HI-RES SCRN FUNCTION

This versatile routine gives you a SCRN-like function for Hi-Res page 1.

The Apple's Lo-Res graphics include a handy way for you to learn the color of any pixel on the screen with the SCRN function. Unfortunately, there is no built-in method to do this on the Hi-Res screen. Here is a short routine that will give you a SCRN-like function for Hi-Res page 1.

USING THE PROGRAM

Hi-Res SCRN uses Applesoft's ampersand (&) command. In your program, when you want to learn whether a pixel is on or off, use

```
&X,Y,A%
```

where *X* is the pixel's X-coordinate, *Y* is its Y-coordinate, and *A%* is the variable that will contain either a 0 (if the pixel is off) or a 1 (if it's on). You must use an integer variable to hold the result. To install the command, type

```
BRUN HSF
```

ENTERING THE PROGRAM

To enter the Hi-Res SCRN function, you may either use an assembler or enter the hex code directly from the System Monitor. If you have an assembler, assemble the source code in **Listing 1** and save the object code as HSF. If you use the monitor, enter the hex code from **Listing 2** and save it with the command

```
BSAVE HSF,A$300,L$42
```

Listing 3 provides a short demonstration of how the Hi-Res SCRN function can be used to detect collisions. Enter the program and save it by typing

```
SAVE HSF.DEMO
```

HOW THE PROGRAM WORKS

HSF starts out by setting up Applesoft's ampersand (&) vector. When the ampersand is called, it sends control of the program to location $3F5. From there, control is passed to the HSF routine.

HSF first calls the routine CHKNUM at $F6B9. This Applesoft routine evaluates the X and Y coordinates given. It checks to make sure that X is within the range of 0-279 and that Y is within the range of 0-191. Almost all of the graphics statements in Applesoft use this routine. It stores the results of its evaluation in the Apple registers as follows:

Register X holds the low order bits of the horizontal coordinate.
Register Y holds the high order bits of the horizontal coordinate.
Register A holds the vertical screen coordinate.

HSF then calls the routine GRCALC at $F411. GRCALC takes the registers set up by CHKNUM and calculates the memory address of the specified location. The Hi-Res screen is set up in a very convoluted way. The actual screen in memory is 192 by 40 bytes wide. Each byte contains 7 dots. 40 multiplied by 7 is 280, which is the number of horizontal dots.

GRCALC stores the address of the vertical position of the point into locations $26,$27. It also stores the horizontal position (divided by 7) into location $E5. Location $30 is set up to be the bit position of the point within the byte of the screen. This is the same as the remainder of the horizontal position divided by 7.

HSF then loads the horizontal position (divided by 7) into register Y. Using the indirect address mode of the Apple, it loads the Accumulator with the contents of $26,$27 + $E5. For instance if $26 and $27 contained address $2000 (the first line) and $E5 contained $1, it would load the Accumulator with the contents of $2001.

HSF then strips away the high bit with AND #$7F and stores the result in location WORK. It then loads the Accumulator with the bit position in location $30 (from GRCALC) and strips away its high bit.

HSF then ANDs the bit position and the memory location at $323 and sets the high bit of the Accumulator. It compares the result (in the Accumulator) and the original bit position. If these two are not equal, no dot was found. If they are equal, a dot was found.

The Demo

To demonstrate the use of the HSF, I have written a short demo. In this game, there are two cars, each controlled by a player. The cars ride around the screen leaving tracks of oil, which are represented by the white dots on the screen. If either car runs into any oil, including its own, it loses. HSF determines whether the car has run into any oil.

CONCLUSION

Hi-Res SCRN has a number of uses. It makes collision detection in games much easier. Also, you could turn the Hi-Res screen into a huge, two-dimensional Boolean array. This would allow you to keep track of 53,760 (280 × 192) true-false variables!

LISTING 1: HSF Source Code

```
1  ************************
2  * HSF                  *
3  * HI-RES SCRN FUNCTION *
4  * COPYRIGHT(C) 1988    *
5  * MICROSPARC, INC.     *
6  * CONCORD, MA 01742    *
7  ************************
8  * MERLIN ASSEMBLER     *
9  ************************
10 *
11 * EQUATES
12 *
13 VARPNT   =    $83        ;ADDR OF VARIABLE
14 WORK     =    $FF        ;TEMPORARY LOCATION
15 VECTOR   =    $3F5       ;AMPERSAND VECTOR
16 CHKCOM   =    $DEBE      ;CHECK FOR A COMMA
17 PTRGET   =    $DFE3      ;ADDR OF VARIABLE TO VARPNT
18 GRCALC   =    $F411      ;GETS ADDRESS OF DOT
19 CHKNUM   =    $F6B9      ;GETS X AND Y COORDINATES
20 *
21 * SET UP AMPERSAND VECTOR
22 *
23         ORG   $300
24 START   LDA   #$4C
25         STA   VECTOR
26         LDA   #<BEGIN
27         STA   VECTOR+1
28         LDA   #>BEGIN
29         STA   VECTOR+2
30         RTS
31 *
32 * ACTUAL SCRN FUNCTION
33 *
34 BEGIN   JSR   CHKNUM
35         JSR   GRCALC
36         LDY   $E5
37         LDA   ($26),Y
38         AND   #$7F
39         STA   WORK
40         LDA   $30
41         AND   #$7F
42         AND   WORK
43         ORA   #$80
44         CMP   $30
45         BNE   NONE
46         LDA   #$1
47         BNE   B0
48 NONE    LDA   #$0
49 B0      PHA
50 *
51 * PUT RESULT INTO INTEGER
52 *
53         JSR   CHKCOM
54         JSR   PTRGET
55         LDY   #$0
56         LDA   #$0
57         STA   (VARPNT),Y
58         INY
59         PLA
60         STA   (VARPNT),Y
61         RTS
```

