D 53 C4 77
13F8:A9 02 85 F1 A9 15 8D 60 61
1400:C1 8D 4E C4 A9 15 8D 6A A9
1408:C1 8D 54 C4 A9 80 8D 7A 30
1410:CB 8D 8B CB 8D 8E CB 8D 7B
1418:9B CB A9 80 8D 37 CA A9 C6
1420:15 8D 7B CB 8D 8C CB 8D 26
1428:8F CB 8D 9C CB A9 15 8D 43
1430:38 CA A9 4C 8D 59 FA A9 92
1438:01 8D 1B 15 A9 93 4C D2 FF
1440:FF 20 E1 FF D0 12 A9 30 FE
1448:8D 00 FF 20 40 FA 24 D7 47
1450:30 03 20 5F FF 20 00 13 DE
1458:6C 00 0A 08 24 D7 10 08 21
1460:A9 30 8D 00 FF 20 04 15 B8
1468:28 6C 16 15 C9 5F F0 5E DF
1470:C9 5E F0 35 C9 58 D0 2E 06
1478:20 A6 14 20 04 15 24 D7 73
1480:30 10 A9 00 8D 30 D0 AD 76
1488:11 D0 09 10 8D 11 D0 4C 2E
1490:97 14 AD 1B 15 D0 03 A9 8C
1498:18 2C A9 31 85 E4 85 ED D8
14A0:A9 30 8D 00 FF 60 6C 18 CD
14A8:15 A9 00 8D 1B 15 78 20 DC
14B0:81 FF 20 84 FF 20 8A FF 7B
14B8:A9 6C 8D 38 03 A9 14 8D 7A
14C0:39 03 A9 41 8D 18 03 A9 0C
14C8:14 8D 19 03 58 60 24 D7 16
14D0:30 03 20 5F FF 8D 1B 15 4D
14D8:20 00 13 A9 93 20 D2 FF D0
14E0:A9 30 8D 00 FF 60 24 D7 3D
14E8:10 17 8D 1A 15 A9 30 8D 6F
14F0:00 FF AD 1A 15 20 01 15 B0
14F8:A9 00 8D 00 FF AD 1A 15 A7
1500:60 6C 14 15 AD 11 D0 29 C5
1508:6F 8D 11 D0 A9 01 8D 30 19
1510:D0 60 00 00 00 00 00 00 BA
1518:00 00 00 00 00 28 50 78 FB
1520:A0 C8 F0 18 40 68 90 B8 E9
1528:E0 08 30 58 80 A8 D0 F8 91
1530:20 48 70 98 C0 E8 10 38 16
1538:60 88 B0 D8 00 28 50 78 12
1540:A0 C8 F0 18 40 68 90 B8 0A
1548:E0 08 30 58 80 A8 00 00 17
1550:00 00 00 00 00 01 01 01 81
1558:01 01 01 02 02 02 02 02 A1
1560:02 02 03 03 03 03 03 03 C9
1568:04 04 04 04 04 04 05 05 95
1570:05 05 05 05 05 06 06 06 A1
1578:06 06 06 07 07 07 07 07 C1
1580:00 00 00 00 00 00 00 00 AA
1588:00 00 00 00 00 00 00 00 B2
```

Program 2: Speed Routines

```
0800:A9 9F 85 0C A9 00 A2 12 E2
0808:8E 00 D6 2C 00 D6 10 FB 74
0810:8D 01 D6 E8 A9 A0 8E 00 7D
0818:D6 8D 01 D6 AD 13 35 85 2E
0820:FB AD 14 35 85 FC A2 30 05
0828:AD 7D 27 8D 37 08 8D 48 69
0830:08 A9 7E 8D 00 FF A0 4F E7
0838:B1 FB 29 7F C9 1F F0 09 F2
0840:C9 20 F0 05 88 D0 F1 A0 B7
0848:4F C8 84 3B A0 00 A9 1F ED
0850:8D 00 D6 B1 FB 2C 00 D6 84
0858:10 FB 8D 01 D6 C8 29 7F DC
0860:C9 1F F0 04 C4 3B D0 EB 1C
0868:18 98 65 FB 85 FB A5 FC 7B
0870:69 00 85 FC E0 00 D0 02 60
0878:84 03 C0 50 F0 36 84 02 14
0880:A9 00 A0 18 8C 00 D6 2C 39
0888:00 D6 10 FB 8D 01 D6 A9 D7
0890:20 A0 1F 8C 00 D6 2C 00 39
0898:D6 10 FB 8D 01 D6 18 A9 AD
08A0:50 E5 02 C9 01 90 0D A0 34
08A8:1E 8C 00 D6 2C 00 D6 10 77
08B0:FB 8D 01 D6 CA F0 03 4C 1C
08B8:36 08 8E 00 FF A5 FB 8D D3
08C0:1B 35 A5 FC 8D 1C 35 A5 1D
08C8:03 8D 12 35 60 A9 05 8D 94
08D0:06 D5 4C 00 13 50 85 00 C7
```

Program 3: Patch

```
MB 10 BANK1:BLOAD"SPEEDSCRIPT
      {SPACE}128",B1
BB 20 FORI=1TO7:READA$:POKEDEC
      (A$),16:NEXT
JS 30 FORI=1TO10:READA$:POKEDE
      C(A$),76:NEXT
ED 40 POKEDEC("1D54"),8:POKEDE
      C("1D53"),0
QK 50 POKEDEC("1C0F"),205:POKE
      DEC("1C10"),8
HX 60 POKEDEC("21E6"),12:POKED
      EC("21AF"),9
QJ 70 DATA1C34,2632,2646,2652,
      2685,2793,2B63
QH 80 DATA1D52,1E14,21CE,2664,
      2691,27F1,2902,2924,2AC5
      ,2DF6
XC 90 BSAVE"SPEED80X50",B1,P71
      69TOP13584
DB 100 BANK15
```

Program 4: Speed Boot

```
MG 5 TRAP 100
GH 10 JF$="80X50 DISPLAY":BLOA
      D(JF$)
XP 20 JF$="SPEED ROUTINES":BLO
      AD(JF$)
BG 30 JF$="SPEED80X50":RUN(JF$
      )
GX 100 PRINT"INSERT A DISK CON
       TAINING "JF$
KR 110 PRINTTAB(7)"PRESS A KEY
        TO CONTINUE"
CH 120 GETKEY A$:RESUME
```