Drawpic

Artworx Software Company
150 North Main Street • Fairport, New York 14450

DRAWPIC

written by Dennis Zander
(C) 1982 by Artworx Software Company, Inc.

THIS PROGRAM REQUIRES:

BASIC cartridge
One joystick
16K RAM w/cassette
32K RAM w/disk

INTRODUCTION

DRAWPIC provides you with the ability to create detailed and colorful graphic images which can be easily incorporated into your own programs. After you create an image using the various options of DRAWPIC, a machine language routine will save the bytes which represent the image and store them as part of the program in an ASCII string subroutine (image string). This subroutine can then be called at a later time and another machine language routine instantly will put the image onto the screen at a point selected with the joystick controlled cursor. Smaller images can be added together to make larger ones, and size can be changed by using a different compatible graphics mode (4 & 6 are compatible - 3,5,7 are compatible). When you are finished, the program and all the images are saved by simply pressing 'Q' for Quit. The image strings and machine language routines can then be used in your own program for some potentially spectacular results. On the disk version, images may be saved to binary files on cassette or disk, and font files may be loaded and the drawing modes used to edit the characters.

GETTING STARTED

Plug your joystick into slot #1 and CLOAD and RUN the cassette; the disk version will autoload. After the program displays the title page, let the program set for a minute and you will see a demonstration of how rapidly images can be put onto the screen. Now go ahead and press any key to start the actual program. You will be in the Point Plot mode in Graphics mode 3. A text window shows you the Graphics Mode, Color, Mode (PLOT or DRAW), free memory in BYTES, and your X,Y coordinates for that Graphics Mode. Your cursor is a little flashing dot in the upper lefthand corner (it starts at X=0,Y=0). The position of this cursor is changed by moving the joystick. Plot or draw functions are activated by using the trigger button. Since you are in the PLOT mode, each time that the trigger is pressed, a single point is plotted on the screen. If you hold the trigger down and move the cursor around you will be able to draw freehand.

You may move in greater jumps by pressing the cursor control keys. The cursor will move in 1/4 screen steps in the direction of the arrow on the key.

With the disk version of DRAWPIC, the entire screen area may be used by pressing the START button. This will eliminate the text window. Pressing SELECT will return the normal text window display. Because of this, you may occasionally lose your cursor behind the text window, so check the Y value if you cannot find your cursor!

To get into the DRAW mode, press 'D'; place the cursor and press the trigger to fix a start point and then move the cursor and press the trigger a second time; a line will then be drawn from the start point to the finish. Now try rubber band mode, press 'R'; the first time that you press the trigger you will nail down one end of a rubber band line whose color is set by the color register you are using (keys 0-3 select color registers). By moving the cursor you will see that a line is drawn and repeatedly updated to stay between the start point and the cursor. Also note that if the line crosses something previously drawn, it will erase it, so be careful. Now try the rubber band fill. If you haven't done so already, press the trigger so that you leave your line and can move just the cursor around. Press down the trigger and hold it down, then move the cursor in the increasing X direction then the increasing Y direction. You will immediately see why this is called rubber band fill (or erase if you select color register 0, the background). If you release the trigger you are in plain rubber band mode and then pressing the trigger one more time will free the cursor.

When drawing in graphic modes 3, 5 or 7, you can use any of three colors simply by pressing the keys '1', '2' or '3'. The fourth color, color 0, is actually background and is used for "erasing." Orange, green and blue are the default colors of the '1', '2' and '3' keys respectively. Use these keys as you would a paint bucket for changing colors. You may change the color of "paint" which you are using, but you will also change the "paint" already present on the screen. These colors are actually set by the values stored in a color register.

To change the hue and luminance of a color register, as SETCOLOR does in BASIC, press 'C' and then select the color register to be changed. You can change colors by using the joystick and following the directions indicated on the screen. Pressing the trigger stores the color information (although it's not permanently stored until the image is SAVEd) and returns you to the drawing mode.

Since we have brought up the topic of storing images, let's store one of your own.

Press 'S' (and in the disk version, press 'S' again to save to a string array), now move the cursor to the upper-left corner of the area to be stored and press the trigger; then move the cursor to the lower-right corner of the area to be stored. When you press the trigger, the bytes included in the rectangle just specified will be written to a string and saved with the program. You will be required to specify an image number and name; this permits easier extraction of the image later.

