DOUBLE HI-RES GRAPHICS FOR THE APPLE II PLUS

DOS 3.3 O 0

Now you can get higher resolution graphics from your Apple II Plus. All you need is a high-resolution monochrome monitor and this short machine language routine.

ProDOS O 0

by Algis J. Matyckas, RD #3, Box 210, Washington, NJ 07882

You don't have to be left out of the double Hi-Res experience just because you own the Apple II Plus. You too can have double high resolution graphics. You can plot 560 points horizontally in two colors, black and white — that's twice the resolution provided by the 280 points plotted in normal Hi-Res mode.

First, a brief technical description of Hi-Res graphics. The II Plus has two blocks of memory reserved for Hi-Res graphics. The first block, Hi-Res page 1, is located in memory addresses $2000 to $3FFF. The second block, Hi-Res page 2, is located at addresses $4000 to $5FFF. Each page is made up of 192 rows, 40 bytes wide, and each byte is made up of 7 bits, each of which controls a point on the screen. (The eighth bit is not displayed.) So, 40 bytes times 7 bits equals 280 points horizontally. For a quick Hi-Res demonstration, type:

```
HGR: VTAB 24: CALL −151
<RETURN>
```

This will display Hi-Res page 1, move your cursor to the text window portion of the screen, and put you into the Monitor. To set the individual bits to plot points on the screen, type:

```
2000:01 <RETURN>
```

You will see a point in the upper-left corner of the screen that results from placing the value $01 in the first byte of Hi-Res screen memory. (In this Monitor command, the number 2000 specifies the memory address in hexadecimal, the one is the value of the byte to be written into this address, and the colon (:) is the Monitor command to write byte(s) to the specified location.) Now enter the following sequence:

```
2000:02 <RETURN>
2000:04 <RETURN>
2000:08 <RETURN>
2000:10 <RETURN>
2000:20 <RETURN>
2000:40 <RETURN>
```

If you were to repeat this sequence using the appropriate address for each of the 40 bytes, you would see all 280 points in the top row.

The eighth bit in each byte is called the color bit. Instead of controlling a point, it determines the color of the points plotted by that byte. You can use the color bit to create double Hi-Res graphics. Each of the first seven bits now represents two narrower points on the screen. If the color bit in the byte is clear (set to zero), then the left point is plotted for that bit. If the color bit is set (to one), then the right point is plotted. Now enter the following sequence and watch the display:

TABLE 1: Applesoft ROM Routines

Applesoft Routine	Function
CHKCOM	Checks for a comma at the location pointed to by TXTPTR.
TXTPTR	A pointer for the next character or token from a program.
COMBYTE	Checks for a comma and gets a byte in the X-Register (uses TXTPTR).
FRMNUM	Evaluates the expression pointed to by TXTPTR, puts the results into FAC, and makes sure it's a number.
FAC	Applesoft's main floating-point accumulator.
GETADR	Converts FAC into a two-byte integer and stores it in LINNUM.
HFNS	Gets and sets coordinates to be plotted in Hi-Res graphics. The program enters HFNS at $F6BF where it picks up the Y-coordinate.
HPLOT	Plots the point at the coordinate set by HFNS.
COLOR	The memory location in which the Hi-Res color byte is stored.
LINNUM	A two-byte location used in Applesoft as a general 16-bit number location.

```
2000:01 <RETURN>
2000:81 <RETURN>
2000:02 <RETURN>
2000:82 <RETURN>
2000:04 <RETURN>
2000:84 <RETURN>
2000:08 <RETURN>
2000:88 <RETURN>
2000:10 <RETURN>
2000:90 <RETURN>
2000:20 <RETURN>
2000:A0 <RETURN>
2000:40 <RETURN>
2000:C0 <RETURN>
```

If you were to repeat this sequence for 40 bytes, you would plot 560 distinct points.

There are two limitations to using the color bit to extend the graphics capability of the II Plus. First, color is dependent on location (this does not matter if you have a monochrome display), and second, all the bits of the byte are affected by changing the color bit. In other words, you can't plot points in positions one and two at the same time.

The machine language routine in **Listing 1** is a simple way to generate double Hi-Res graphics. It can be located anywhere in memory and is called from Applesoft with the command:

```
CALL D,C,X,Y
```

where *D* is the decimal address for the beginning of the machine language routine

(768 as listed here); *C* sets the color (0 for black and 1 for white); *X* is the X-coordinate, which ranges from 0 to 559; and *Y* is the Y-coordinate, which ranges from 0 to 191.

