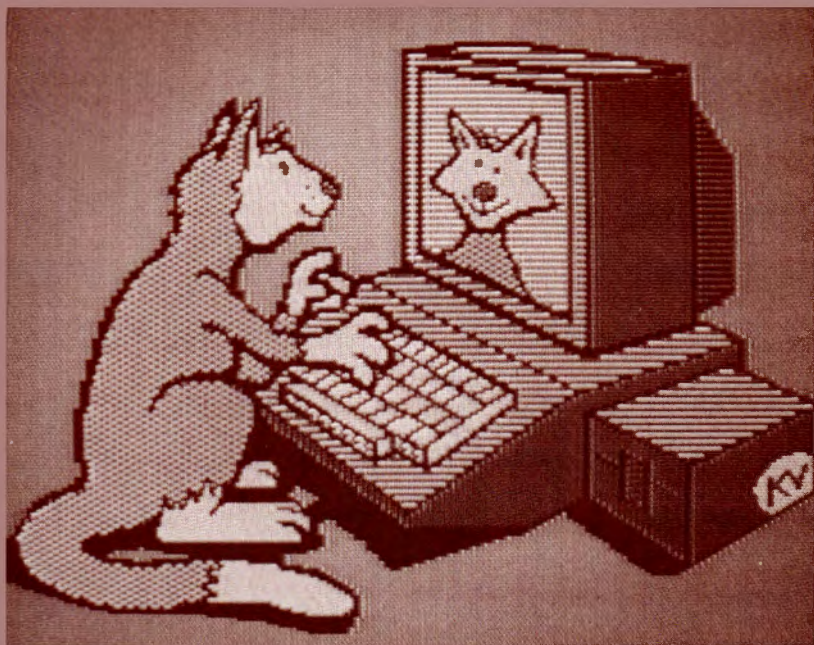# CAT

GRAPHICS™

by David Shapiro

Adds Dozens of New Graphics Commands to BASIC

penguin software™
the graphics people

CAT GRAPHICS ™

by David Shapiro

The enclosed product is supplied on a disk that is NOT copy-protected. It is our intent to make this product as useful as possible, and we feel that with applications software the ability to make your own backup copies is a great asset. We ask that you not abuse our intentions by making copies for others, and by not copying this manual. The result of such activity only helps promote the unfortunate existing situation of most software being protected, and, in our opinions, less usable. We hope that the commercial success of non-protected products such as this one will signal other publishers that copy-protection is not necessary for a product's survival and help reverse the trend in protected applications software. At Penguin, through policies such as this and our pricing, we are trying to look out for your best interests. Please help us.

We hope you find many hours of enjoyment and use in this package.

Sincerely,

Mark Pelczarski
President, Penguin Software

49887

penguin software ™
the graphics people

P.O. Box 311, Geneva, IL 60134 (312) 232-1984

This package is dedicated to
Catherine Ann Francis,
a very special person.

Table of Contents

I. Introduction

Cat Graphics is a collection of routines to make it easier to design and use graphics on the Apple. The package adds 55 commands to Applesoft Basic, and includes programs to help you design shapes, fonts, and sound effects. There's also an "Electric Fire" program for your entertainment.

Cat Graphics also supports the new double res graphics mode available on the //e and //c. To use this version with a //e, you need to have the extended version of the 80 column card, and your motherboard must be revision B or later. There is also a jumper that needs to be installed on the 80 column card. If you need help with this, or if you're not sure if your Apple can display double res graphics, you should consult your dealer.

The double res graphics mode gives a resolution of 560 by 192 dots, and gives you a choice of 16 base colors to work with. If you want to use all of the colors, however, your effective horizontal resolution is lessened, as with normal Apple graphics. Due to the extra colors, the double res version of **Cat Graphics** has 256 blended color patterns, as opposed to 108 in the single res version. Most of the commands work the same way in both versions; any differences are noted in the descriptions of the relevant commands.

II. Getting Started

If you want to use **Cat Graphics** with your own Basic programs, all you have to do is transfer the binary file GR1 to the disk with your program on it. For instructions on transferring various files from the **Cat Graphics** disk, see Appendix A. You can initialize **Cat Graphics** by typing "BRUN GR1" before running your program. This loads the routines just below DOS, and reserves the space so that they won't be overwritten. If you want to do everything in one step, you can enter the following short program:

```
10 D$=CHR$(4)
20 PRINT D$;"OPEN LOADER"
30 PRINT D$;WRITE LOADER"
40 PRINT "BRUN GR1"
50 PRINT "RUN YOUR PROGRAM"
60 PRINT D$;"CLOSE LOADER"
```

When you run this program, it will create a text file named LOADER. Now to initialize **Cat Graphics** and run your program, you can simply type "EXEC LOADER". If you want to have your program run automatically when you boot your disk, you can use the following line as your HELLO program:

```
10 PRINT CHR$(4); "EXEC LOADER"
```

Note that to initialize **Cat Graphics,** you must either type the "BRUN GR1" command from the keyboard, or put it in a text file as in the above example. The reason for this is that when **Cat Graphics** is initialized, it erases any Basic program currently in memory. So if you try to have a Basic program issue the BRUN command, the program will destroy itself.