You can view all stored images by pressing 'V' followed by the image number which you want to start with. This will clear the screen and then display each stored image until any key is pressed. Note an image and it's identifying number and then press 'L' (followed by 'S' for string array in the disk version); this loads an image. After you type in the image number, answer whether or not you want the screen cleared

of existing images. If you do not erase, then the loaded image will be drawn in the current graphics mode regardless of the mode in which the image was originally drawn. Position the cursor where you would like the upper left hand corner of the image. Each time you press the trigger, the image will be put on the screen. To exit image loading just press any key and you will return to normal drawing mode.

Once you have an image on screen, it or any portion of it may be replicated by using the MOVE command. Press 'M' then specify the start and end bytes as in a normal SAVE. You will immediately be asked for a new start byte just as when you are loading images to the screen. The image will be replicated at a new location each time you press the trigger. You may exit by pressing any key.

We've talked a lot about creating, now let's discuss erasing. To get a clean screen, you can do a shift-CLEAR; to erase just a point or line you can use the appropriate draw mode with color '0' (background). Stored images can be deleted by pressing shift-DELETE. Once you specify a starting image, it and all subsequent images will be deleted.

IMPORTANT! So far everything has been done with information stored in the computer. To have images permanently stored on cassette or disk, first save the image using the 'S' option and then press 'Q' for quit in order to save the program and all new images to disk.

With the disk version of DRAWPIC, you may also save and load binary image files to and from disk and cassette. The procedure is similar to that for strings except faster! The file format is explained in the addendum. After pressing 'S' or 'L' you will have to respond with 'S' for string save (as above), 'C' for cassette save or 'D' for disk. You may also respond with 'F' for a font load/save. That is covered in more detail in the addendum. If you select 'D', then input the file name ('D:' is optional). Start and end bytes will be requested as in string saves, and just the start byte for a load.

You can examine the disk directory by pressing 'X', the first 6 entries will be displayed and 6 more each time that you press SELECT. Pressing START will return you to the normal text display.

The DRAWPIC instruction set is summarized in Appendix 1.

IMAGES IN YOUR PROGRAM

The program listing in Appendix 2 shows the code necessary to make use of your created images. Lines 10-16 set things up. Line 16 sets colors after getting them from the subroutine call in line 14. Lines 5014-5199 may be listed to cassette or disk directly from DRAWPIC.

The subroutine at 5014 must be called after setting the graphics mode (since line 5100 calculates the start of screen data (DAT) which will be different for different graphics modes). Line 5110 calculates the address that the first byte of an image will be put based on the X,Y position specified. Constants LC and PX are evaluated in line 11 but must be set as indicated in the REMs of Appendix 2. Line 5014 is the machine language subroutine for putting images in screen memory per the call in the subroutine at 5199:

```
A = USR(PICT,STRT,ADR(D$),BYT,LIN,LC)
PICT = Address of machine language subroutine
STRT = Starting address of image
ADR(D$) = Address of data for image
BYT = Image width in bytes
LIN = Image length in bytes
LC = # bytes per line in this graphics mode
```

Lines starting at 21,000 are your image(s). They can be LISTed out of DRAWPIC and ENTERed into your program. If your image lines can easily be listed to the screen with room to spare (after you press 'Q' for a permanent save), then you can load your program with CLOAD or LOAD "D:PROGRAM" then move the cursor to the first line of image data and press return for each line of the image to add it to your program. If your image is too big for this, then you will have to 'LIST' the lines to cassette or disk and 'ENTER' them after loading your program.

For example, if your image is stored in lines 21100 to 21115, then type: LIST "D:IMAGE.LST",21100,21115. This will create a disk file containing your image. If you have a cassette recorder, then insert a data tape and type: LIST"C:",21100,21115. This will store the image onto tape. To enter the image into your own program, type: ENTER"D:IMAGE.LST" from disk, or rewind your data tape and: ENTER"C:" from cassette. Make sure that you include all of the lines for your image including the one with the REM and image name.

Lines may be automatically listed to disk or cassette by using the 'W' or WRITE command. Here you specify the consecutive image numbers you wish to have included. The program will automatically calculate the line numbers and write them to the file indicated (for example, C: or D:IMAGE.LST) on cassette or disk.