You can use either Hi-Res graphics page with this routine. **Table 1** shows the Applesoft routines that the program uses and their functions.

ENTERING THE PROGRAMS

Key in the program DOUBLE.HIRES as it is shown in **Listing 1** and save it on disk with the command:

BSAVE DOUBLE.HIRES,A$300,L$2E

Type in the Applesoft demonstration program DHR.DEMO from **Listing 2** and save it in with the command:

SAVE DHR.DEMO

For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

HOW IT WORKS

The program is well documented. When called, the routine first initializes the color byte to black (zero). It next uses COMBYTE to check the CALL statement for a comma and puts the color into the X-Register. If the color is white, the color byte is changed (set to 127 or color 3). The routine now needs to get the X-coordinate. CHKCOM is used to check for a comma. FRMNUM then gets the X-coordinate from the CALL statement and GETADR evaluates it and stores it in LINNUM. The Carry bit is then cleared and the X-coordinate divided by two. This is accomplished by rotating the byte in LINNUM to the right using the ROR instruction.

After the rotation, the Carry bit, which now contains the value previously in bit 0, is checked to see if the coordinate is odd or even. If the Carry is set, then the coordinate is odd and the sign bit of the color byte is set. If the Carry is clear, the X-coordinate is even and the sign bit of the color byte remains zero. The even coordinate plots the left half of the plotting bit, and the odd coordinate plots the right half. The X-coordinate in LINNUM is now in the range of the Applesoft HPLOT command. The routine enters the Applesoft routine, HFNS, where it gets the Y-coordinate and then HPLOTs to plot the point. Control is then returned to the BASIC program.

THE DEMONSTRATION PROGRAM

The demonstration program (**Listing 2**) is a collection of shapes drawn in double Hi-Res and standard Hi-Res graphics. The double Hi-Res routine is loaded at decimal location 768. We can see that the screen appears contracted in double Hi-Res mode. A 50-point by 50-point box looks like a rectangle, for instance, and a circle in Hi-Res looks like an oval. (To obtain more normal proportions, double the X-coordinates in double Hi-Res.) However, resolution is finer — a sine wave has more plotted points and looks smoother.

LISTING 1: DOUBLE.HIRES

```
 THE ASSEMBLER 1.Ø

SOURCE FILE - DOUBLE.HIRES

                       ;
                       ;************************
                       ;*     DOUBLE.HIRES     *
                       ;* BY ALGIS MATYCKAS    *
                       ;* COPYRIGHT (C) 1985   *
                       ;* BY MICROSPARC, INC   *
                       ;* CONCORD, MA  Ø1742   *
                       ;************************
                       ;
                       ;MICROSPARC ASSEMBLER SOURCE
                       ;
                       ;--- APPLESOFT ROUTINES ---
                       ;
                       CHKCOM   EQU $DEBE       ;CHECKS TXTPTR FOR COMMA
                       COMBYTE  EQU $E74C       ; GET A BYTE IN X REG
                       FRMNUM   EQU $DD67       ;EVALUATE EXPRESSION
                       GETADR   EQU $E752       ;CONVERT INTO INTEGER
                       HFNS2    EQU $F6BF       ; SET COORD.
                       HPLOT    EQU $F457       ;PLOTS A POINT AT COORD SET
                       COLOR    EQU $E4         ;HI-RES COLOR BYTE
                       LINNUM   EQU $5Ø         ;16 BIT NUMBER LOCATION
                       ;
                       ;ROUTINE CAN BE RELOCATED ANYWHERE IN MEMORY
                       ;
                                ORG $3ØØ
                       ;
                       ;--- GET COLOR (BLACK OR WHITE) ---
                       ;
Ø3ØØ  A9 ØØ                     LDA #$ØØ        ;INITIALIZE COLOR BYTE
Ø3Ø2  85 E4                     STA COLOR       ;WITH BLACK
Ø3Ø4  2Ø 4C E7                  JSR COMBYTE     ;GET COLOR FROM CALL
Ø3Ø7  EØ ØØ                     CPX #$ØØ        ;IS COLOR BLACK?
Ø3Ø9  FØ Ø4                     BEQ BLACK       ;THEN GO ON ELSE
Ø3ØB  A9 7F                     LDA #$7F        ;CHANGE COLOR BYTE
Ø3ØD  85 E4                     STA COLOR       ;TO WHITE
                       ;
                       ;
                       ;--- GET PLOTTING COORDINATES ---
                       ;
Ø3ØF  2Ø BE DE         BLACK    JSR CHKCOM      ;CHECK FOR COMMA AND
Ø312  2Ø 67 DD                  JSR FRMNUM      ;GET X COORDINATE
Ø315  2Ø 52 E7                  JSR GETADR      ;EVALUATE AND STORE IN LINNUM
Ø318  18                        CLC             ;CLEAR CARRY REG
Ø319  66 51                     ROR LINNUM+1    ;DIVIDE BY TWO
Ø31B  66 5Ø                     ROR LINNUM      ;AND CHECK IF EVEN
Ø31D  9Ø Ø8                     BCC EVEN        ;IF EVEN THEN PLOT
Ø31F  D8                        CLD             ;ELSE CLEAR FOR BINARY ADDITION
Ø32Ø  18                        CLC             ;CLEAR CARRY REG
Ø321  A5 E4                     LDA COLOR       ;LOAD COLOR BYTE
Ø323  69 8Ø                     ADC #$8Ø        ;SET SIGN BIT
Ø325  85 E4                     STA COLOR       ;AND STORE COLOR BYTE
                       ;
                       ;
                       ;--- PLOT POINT ---
                       ;
Ø327  2Ø BF F6         EVEN     JSR HFNS2       ;GET Y COORDINATE
Ø32A  2Ø 57 F4                  JSR HPLOT       ;PLOT THE POINT
Ø32D  6Ø                        RTS             ;RETURN

ØØØ  ERRORS

Ø3ØØ  HEX START OF OBJECT
Ø32D  HEX END OF OBJECT
ØØ2E  HEX LENGTH OF OBJECT
95AA  HEX END OF SYMBOLS
```