END OF LISTING 1

LISTING 2: HSF

```
Start: 300          Length: 42

24 0300:A9 4C 8D F5 03 A9 10 8D
EA 0308:F6 03 A9 03 8D F7 03 60
68 0310:20 B9 F6 20 11 F4 A4 E5
3E 0318:B1 26 29 7F 85 FF A5 30
CB 0320:29 7F 25 FF 09 80 C5 30
1D 0328:D0 04 A9 01 D0 02 A9 00
28 0330:48 20 BE DE 20 E3 DF A0
E0 0338:00 A9 00 91 83 C8 68 91
F4 0340:83 60

TOTAL: 0225
```

END OF LISTING 2

LISTING 3: HSF.DEMO

```
37 10  REM ************************
C0 20  REM * HSF.DEMO             *
B9 30  REM * BY CHRIS MEYER       *
AE 40  REM * COPYRIGHT(C) 1988    *
CB 50  REM * MICROSPARC, INC.     *
24 60  REM * CONCORD, MA 01742    *
45 70  REM ************************
E5 80  TEXT : HOME : PRINT  CHR$ (21): INVERSE :
       HTAB 15: PRINT " SURROUND ": PRINT : PRINT
        : NORMAL : PRINT "SURROUND IS A GAME IN WH
       ICH TWO PLAYERS": PRINT "RIDE CARS AROUND L
       EAVING TRAILS OF OIL": PRINT "BEHIND THEM.
       NO CAR CAN TOUCH THE OIL"
0C 90  PRINT "AND IF HE DOES, HE LOSES. THE OIL IS
       ": PRINT "REPRESENTED BY THE WHITE DOTS ON
       THE": PRINT "SCREEN. THE CONTROLS ARE AS FO
       LLOWS:": PRINT : PRINT : PRINT : PRINT "LEF
       T CAR              RIGHT CAR"
92 100  PRINT "--------                 ---------";
       PRINT "X - DOWN                  . - DOWN": PRIN
       T "A - LEFT                K - LEFT": PRINT "W
       - UP                O - UP": PRINT "D - RIGHT
                  ; - RIGHT": REM 8 DASHES,12 SPAC
       ES;9 DASHES;12, 12, 14 AND 11 SPACES
47 110  VTAB 22: PRINT "PRESS RETURN TO CONTINUE":
       VTAB 23: PRINT "OR 'Q' TO QUIT";: GET A$:
       IF A$ = "Q" OR A$ = "q" THEN  HOME : END
D8 120  POKE  - 16368,0: HOME
32 130  PRINT  CHR$ (4)"BRUNHSF":X1 = 50:Y1 = 50:X
       2 = 220:Y2 = 110: HGR : HCOLOR= 3: HPLOT 0,
       0 TO 279,0 TO 279,159 TO 0,159 TO 0,0:M1 =
       1:M2 = 0:M3 =  - 1:M4 = 0
DB 140  HPLOT X1,Y1:X =  PEEK ( - 16336): HPLOT X2
       ,Y2:X =  PEEK ( - 16336):X =  PEEK ( - 1638
       4): IF X = 196 THEN M1 = 1:M2 = 0
D8 150  IF X = 215 THEN M2 =  - 1:M1 = 0
2F 160  IF X = 193 THEN M2 = 0:M1 =  - 1
4E 170  IF X = 216 THEN M2 = 1:M1 = 0
4B 180  IF X = 207 THEN M3 = 0:M4 =  - 1
44 190  IF X = 187 THEN M3 = 1:M4 = 0
84 200  IF X = 203 THEN M3 =  - 1:M4 = 0
81 210  IF X = 174 THEN M3 = 0:M4 = 1
39 220 X1 = X1 + M1:Y1 = Y1 + M2:X2 = X2 + M3:Y2 =
       Y2 + M4: & X1,Y1,A%: IF A% THEN 250
9D 230  & X2,Y2,A%: IF  NOT A% THEN 140
7E 240  VTAB 22: PRINT "PLAYER ON THE LEFT WINS.":
       POKE  - 16368,0: VTAB 23: PRINT "PRESS RET
       URN TO CONTINUE";: GET A$: POKE  - 16384,12
       8: POKE  - 16368,0: RUN
40 250  VTAB 22: PRINT "PLAYER ON THE RIGHT WINS."
       : POKE  - 16368,0: VTAB 23: PRINT "PRESS RE
       TURN TO CONTINUE";: GET A$: POKE  - 16384,1
       28: POKE  - 16368,0: RUN
```

TOTAL: AA4B

END OF LISTING 3