ADDITIONAL USEFUL INFORMATION

In order to minimize the number of bytes required to store images, only the bytes necessary are stored as indicated by the start and stop points you set when doing a STORE. Now each byte in Modes 3,5,and 7 stores 4 graphics points (pixels) with 2 bits per pixel. This is why you can have four colors in these modes, counting the background of course. In graphics modes 4 and 6 there are 8 pixels per byte and only color 1 or background!

Therefore, mode 7 has 40 bytes for 160 pixels per line and mode 6 has 20 bytes for 160 pixels per line, (LC = 40 and LC = 20 respectively), and PX = 4 pixels per byte for graphics mode 7, and PX = 8 pixels per byte for graphics mode 6. These proportions hold for the other modes as well. This means that for graphics mode 7, the pixels at X = 0-3 are in byte 1 and X coordinates 4-7 are in byte 2:

PIXEL, X =	0	1	2	3	4	5	6	7
BITS	00	01	01	10	10	00	11	11
COLOR REG	0	1	1	2	2	0	3	3
BYTE #	1				2			

In order to get images to butt together without any gaps, it is necessary to start at the beginning of a byte (any multiple of 4 for odd # modes and 8 for even # modes) and to finish at the end of a byte (a multiple of 4 or 8 minus 1). For instance, X = 8 to X = 31 will be exactly 3 bytes wide.

This brings up an important point. As seen in the string starting at line 21000 of the listing in Appendix 2, the graphics mode that the image was drawn in is stored GR = 3, however, when it was displayed it was done in graphics 7, see line 11. Does that give you any ideas?

DRAWPIC ADDENDUM

After considerable use, several DRAWPIC users requested a method for fast storage and retrieval of files similar to those in the Artworx HODGE PODGE program. The lines from 3600-3960 implement this. F$ contains the file name and K$ will indicate L-load or S-save. I$ has the device to be used. Lines 3840, 3850 use machine language routines defined in lines 5011, 5012. In line 3920 another routine is used: WORD. These routines are explained more fully later.

BINARY FILE FORMAT

Both disk and cassette files have the same binary file format:

BYT,LIN,DBYTE1,DBYTE2,DBYTE3,....,COLOR 0,COLOR1,....COLOR4

1st byte - width of image in bytes
2nd byte - length of image in lines
3rd-Nth byte - data bytes (Number=BYT*LIN)
LAST 5 bytes - color register values 0-4

LOADING BINARY FILES FROM YOUR PROGRAM

The binary files described above can be loaded and displayed in two ways. They can be loaded into a string (like 'D$') and then moved to the screen using the PICT subroutine, or they can be loaded directly to the screen if they are full screen width images. To load binary files, you will need lines 3900 to 3960, the WORD subroutine, and lines 5013 and 5014. These are all listed in the demo in Appendix 3. If loading from cassette, you will also need to let AUX2=128. F$ will be the file to be loaded and should have the following form: 'D:PROGRAM.EXT' for disk; 'C:' for cassette. If K$="L" then the file data will be loaded into memory starting at address BFA. The first two bytes in the file are BYT,LIN; they will be loaded first and used to calculate the number of data bytes to be loaded (BLEN) according to BLEN=BYT*LIN+5. The additional 5 bytes loaded are the values for the color registers and shouldn't be loaded if the address BFA is the first byte of screen memory. If loading to a string (i.e., BFA=ADR(D$)) then make sure that D$ is dimensioned for the largest image plus 5 bytes for color information.

DIRECT METHOD: Full width images, even if they are not full height, can be loaded directly to the screen very easily. After determining the screen address, set BFA equal to the screen address plus 40 * # OF LINES FROM TOP. Now when you GOSUB 3900, the file data will be loaded directly to the screen. The calculation for BLEN in line 3912 of Appendix 3 should not have the +5 (so that the color bytes will not be put onto the screen). Line 3945 should also be disabled.

D$ METHOD: All images may be loaded this way, full screen width or not. First, set up and dimension an array D$ to be BLEN bytes long. Then let D$="", and BFA=ADR(D$). When the file data is loaded during the GOSUB 3900, it will load into D$. The start address for displaying the image must be calculated by setting X,Y and a GOSUB 5100. Then GOSUB 5199, which uses the PICT subroutine to put the image on the screen at STRT. Line 3945 will set the color registers; disable that line if you do not want them changed. The advantage of this method in addition to loading partial images is that since D$ still contains the image, you can re-use it anytime. The disadvantage is that you need the additional memory required for storing D$, which could be nearly 4K bytes for a full mode 7 image.