There is another version of **Cat Graphics** in the file GR2. This version will only work on 64K Apples. It uses the same memory that Integer Basic and Pascal are loaded into, and it leaves you more room for Basic programs. The double res version of **Cat Graphics** is in file GR5, and uses the same area of memory. There is also a program that lets you create a copy of the single res version of **Cat Graphics** that resides at the bottom of memory, and allows you to eliminate the commands you don't need to save space. For more information on the various versions, see Appendix B.

III. Using the ampersand routines

Once **Cat Graphics** has been loaded, 55 new commands are available in Basic. All of the commands begin with the ampersand character (&), which is set aside in Applesoft to allow extensions to the language.

To learn how the commands work, it may be a good idea to try some of them out as you read the descriptions of what they do. Boot the **Cat Graphics** disk and press **0** to exit to Basic. Now you can start experimenting. You may also want to list the menu program to see some examples of how to use the new commands.

Explanations of each command and how to use it follow. There is also an alphabetical list of all the commands in Appendix D.

The routines take two kinds of parameters. Where it says "var", the name of a legal Applesoft variable must be specified to store the result of the operation. All other parameters may be either a number, a variable, or a valid Applesoft expression. The one exception is the &LOAD command, which requires a name in quotes as one of its parameters.

Drawing commands

&DOT,x,y

Plots a dot at the coordinate x,y. The coordinates in single res can range from 0-279 for x, 0-191 for y.

In the double res version, x coordinates can range from 0-139, 0-279, or 0-559, depending on how you use the &RES command. The starting range is 0-139, since that is the maximum resolution when using color. You may want to use the 0-279 range if you're writing programs that are intended for use in both single and double res. The 0-559 range is best to use when you're working in black and white only. Whatever coordinate range you select, it will be used for all commands that require you to specify a position on the screen. These commands are: &DOT, &LINE, &BOX, &CIRCLE, &CURVE, &SHAPE, &FILL, &PENGUIN, &MAG, and &CHKDOT.

The dot routine uses the values set by the &MODE and &COLOR commands to determine the way to plot the dot and the color to use. The defaults are plot mode and white. The mode and color values also control the &LINE, &BOX, &CIRCLE, &CURVE, &SHAPE, and &PENGUIN commands.

Example: &DOT,10,20   plots a dot at 10,20.

&LINE,x1,y1,x2,y2

Draws a line from coordinates x1,y1 to x2,y2. This command uses the current mode and color, and works identically in single and double res.

Example: X=50:&LINE,X,30,X,170   draws a horizontal line from 50,30 to 50,170

&BOX,x1,y1,x2,y2

Draws a box with corners at x1,y1 and x2,y2. This command uses the current mode and color, and works identically in single and double res.

Example: NEW
    10 INPUT "Please enter two numbers:";A,B
    20 &BOX,A,B,A+20,B+20
    RUN

This short program will ask you to type in two numbers, and will then plot a 20 by 20 box with its upper left corner at those coordinates.

&CIRCLE,x,y,radius

Draws a circle with the specified radius, with its center at x,y. This command uses the current mode and color, and works identically in single and double res.

Example: FOR N=1 TO 15:&CIRCLE,8*N+10,5*N+32,13:NEXT

This example will plot a series of 15 overlapping circles with radii of 13.

&CURVE,x1,y1,x2,y2,a,b

Draws a curve from x1,y1 to x2,y2. A and b are numbers that control the shape of the curve, and should range from -127 to +127. This command uses the current mode and color, and works identically in single and double res.

Example: &CURVE,30,70,120,95,46,20 draws a curve from 30,70 to 129,95.

&SHAPE,n,x,y

Draws the nth shape in the currently active table on the screen at x,y. These have nothing to do with standard Applesoft shape tables, which are stored in a different format. This command is intended to be used with shapes that are to be drawn in a single color only. The next command described, &BSHAPE, can be used to draw multicolored shapes, but it does have some limitations that the shape command doesn't. In the double res version of **Cat Graphics,** however, the &SHAPE command can be used to plot multicolored shapes, and the &BSHAPE command is not available. Another difference in double res mode is that shapes will normally appear either half as wide or twice as wide as in single res. To see how they will look while editing, you should use the double res version of the shape editor. If you want shapes to look the same in double res, you can make them by issuing the commands &THIN:&SHAPEMODE,1. The same applies to fonts, only you should use &CHARMODE instead of &SHAPEMODE. These commands are explained in the next section.

Note that for &SHAPE and all other table based routines, things are numbered starting at zero. So if you had a table with five shapes in it, they would be numbered 0 through 4. This command uses the current mode and color.

Example: &SHAPE,0,60,30 plots first shape in current table at 60,30.

To try this example, you'll have to draw some shapes with the shape editor and use the &LOAD command (described later) to load them in. If no shapes are loaded, the &SHAPE command simply won't draw anything.