END OF LISTING 1

LISTING 2: DHR.DEMO

```
1   REM ***********************
2   REM *       DHR.DEMO      *
3   REM * BY ALGIS MATYCKAS   *
4   REM * COPYRIGHT (C) 1985  *
5   REM * BY MICROSPARC, INC  *
6   REM * CONCORD, MA  Ø1742  *
7   REM ***********************
5Ø  REM  INITIALIZE AND SET UP HGR SCREEN
6Ø  HOME : HGR : HCOLOR= 3: PRINT
7Ø  PRINT  CHR$ (4);"BLOAD DOUBLE.HIRES"
8Ø DHR = 768: REM  ADDRESS OF    DOUBLE HI-RE
     S ROUTINE
9Ø  HPLOT 14Ø,Ø TO 14Ø,159: HPLOT Ø,159 TO 27
     9,159
1ØØ  INVERSE : VTAB 21: PRINT "   DOUBLE HIRE
     S              HIRES"; TAB( 4Ø);" "
11Ø  VTAB 22: PRINT  TAB( 4Ø);" ";
12Ø  VTAB 23: PRINT  TAB( 4Ø);" ";: NORMAL
13Ø  REM  **** DRAW BOX ****
14Ø  VTAB 22: HTAB 15: PRINT "5Ø X 5Ø BOX"
15Ø  REM  DOUBLE HI-RES
16Ø  FOR X = 115 TO 165: CALL DHR,1,X,55: NEXT
17Ø  FOR Y = 55 TO 1Ø5: CALL DHR,1,165,Y: NEXT
```