FONT FILE FORMAT

Character sets or 'fonts' may be loaded and modified or a complete image can be drawn and saved as a character set. Programs such as INSTEDIT provide an easy method to incorporate this new character set into your program. DRAWPIC will not check for duplicate characters; that must be done by the user.

ex. A GROUP OF 3X2 CHARACTERS:

	COL. 1	COL. 2	COL. 3
ROW 1	BYTE 1	BYTE 9	BYTE 17
	BYTE 2	BYTE 10	BYTE 18
	BYTE 3	BYTE 11	BYTE 19
	BYTE 4	BYTE 12	BYTE 20
	BYTE 5	BYTE 13	BYTE 21
	BYTE 6	BYTE 14	BYTE 22
	BYTE 7	BYTE 15	BYTE 23
	BYTE 8	BYTE 16	BYTE 24
ROW 2	BYTE 25	BYTE 33	BYTE 41
	BYTE 26	BYTE 34	BYTE 42
	BYTE 27	BYTE 35	BYTE 43
	BYTE 28	BYTE 36	BYTE 44
	BYTE 29	BYTE 37	BYTE 45
	BYTE 30	BYTE 38	BYTE 46
	BYTE 31	BYTE 39	BYTE 47
	BYTE 32	BYTE 40	BYTE 48

Data is stored 8 bytes per row, column by column. Regardless of the number of rows and columns, 1024 bytes are saved. The number of characters displayed will be N= #ROWS * #COLS. If that is less than 128 then the first N characters will be displayed.

Typical Column, Row values:

GR.7	32,4 or 16,8
GR.6	16,8 or less
GR.5	16,8 or less

By using the font load and save mode which is compatible with nearly all character editors (especially INSTEDIT), you can draw images and then save them to a character set. The ATARI font file 'ATARI.SET' can be loaded into graphics mode 4 or 6. Try loading ATARI.SET into graphics 6 with 16 columns and 8 rows, or DEMO.SET into graphics 7 with 32 columns and 4 rows. Other modes will make that set look strange. A multi-color set can be loaded or created in graphics modes modes 3,5, or 7. These multi-color modes are demonstrated and explained in INSTEDIT. You can draw an image and then using the MOVE command put it anywhere in the character set. The ATARI Technical Reference Manual explains how character color data is taken from each byte of a character. As it turns out, it is the same for Antic modes 4 and 5 as for BASIC graphics modes 3,5,7.

MACHINE LANGUAGE ROUTINES

WORD: The routine in 5013 will get a word (2 bytes) from memory or put a word into memory:

```
A=USR(WORD,ADDRESS,[VALUE])
```

where
WORD is the address of the routine
ADDRESS is address to be stored to or loaded from
VALUE is the two byte value (address) put into ADDRESS.

If VALUE is not used, i.e. DLIST=USR(WORD,560), then DLIST will be the address contained in 560,561. If VALUE is used, i.e. A=USR(WORD,88,NEWSCREEN), then the address NEWSCREEN will be put into 88,89 all at machine speed!

CHRL: This routine takes a character set stored in a buffer, BFA (like D$), and puts it on the screen starting at STRT.

```
A=USR(CHRL,STRT,BFA,COL,ROW,LC)
```

where
CHRL is the address or the routine.
STRT is the byte to start loading into on the screen.
BFA is the address of the data was loaded into.
COL is the # of columns to be used.
ROW is the # of rows to be used.
LC is the # of byte in each line.

CHRS: This is similar to CHRL except that it saves a character set to a buffer to be output to disk.

SPECIAL NOTE!!! Whenever using lines from DRAWPIC be sure to equate variables such as Q1, Q2, Q3 etc. as given in lines 2 and 3.

APPENDIX 1.

THESE INSTRUCTIONS APPLY TO BOTH CASSETTE AND DISK VERSIONS

G - GRAPHICS MODE, Selects from graphics modes 3-7.

D - DRAW LINE, Draws a line between two selected points.

R - RUBBER BAND, Draws a line continuously from a start point to the current cursor position and leaves all lines on the screen if trigger is held down (rubber band fill).

P - POINT PLOT, A point is plotted on the screen each time that the trigger button is pressed.

C - COLOR SET, The hue and luminesence of the selected color register are set using the joystick. Mode is terminated when any key is pressed.