&BSHAPE,n,x,y

Draws the nth bshape in the currently active table on the screen. Bshape stands for "byte shape". Note that the x coordinate here specifies which *byte* to start drawing at. Since there are 7 dots per byte, using an x value of 3 would start at the 21st dot on the screen. This is a very fast routine, but there are only 40 horizontal positions you can plot bshapes at. There are some other oddities about bshapes. They will not reflect, magnify, or pay any attention to any of the other graphics mode settings. They also are limited to the size of 28 dots wide by 21 dots high, rather than allowing you to choose any size as the shape routine does. Lastly, they will show up in different colors at odd x coordinates than they do at even coordinates. The &BSHAPE command only works in single res. If it is issued with the double res version of **Cat Graphics** loaded, it will be ignored.

Example: &BSHAPE,3,10,60 plots 4th bshape from current table at 70,60.

You will need to load a bshape table before experimenting with this command.

&FILL,x,y,color
    Fills the region containing the point x,y with color. This command will fill on either a white or a black background. If something goes wrong when you try to fill in an area, you can press any key on the keyboard to stop it. You can specify color values from 0-107 in single res, or 0-255 in double res. When using the double res version, you should make sure &THICK mode is set before doing any fills.

Example: &FILL,120,50,16   fills the area containing the point 120,50 with color number 16.

&CLR,color
    Clears the screen to chosen color. Colors 0-7 are available in single res, 0-15 in double res. The colors are as follows:

Color number	Single res color	Double res color
0	black1	black
1	green	magenta
2	purple	dark blue
3	white1	purple
4	black2	dark green
5	orange	grey
6	blue	medium blue
7	white2	light blue
8	n/a	brown
9	n/a	orange
10	n/a	grey
11	n/a	pink
12	n/a	green
13	n/a	yellow
14	n/a	aqua
15	n/a	white

Example: &CLR,0   clears the screen to black.

&FLIP,color
    Changes the colors/dots on the screen. Colors 0-7 are available in single res, 0-15 in double res. Flipping twice in a row with the same value restores the original picture, and can create a nice effect for games. The colors used here are the same as the &CLR command (see above). In all cases, any part of the screen that is black will change to your chosen color when you flip, and anything that is already that color will change to black. Everything else will generally change to a different color. Using a value of 0 will leave the screen unchanged. Using a value of 4 in single res will change all of the colors on the screen without changing anything that is black or white.

Example: &FLIP,3
    If the screen has a black background, this will change it to white if you're using single res, or purple in double res.

&COPY,mode
    Copies contents of one graphics page onto the other based on the mode chosen, 0 through 3 copy from page one to page two, 4 through 7 copy from two to one.
    Modes 0 and 4 will wipe out whatever was there and replace it with the contents of the page being copied from.

Modes 1 and 5 will "add" the two pictures together, keeping all dots that are on in either picture.

Modes 2 and 6 will "filter" the pictures, keeping only the dots that are on in *both* pictures.

Modes 3 and 7 will exclusive-or the two pictures together, which keeps dots that are on in one picture or the other, but not both.

This command works identically in single and double res.

Example: &COPY,4   copies the picture from page two onto page one.
    This does not change the contents of page two.

&VSCROLL,n

Slides the screen up or down n lines, depending on whether the value is positive or negative. N should be between -127 and +127. This command will only work with graphics page one in the double res version of **Cat Graphics.** The single res version will work with either page one or page two.

Example: &VSCROLL,15   moves contents of screen up 15 lines.

&HSCROLL,n

Slides the screen left or right n *bytes* (groups of 7 dots), depending on whether n is positive or negative. Note that if n is an odd number, the color of things on the screen will change. Scrolling an odd distance again will restore the original colors. In the double res version, the screen is scrolled over the same distance as in the single res version, but this represents 14 dots because of the greater resolution.

Example: &HSCROLL,-20   moves contents of the screen halfway over to the right.

&PENGUIN,x,y

By far the most important routine in this package, this will plot a penguin on the screen at x,y. This command uses the current mode and color. The penguin is drawn as a shape, so in double res he/she will have a different width unless you use the commands &THIN: &SHAPEMODE,1.

Example: &PENGUIN,20,40   plots a penguin at 20,40.

Graphics mode control

All of the commands in this section will affect everything drawn to the screen except for bshapes, and, where specified, text drawn by the hi-res character generator.

&MODE,n

This command sets the drawing mode as follows:

    0 - draw things on screen (default)
    1 - erase things from screen
    2 - flip contents of screen

The last mode is handy to keep things from destroying the background when they move around: if you draw a shape in flip mode twice at the same spot, it will leave the area you drew it on the same way it was before. Also, when using flip mode, things drawn with one color on top of a different one will usually show up in a third color. This command works identically in single and double res, and does not affect hi-res text.

Example: &MODE,2   sets flip mode.