LISTING 2: DHR.DEMO *(continued)*

```
18Ø  FOR X = 165 TO 115 STEP  - 1: CALL DHR,1
     ,X,1Ø5: NEXT
19Ø  FOR Y = 1Ø5 TO 55 STEP  - 1: CALL DHR,1,
     115,Y: NEXT
2ØØ  REM  STANDARD HI-RES
21Ø  FOR X = 185 TO 235: HPLOT X,55: NEXT
22Ø  FOR Y = 55 TO 1Ø5: HPLOT 235,Y: NEXT
23Ø  FOR X = 235 TO 185 STEP  - 1: HPLOT X,1Ø
     5: NEXT
24Ø  FOR Y = 1Ø5 TO 55 STEP  - 1: HPLOT 185,Y
     : NEXT
25Ø  GOSUB 63Ø: GOSUB 61Ø: REM   WAIT FOR KEY
     STROKE AND        DIVIDE SCREEN
26Ø  REM  **** DRAW CIRCLE ****
27Ø  VTAB 22: HTAB 12: PRINT "CIRCLE RADIUS 5
     Ø"
28Ø  REM  DOUBLE HI-RES
29Ø XC = 14Ø:YC = 8Ø:R = 5Ø:PA = Ø:PB = 6.283
     18:DP = .Ø174532778
3ØØ  FOR P = PA TO PB STEP DP:X = R *  COS (P
     ):Y = R *  SIN (P):X = XC + X:Y = Y + YC
     : CALL DHR,1,X,Y: NEXT
31Ø  REM  STANDARD HI-RES
32Ø  HCOLOR= 3
33Ø XC = 21Ø:YC = 8Ø:R = 5Ø:PA = Ø:PB = 6.283
     18:DP = .Ø174532778
34Ø  FOR P = PA TO PB STEP DP:X = R *  COS (P
     ):Y = R *  SIN (P):X = XC + X:Y = Y + YC
     : HPLOT X,Y: NEXT
35Ø  GOSUB 63Ø: GOSUB 61Ø
36Ø  REM  **** DRAW SINE WAVE ****
37Ø  VTAB 22: HTAB 1: INVERSE : PRINT  TAB( 1
     5);" ";: NORMAL : PRINT "SINE WAVE";: INVERSE
     : PRINT  TAB( 4Ø);" ";: NORMAL
38Ø  REM  DOUBLE HI-RES
39Ø  FOR A = Ø TO 278
4ØØ X = (A - 14Ø) / 38:Y =  SIN (X):YP = 96 -
     (Y * 3Ø): IF YP < Ø AND YP > 191 THEN 42
     Ø
41Ø  CALL DHR,1,A,YP
42Ø  NEXT
43Ø  REM  STANDARD HI-RES
44Ø  FOR A = 14Ø TO 278
45Ø X = (A - 21Ø) / 19:Y =  SIN (X):YP = 96 -
     (Y * 3Ø): IF YP < Ø AND YP > 191 THEN 47
     Ø
46Ø  HPLOT A,YP
47Ø  NEXT
48Ø  GOSUB 63Ø: GOSUB 61Ø
49Ø  REM  **** DRAW DIAGONAL ****
5ØØ  VTAB 22: INVERSE : PRINT  TAB( 8);" ";: NORMAL
     : PRINT "PARALLEL DIAGONAL LINES";: INVERSE
     : PRINT  TAB( 4Ø);" ";: NORMAL
51Ø  REM  DOUBLE HI-RES
52Ø  FOR X = Ø TO 159: CALL DHR,1,X,X: NEXT
53Ø  REM  STANDARD HI-RES
54Ø  HPLOT 14Ø,Ø TO 22Ø,159
55Ø  GOSUB 63Ø
56Ø  REM  INSTRUCTIONS *****
57Ø  TEXT : HOME : PRINT  TAB( 14);"DOUBLE HI
     RES": VTAB 3: PRINT "CALL DHR,C,X,Y": PRINT
     : PRINT " DHR=DECIMAL LOCATION OF DOUBLE
      HI-RES         ROUTINE"
58Ø  PRINT : PRINT " C=COLOR (Ø=BLACK,1=WHITE
     )": PRINT : PRINT " X=X COORD. RANGE (Ø
     TO 559)": PRINT : PRINT " Y=Y COORD. RAN
     GE (Ø TO 191)": VTAB 2Ø: PRINT " END OF
     DEMO"
59Ø  END
6ØØ  REM  SUBROUTINE TO DIVIDE   SCREEN
61Ø  HGR : HCOLOR= 3: HPLOT 14Ø,Ø TO 14Ø,159:
      HPLOT Ø,159 TO 279,159: RETURN
62Ø  REM  SUBROUTINE TO WAIT FOR RETURN TO BE
      PRESSED
63Ø  VTAB 24: PRINT  TAB( 13);"PRESS <RETURN>
     ";
64Ø X =  PEEK ( - 16384): IF X < 128 THEN 64Ø
65Ø  POKE  - 16368,Ø
66Ø  IF X <  > 141 THEN 64Ø
67Ø  VTAB 24: HTAB 1: CALL  - 958
68Ø  RETURN
```

END OF LISTING 2