S - STORE PICTURE, Use to store entire screen or partial images. Set start point in upper-left corner and end point in lower-right corner. Image is saved as a string array. Line number starts at:

    21000 + N * 100
    where N is the image number. (N = 2, Line = 21200)

L - LOAD PICTURE, After image number is specified, the image can be placed on the screen as many times as desired at the cursor location each time the trigger is pressed.

V - VIEW STORED PICTURES, Clears screen and puts stored images on screen one at a time starting at the number input until the last picture or until a key is pressed. (Good for finding an image).

0 - 3, Selects color register, pick the color to draw with; color 0 is background and is used for erasing.

shift CLEAR, Clear screen.

shift DELETE, Delete stored pictures, all images starting at the one indicated are erased from memory.

Q - QUIT, Saves the current program and all images to disk or cassette. (This is the only permanent storage).

control ↑ - Move cursor up 1/4 screen.
control ↓ - Move cursor down 1/4 screen.
control ← - Move cursor left 1/4 screen.
control → - Move cursor right 1/4 screen.

NOTE: arrow keys are not operative during LOAD, SAVE and block MOVE.

M - Move and copy a block of bytes to a new place on the screen.

W - Write a LIST file for the first to last images indicated. This will be in LIST format and may be ENTERed into your program.

INSTRUCTIONS FOR DISK VERSION ONLY

START - turns off the text window for full screen drawing
SELECT - returns normal text window display
X - eXamine disk directory and display in text window
    SELECT - get the next 6 entries
    START - return to normal text display
S - Save to device or L - Load from device
    RESPOND WITH:
    C - cassette binary file
    D - disk binary file
    S - string of characters to be stored in DRAWPIC
    F - font file (Loaded from disk only)

APPENDIX 2: DRAWPIC DEMONSTRATION PROGRAM

```
0 REM **DRAWPIC DEMO** (C)1982 ARTWORX
1 REM by Dennis Zander
2 REM FOR YOUR OWN PROGRAM YOU ONLY
3 REM NEEED LINES 10-5199 AND YOUR
4 REM STRING ARRAY IMAGES (LIKE 21200-21204).
5 REM THE GOSUB IN LINE 14 MUST HAVE THE SAME LINE #
6 REM AS YOUR DATA SUBR.  ADJ. AS NEEDED:
7 REM FOR GR.3 LC=10 PX=4; GR.4 LC=10 PX=8; GR.5 LC=20 PX=4
8 REM & GR.6 LC=20 PX=8; GR.7 LC=40 PX=4
10 DIM D$(100),C(4):REM D$ must be DIM for the largest image you will use.
11 GRAPHICS 7:PX=4:LC=40:X=72:Y=32
12 GOSUB 5013:REM TO FIND SCREEN,ETC.
14 GOSUB 21200:REM OR YOUR SUBR LINE#
16 FOR I=0 TO 4:POKE 708+I,C(I):NEXT I:REM SET PER STORED COLORS
18 GOSUB 5199:REM DRAW (1) PICTURE!
100 GOTO 30000:REM YOUR PROGRAM HERE.
5013 WORD=ADR("h(hQhPphPhP`1PTH1PU`"):REM **Enter line 5013 of DRAWPIC**
ZMeQM{fNJP[`"):REM **Enter line 5014 of DRAWPIC**
5100 DAT=USR(WORD,88):REM or skip line 5013 and use: DAT=PEEK(88)+256*PEEK(89)
5110 STRT=DAT+Y*LC+INT(X/PX):RETURN
5199 GOSUB 5110:A=USR(PICT,STRT,ADR(D$),BYT,LIN,LC):RETURN
21200 D$="P_U?y3¦~PZ¦~¦~P?z3_YX":REM SEE BELOW
21201 REM D$="ctrl(,) ctrl(U) P ctrl(,) ctrl(A) shift(-) inv(U) ctrl(,) ctrl(E) inv(?) inv(y) 3 ctrl(V)
21202 REM inv-shft(=) esc-ctrl(del) P Z inv-shft(=) esc-ctrl(del) inv-ctrl(T) ctrl(V) inv-shft(=) esc-ctrl(del)
21203 REM P ctrl(E) inv(?) inv(z) 3 ctrl(A) shft(-) inv(Y) 2-ctrl(,) ctrl(U) X 7-ctrl(,) ctrl(H) ctrl(,)"
21204 BYT=4:LIN=11:GR=3:C(0)=246:C(1)=62:C(2)=148:C(3)=70:C(4)=192:RETURN
30000 ?
30020 TRAP 30030:? "OLD X:";X;"  NEW X    ";:INPUT X:IF X)179-BYT THEN ? ")TOO BIG!!!":GOTO 20
30030 TRAP 30040:? "OLD Y:";Y;"  NEW Y    ";:INPUT Y:IF Y)79-LIN THEN ? ")TOO BIG!!!":GOTO 30
30040 TRAP 30050:N=1:? "HOW MANY";:INPUT N
30050 FOR J=1 TO N
30060 GOSUB 5199:REM PUT PICT. ON SCREEN
30070 FOR I=20 TO 0 STEP -1:SOUND 0,0,2,I:NEXT I:REM MAKE A NICE SOUND!
30080 X=X+4*BYT:NEXT J
30090 ? ")":GOTO 30020:REM DO IT AGAIN?
```