&COL,color

Sets color to draw objects in. All of the commands that use the color you set with the &COL command can be drawn in 108 different colors in single res, or 256 in double res. These are the same colors used in our other graphics programs. You can see what the 108 single res colors look like by running program CTEST. They are also pictured on the back of the box. The blended colors may not work very well if you try to draw lines or detailed shapes with them. They look best if you use them for color fills or drawing things that are reasonably thick. Note that in single res, three commands, &CLR, &CHKDOT and &FLIP, use only the basic 8 colors, so they're numbered from 0-7, the same way as Applesoft HCOLOR values. The following table gives the names of these 8 colors, and the equivalent numbers to use for the &COL, &TXTCOL, and &FILL commands. In double res, the basic 16 colors can be obtained by taking the values used with &CLR, &CHKDOT and &FLIP, and multiplying them by 16.

Color number	Color name	Single res equivalent
0	BLACK1	80
1	GREEN	87
2	PURPLE	101
3	WHITE1	77
4	BLACK2	53
5	ORANGE	60
6	BLUE	70
7	WHITE2	52

This command does not affect hi-res text.

Example: &COL,0   selects color 0 (white in single res, black in double res).

&OUTLINE

This command affects boxes and circles only, and works identically in single and double res. After this command is issued, only the outlines of boxes and circles will be drawn. This is the default condition.

&FULL

This command affects boxes and circles only, and works identically in single and double res. After this command is issued, all boxes and circles will be drawn filled in.

&SHAPEMODE,n

This command controls the way that shapes are drawn on the screen, and works identically in single and double res. The legal values for n are 0 through 7, which have the following effects:

    0 - normal
    1 - double width
    2 - double height
    3 - double height and width
    4 - inverse
    5 - double width inverse
    6 - double height inverse
    7 - double height and width inverse

Example: &SHAPEMODE,4:&PENGUIN,50,50   draws an inverse penguin.

&THICK

This command works somewhat differently in single and double res. In single res, lines drawn in black or white are normally made as thin as possible, and may sometimes have color fringing. The &THICK command causes everything drawn in black or white to be made twice as thick. This will keep you from drawing things with quite as much detail, but there will be little or no color fringing. All colors are available in single res regardless of whether &THICK or &THIN mode is active (thin mode is the default).

In double res, &THICK mode is the default, and it causes everything to be drawn at an effective resolution of 140 by 192. This will cause shapes to appear twice as wide as in single res. It is necessary to use &THICK mode in double res to get any colors besides black and white.

&THIN

This command is the counterpart of &THICK. In single res, it allows anything drawn in black or white to be drawn as thin as possible.

In double res, the &THIN command allows you to do 560 by 192 graphics in black and white (on a color monitor, you will get some color fringing). You should use color values 0 for black and 1 for white when using this mode.

&SMOOTH,n

Sets smooth scrolling flag. 0 is the default, and scrolls vertically in a single jump. Issuing a &SMOOTH,1 command will enable smooth scrolling, which is smoother but slower. This flag does not affect horizontal scrolling, and you can't scroll backwards when smooth scrolling is in effect. This command is only available in single res. Trying to use it with the double res version of **Cat Graphics** will result in a syntax error.

Example: &SMOOTH,1   enables smooth scrolling.

&PAGE,n

Selects which hi-res page to draw things on. The page that is active when you initialize **Cat Graphics** will depend on which version you are using. Read Appendix B for more information on this.

In single res, there are two hi-res pages that can be displayed. In double res, there is only one. For compatibility, 16K of the auxiliary memory has been set aside to act as a second graphics page. This page can be drawn to, but not displayed.

Example: &PAGE,2   directs all graphics to be drawn on page two.

&DISPLAY,n

Determines which page is displayed. This command can also be used to set hi-res, lo-res or text mode, and control the four line text window at the bottom of the screen. In single res, the values of n set the display modes as follows:

    1 - hi-res page 1
    2 - hi-res page 2
    3 - hi-res page 1 with text window
    4 - hi-res page 2 with text window
    5 - lo-res page 1
    6 - lo-res page 2
    7 - lo-res page 1 with text window
    8 - lo-res page 2 with text window
    9 - text page 1
    10 - text page 2

This command can be useful if you want to keep two different displays available from your program. Also, a common animation technique is to display one page while drawing new things on the other, and then switch when you're finished drawing. That way people that use your program won't actually see the shapes being drawn, filled in, or whatever. This is known as "page flipping", and is a handy way to make animation look smoother.

In double res, the second hi-res page cannot be displayed. Also, the text and lo-res pages are displayed in 80 column mode. For 40 column text or lo-res graphics, you should add 16 to the value listed for the mode you wish to display. It's also important to note that the hi-res page must be displayed when you attempt to draw on it, when using double res **Cat Graphics.** With the single res mode, any display can be active while drawing on the hi-res screen.

Example: &DISPLAY,1   displays hi-res page one.

&MAG,n,x,y

This command causes everything drawn within a certain region to be shown in a magnified view on the other page. If you want to switch to the magnified view and draw some things on top of it, you should turn this option off, or the things you draw may be magnified on top of your original picture! Drawing on page 1, for instance would produce a magnified image on page 2. Simply activating page 2, however, does not disable magnification, so everything you draw will now be magnified to page 1, if it falls within the area of magnification. If you don't want this to happen, you should deactivate the magnification option. This doesn't mean that you can't look at the magnified view while it is active, however. Issuing the commands &PAGE,1:&MAG,3,140,80:&DISPLAY,2 will allow you to view the magnified image on page 2 while drawing things on page 1. This can make it easy to write shape editors or other programs for working with fine details.

X and y specify the center of the region that you want magnified, and n chooses the level of magnification as follows:

    0 - turn magnification off
    1 - magnify times 2
    2 - magnify times 4
    3 - magnify times 8

Notice that when using magnification modes 2 and 3, there will be a small region off to the right of the magnified view which displays the magnified region normal size.

In the double res version, you must use &THICK mode while the magnifier is active, page one must be the active page, and you must avoid drawing any shapes or hi-res text in the magnified region. Also, mode 3 (magnify times 8) is not available.

Examples: &MAG,2,60,70   magnify region centered at 60,70 by 4.
    &MAG,0,0,0   turn off magnification.

&RESCAN

This command works in conjunction with the magnification option. Since the &MAG command will only let you show things that are drawn *after* it is issued, you can issue a &RESCAN command to blow up everything that is already on the screen in the magnified region. In modes 2 and 3, it will also draw a box around the normal size view of the region. If magnification is not currently in effect, this command will do nothing. This command works identically in single and double res.

&REFLECT,n

This command is useful for creating nice symmetrical designs (like the curves on the menu page) or writing kaleidoscope programs. &RELFECT,1 will cause everything that is drawn to be reflected four ways on the screen, &REFLECT,0 turns the reflection off again. This command works identically in single and double res.

Example: &REFLECT,1   turns reflection on.

Text Commands

&UNHOOK

After you initialize **Cat Graphics**, all text will be drawn on the hi-res screen. The &UNHOOK command will disconnect the hi-res text option so you can print to the normal text screen again. You should always do an &UNHOOK before using the FP command to clear the routines out of memory. Note that you may have some trouble using **Cat Graphics** with 80 column cards, as it normally reactivates the standard 40 column output routines when you do an &UNHOOK. This command works identically in single and double res.

&REHOOK

This command will reconnect the **Cat Graphics** input/output routines after an &UNHOOK command, so you can display things on both the text and graphics pages. Note that all ampersand commands are still available after an &UNHOOK, only the hi-res character generator is deactivated.

When hi-res text is active, a few special characters can be used: Control-W moves the cursor up half a line (for superscripts), and control-E moves it down half a line (for subscripts). Control-Q toggles text entered from the keyboard between upper and lower case, and control-i toggles between inverse and normal display modes. If you have an Apple //e or keyboard enhancer, you should be able to use the shift key to enter upper/lower case normally.

This command works identically in single and double res.

&AT,x,y

This command repositions the cursor anywhere on the screen. The next characters printed will appear at this location. Note that you can put text absolutely anywhere, rather than being limited to 24 vertical by 40 horizontal positions as with some character generators. Also, this is the ONLY command you can use to position text. Normal Applesoft commands like HOME, VTAB, HTAB, etc. will affect the text screen only. This command works identically in single and double res, though the range of coordinates used in double res may vary. The coordinates in single res can range from 0-279 for x, and 0-191 for y. In double res, x coordinates can range from 0-139, 0-279, or 0-559, depending on how you use the &RES command.

&LMARGIN,x

Normally, at the end of a line, (or when you hit the return key), the cursor moves back to the left side of the screen to start the next line. With this command you can change the left margin. Regardless of the setting, however, the entire screen will still scroll when you reach the bottom. This command works identically in single and double res.

Example: &LMARGIN,70   sets left margin to 70.

&HSPACE,n
Sets horizontal spacing between characters. Default value is zero. This command works identically in single and double res.

Example: &HSPACE,4   set space between character to 4 dots.

&VSPACE,n
Sets vertical spacing. This also defaults to zero. This command works identically in single and double res.

Example: &VSPACE,8   set space between lines to 8.

&REWRITE
This sets a mode in which the background of each character is cleared to black as it is drawn. In this mode you can type or print text on top of something and it will still be readable. This is the default mode. This command works identically in single and double res.

&WRITE
This sets a mode where the background of all characters is let alone. This allows you to superimpose characters on top of one another, or on top of colored backgrounds. This mode is a bit messy to use if you're typing and want to back up to correct an error, but it's much faster than rewrite mode. It's a good idea to type &WRITE before starting to edit a program, so it won't take so long to list to the screen while you work on it. This command works identically in single and double res.

&CHARMODE,n
This command changes the way characters are drawn on the screen. The legal values for n are 0 through 8, which have the following effects:

    0 - normal
    1 - double width
    2 - double height
    3 - double height and width
    4 - inverse
    5 - double width inverse
    6 - double height inverse
    7 - double height and width inverse
    8 - bold mode

Note that bold mode will not work properly unless you are also in write mode (see the &WRITE command). This command works identically in single and double res.

Example: &CHARMODE,3   prints text at double height and width.