APPENDIX 3: BINARY FILE LOADING DEMO

```
0 REM ***LOADER.DEM***
1 REM ***by Dennis Zander
2 REM
3 REM ___THIS LOADS A BINARY FILE DIRECT TO SCREEN IF BYT=40. IF NOT IT MUST LOAD TO D$ FIRST.
4 REM ___TO ALWAYS LOAD TO D$ FIRST, DELETE LINE 3915, AND LET BFA=ADR(D$).
5 REM ___IF YOU ALWAYS LOAD AN IMAGE WITH BYT=40 THEN YOU MAY DELETE D$, LET BFA=STRT, AND SAVE THE SPACE OF D$.
6 REM ___FOR YOUR OWN PROGRAM YOU ONLY NEED L. 3900-5199, BUT SOMEPLACE YOU MUST DO WHAT'S DONE IN L. 10-30.
7 REM
10 GRAPHICS 7:LC=40:PX=4:REM SET GR. MODE OF IMAGE TO BE LOADED, LC-BYT/LIN, PX-PIXELS/BYT
15 X=0:Y=0:REM SET UPPER LEFT CORNER OR IMAGE.
20 BUF=3200:Z=6:REM SET SIZE= LARGEST IMAGE.
25 GOSUB 5000:REM FIND SCREEN, ETC.
30 F$="D:TITLE.BIN":REM SET IMAGE NAME
40 TRAP 200:GOSUB 3900:REM LOAD IMAGE TO D$
50 GOSUB 5199:REM PUT IMAGE ON SCREEN
100 ? ")  NEXT FILE TO LOAD";:INPUT F$:GOTO 40
200 REM IN CASE F$ DOESN'T EXIST
205 STAT=PEEK(IOCB+3):IF STAT)127 THEN ? "ERROR ";STAT:FOR I=1 TO 200:SOUND 0,0,2,1:NEXT I:SOUND 0,0,0,0
210 POP :CLOSE #1:GOTO 100
1000 REM **THIS SUBR. WILL LOAD IMAGES**
3900 WR=4:SIO=7:AUX2=0
3910 OPEN #1,WR,AUX2,F$
3912 GET #1,BYT:GET #1,LIN:BLEN=BYT*LIN+5
3915 BFA=ADR(D$):IF BYT=40 THEN BFA=STRT
3920 IOCB=832+16:POKE IOCB+2,SIO:A=USR(WORD,IOCB+4,BFA):A=USR(WORD,IOCB+8,BLEN)
3940 I=USR(ADR("hhh*LVd"),16)
3945 FOR I=0 TO 4:C(I)=PEEK(BFA+BLEN-5+I):POKE 708+I,C(I):NEXT I
3960 CLOSE #1:RETURN
5000 DIM D$(BUF),F$(18),C(4):X=0:Y=0:LX=0:LY=0:CV=1:D$=" ":BFA=ADR(D$)
5013 REM INSERT LINE 5013 FROM DRAWPIC
5014 REM INSERT LINE 5014 FROM DRAWPIC
5100 DAT=USR(WORD,88)
5110 STRT=DAT+Y*LC+INT(X/PX):RETURN
5199 GOSUB 5110:A=USR(PICT,STRT,BFA,BYT,LIN,LC):RETURN :REM ***THIS PUTS IMAGES ON SCREEN FROM BFA***
```