&TXTCOL,c
This command sets the color that the characters are drawn in. Colors 0-107 are available in single res, or 0-255 in double res. Note that text will not be very readable in some colors unless you're using a font with fairly thick characters or one of the expanded size options.

Example: &TXTCOL,53   prints text in color 53.

Table related commands

There are 4 kinds of tables that you can use in **Cat Graphics:** font tables, shape tables, sound tables, and bshape tables. You can recognize these on a disk catalog by their suffixes: .CFNT, .CSHP, .SLIB, or .BSHP. Normally you don't have to type these suffixes, they are automatically appended to the end of the file name you specify by the **Cat Graphics** utilities. You will need to type out the full name when using normal Dos utilities to transfer files and so on. The other commonly used suffixes in **Cat Graphics** are .PIC for a hi-res picture, .PAC for a single res packed picture, and .DPK for a double res packed picture.

Cat Graphics will automatically decide where to put any tables you want to use and keep track of them for you, so you don't have to worry about keeping track of memory addresses and stuff like that. For those that want more control over how things are arranged, there's more information in Appendix B.

&LOAD,n,"filename"

Loads a table into memory. If no other tables of that type are currently loaded, it will become the active table. The filename should be just the name of the table you want to load in quotes. You should leave off suffixes such as ".CFNT" or ".PAC", as they will automatically be appended for you. You may not use string variables, and you may not append Dos parameters like ",D2". See the &DRIVE command if you want to use two disk drives. N specifies the type of table to load:

1 - font table
2 - shape table
3 - sound table
4 - bshape table
5 - packed picture
6 - binary file

The last two types listed here are special. For type 5, a packed picture, it is assumed that you just want to load it and unpack it, and that after that you don't want to keep it around. So, when a packed picture is loaded, the memory it is loaded into is not reserved, and the next thing you load with the &LOAD command will go right on top of it. This is convenient for programs that use a lot of pictures, such as hi-res adventures, because if every picture loaded were kept around after being unpacked, you would run out of memory in no time. If you use type 6, no suffix will be added on to the name you specify, and your file will be loaded into memory where nothing will overwrite it. You can use this for any data you may have saved in a binary file, or machine language routines, provided that they're completely relocatable. You can find the address that your binary file was loaded at with the &ADDR command, which is described below. The &LOAD command will give an ?OUT OF MEMORY ERROR if there isn't enough room left for the table you want to load. This may also overwrite part of DOS, if you were trying to load a very big file, so it's a good idea to reboot if you get an error, just to be safe. You are also limited to 8 tables of each type, trying to load more than that will also cause an out of memory error.

The only difference between the single and double res version when using the &LOAD command is that the single res version will load single res packed pictures (.PAC suffix), and the double res version loads double res packed pictures (.DPK suffix).

Example: &LOAD,3,"PIANO":&SOUND,15

This example loads the sound table named PIANO on the **Cat Graphics** master disk, and plays the 16th sound in the table.

&ADDR,var
Sets var to the address at which the last binary file was loaded. This command works identically with single and double res. It will not only tell you where something was loaded by the &LOAD command, but will also tell you where a BLOADed file went. For instance, the following short program could serve as a utility to find the start of any binary file on one of your disks.

```
10 INPUT "NAME OF FILE?";A$
20 PRINT CHR$(4);"BLOAD ";A$
30 &ADDR,X
40 PRINT "THE BINARY STARTS AT ADDRESS ";X;"."
50 END
```

&STAB,n
Selects nth table as the active shape table. Notice that since the first shape table loaded is automatically selected as active, you don't have to use this command (and the following ones) unless you want to have more than one table available at the same time. This command works identically with single and double res.

Example: &STAB,2 activates second shape table loaded.

&FTAB,n
Selects nth font. Note that the font which is active when **Cat Graphics** is first loaded is font number one, any additional fonts you load will start with two. The exception to this rule comes when you reconfigure **Cat Graphics.** The reconfigured version has no built in font, so the first font you load will be number one. This command works identically with single and double res.

Example: &FTAB,5 activates fifth font loaded.

&SNTAB,n
Selects nth sound table. This command works identically with single and double res.

Example: &SNTAB,1 activates first sound table loaded.

&BTAB,n
Selects nth bshape table. This command will be ignored if issued while the double res version of **Cat Graphics** is loaded.

Example: &BTAB,2 activates second bshape table loaded.

&WIPEMEM
Clears all tables (except for the built in default font) out of memory, making room for more tables. This command works identically in single and double res.

&START,addr
Sets the beginning of the area in memory reserved for tables. This command works identically in single and double res. See appendix B for more information on using it.

Example: &START,32768 starts loading tables at 32768.

&FINISH,addr
Sets the end of the area in memory reserved for tables. This command works identically in single and double res. See appendix B for more information on using it.

Example: &FINISH,32800 sets end of table space to 32800.

Sound effects and miscellaneous utilities

&SOUND,n

Plays the nth sound from the currently active table. This command works identically in single and double res.

Example: &SOUND,0 Plays first sound in current table.

&NOISE,f1,s1,f2,s2,d,i1,i2,i3,i4,i5,r

Instead of setting up a sound table, you can use this command to make one particular noise, giving it the same parameters as the numbers listed on the screen in the sound effects editor. See Chapter V for more information on what each parameter does. This command works identically in single and double res.

Examples: &NOISE,9,2,7,-2,4000,0,4,0,0,-777,6
&NOISE,111,30,7,-19,65535,-43,12,11,0,-777,1
&NOISE,30,1,7,-1,65535,0,0,0,0,0,1
These examples make various funny noises.

&CHKDOT,x,y,var

Sets var to the color of the dot on the screen at position x,y. The colors are numbered the same way as described in the explanation of the &CLR command. They range from 0 to 7 when using the single res version of **Cat Graphics.**

In double res, values will range from 0 to 15 when in &THICK mode. When in &THIN mode, however, the &CHKDOT command will return either a 0 for black, or a 1 for white.

Example: &CHKDOT,140,20,N1 sets N1 to the color of dot at 140,20.

&PACK,loc,var

This command will take the picture on the currently active screen and make a compressed version of it in memory. The packed picture will be stored starting at the memory location specified by loc, and var will be set to the length of the compressed version. If you save a single res picture on disk with a filename ending in .PIC, you can use the pack/unpack option from the main menu, and it will automatically save the packed picture for you. If you want to save it yourself, you should use something like the following sequence:

```
&PEEK 792,N:IF N>49000 THEN &PEEK 103, N:N=N+2000:REM finds space for packed
picture
&PACK,N,L
PRINT CHR$(4); "BSAVE";N$;".PAC,A";N;",L";L
```

For a double res picture, you should change ".PAC" to ".DPK".

Normally a single res picture will take up to 33 or 34 sectors on disk (when you save a picture, use ",L$1FF8" instead of ",L$2000" for the length, and it will only take up 33 sectors). Packed pictures take up quite a bit less, usually somewhere between 10 and 25 sectors. If you want to use them from one of your programs, though, you'll need to set aside an area of memory large enough to load them into for unpacking. The &LOAD command will help you take care of this, but you should make sure there is a good deal of space left open when you try to load a packed picture.

&UNPACK,loc

This command will unpack a picture at memory location loc onto the current hi-res page. If enough space has been set aside, you can use this along with the &LOAD command to display a packed picture stored on disk. If your picture is called "TREEFROG.PAC", then the following sequence of commands should load and unpack it for you:

```
&LOAD,5,"TREEFROG"
&ADDR,L:&UNPACK,L
```

This would also work in the double res version with a picture stored in file "TREEFROG .DPK".

&POKE addr,n

The normal Applesoft POKE command puts a value into one memory location, so you can only specify a value from 0 to 255. Often you'll have to poke an address into two successive locations with two pokes, or you may just want to store a bigger number somewhere where other programs can read it back later. So, the &POKE command will take the 16 bit value n and put it into memory locations addr and addr+1. The value will be stored low byte first, high byte second, just as addresses are in 6502 assembly language. This command works identically with single and double res.

Example: &POKE 0,R% stores the value of R% in locations 0 and 1.

&PEEK addr,var

To go along with the above, this command will put the 16 bit value found at addr and addr+1 into the variable you specify. This command works identically with single and double res.

Example: &PEEK 0,KV same result as KV=PEEK(0)+256*PEEK(1)

&PDL n,var

Reads paddle number n and puts the result into var. This function is provided because the Applesoft PDL(n) function only gives values from 0-255, but the hi-res screen is 280 dots across, and it's often handy to be able to pick any point on the hi-res screen with the paddles or joystick. Most joysticks can provide a wider range of values, up to about 340. If you want to test yours out, try entering the following short program:

    10 &PDL 0,X: PRINT X: GOTO 10

&DRIVE,n

Lets you specify the active disk drive for all Dos commands without actually having to issue a Dos command with the ",D1" or ",D2" parameter. Very useful with the &LOAD command, which won't take the D parameter.

Example: &DRIVE,2 selects drive 2 as default drive.

&PENGUIN,x,y

Just thought I'd list it here again in case you missed it the first time. This command draws a penguin on the screen at x,y. Frnk frnk!

IV. The shape/font editor

Types of tables

This program (option 2 from the main menu) lets you design shapes to use from your own programs. There are three different types of tables that you can use to edit as follows:

*Fonts are groups of 96 characters that replace the standard character set with alternate typestyles. These characters are defined on a grid of varying size, but all characters within a given font must be the same size. The fonts provided for you range from 5 x 6 dots up to 9 x 9. You can make fonts smaller or larger than that if you like, but keep in mind that larger fonts will take up more memory and print more slowly.

*Shapes are similar to fonts, except you can have any number of them in a table, from 1 to 104. Also, instead of redefining the character set, these are meant for you to design whatever you might find handy to draw on the screen - airplanes, dragons, trees, or whatever. You should limit your shapes to 40 x 32, or the editor may not be able to handle them properly.

*Bshapes are a special kind of shape. They come in only one size, 28 x 21, and you can only have up to 42 of them in a table. Also, instead of being able to plot them anywhere on the screen, the x coordinates are limited to 40 positions. However, they do have the advantage of being very fast, and they let you draw multicolored shapes with pretty good control over how the final result looks on the screen. The editor for these shapes is actually a modified version of the editor used for shapes and fonts, and has a few minor differences. Any exceptions to the general procedures described in this section will be noted in parenthesis.

Options 1 through 3 will allow you to edit a table that's already been set up. If you pick one of these options, you will be asked for the name of the file you want to work with, and it will be loaded. If you pick an option 4 through 6, you will be asked for the name of the file, then, where appropriate, for the number of shapes and/or the size of shapes you want to use. After all of this is taken care of, the program will print the instructions screen and run the actual editor.

Editing shapes

The editor will pause until you press a key, giving you a chance to look over the instructions. Then it will display all of the shapes on the screen. If you have just set up a new table, then the screen should be blank except for the instructions at the bottom. At this point, you should use the joystick to move the box to the shape you want to edit, and press button 0. When editing a new table, move the box to the upper left hand corner of the screen to start with the first shape. (If you're using the bshape editor, you move the box with the arrow keys on the keyboard instead of the joystick, and press **Return** to select a character to edit).

After you've chosen a shape, a grid will be drawn in the upper left corner of the screen with a blown up view of the shape. The shape will be drawn normal size twice, once at the upper right, once at the lower right. The one at the top will change as you draw things, the one at the bottom will stay the same, so you can compare any changes you make to the way the shape looked before, and see which way you like it better. For normal shapes, next to the bottom shape will be an example of what the shape will look like in color. This may vary slightly in practice depending on which color you use, and whether you draw the shape at an odd or even position. If you're using the bshape editor, there will be two additional views in

-19-

between. These show what it looks like as a bshape. The upper one shows how it looks if drawn at an odd horizontal position, the bottom how it will look at an even position. Because of the way the Apple does its colors, the two should have some of the colors switched. To keep this from causing trouble in programs, it's a good idea to plot bshapes on even coordinates only, or on odd coordinates only.

Once the shape has been drawn, you should see a blinking square in the upper left corner. This is your cursor, and you use it to move around and change parts of the shape. You can move it in 8 directions with the keys centered around the "**S**" key. To get where you're going a bit faster, you can also move in jumps of 5 dots by holding down the control key along with the movement key. The "S" key itself can be used to draw things, it will flip the dot underneath the cursor. So if it was off, it will be turned on, if it was already on, it will be turned back off again.

Of course, it would be rather bothersome to have to turn on every single dot by hand, so there's an option to let you leave behind a trail of dots as you move. Press the "**1**" key and try it. If you press the "**2**" key, you will *erase* any dots you move over. The "**3**" key will let you move around again without affecting anything. Down at the bottom of the screen, the top line of text at the right should say "PLOT MODE", "ERASE MODE", or "TRAVEL MODE", to tell you which of the three options is active. (In the bshape editor, it will say "COLOR= WHITE", "COLOR=BLACK" or "COLOR=NONE". This is because you can also use the keys **4** through **7** to select other colors to draw in. Note that the separate strip of four dots to the right is used to control the colors of the shape, and you shouldn't try to draw anything there. You can fiddle around with those to change the colors by hand if you know what you're doing, but I won't try to explain it here).

If you want to clear the grid out and start all over again, press **control-Y.** (When using the bshape editor, if your drawing color is set to anything between 4 and 7, the grid will be cleared to that color). You can invert everything by pressing **control-I,** and if you don't like your shape that way, you can return it to normal by pressing **control-I** again. Pressing **control-F** will let you fill in a large solid area, as long as you've drawn the entire border with no gaps. If the cursor is on top of a black region, it will fill it with white, and vice versa. (In the bshape editor, this command was changed to **control-P** to make it harder to fill a region by accident). Lastly, if you want to move your shape around on the grid, you can use the keys **I, J, K,** and **M** to move your shape up, left, right and down respectively. Any part of your shape that slides off the edge of the grid will be lost. Also note that if you slide a bshape left or right you'll probably mess up the color scheme.

When you're done editing a shape, you can use the **arrow** keys to move to the next (or previous) shape in the table. (This will not work in the bshape editor). Or you can return to the menu of shapes by pressing the **escape** key and picking another one. You should also return there when you're all done. Pressing **control-S** at the main menu will save your table to disk. If you don't want to keep any changes you've made, press **control-X** to simply exit the program.

V. The sound editor

When you run the sound effects editor (option 3 from the main menu), it will give you a list of four options. Option 1 will let you edit a table of sounds that you already have on disk. The disk comes with a table on it called "piano" that has the notes in it from one octave below middle C to two octaves above, including sharps and flats. Option 2 will create a new table for you with just one sound in it. Note that unlike the shape routines, you don't have to decide how many sounds you want to make room for, but can add and delete sounds at any time. Option 3 will let you edit the currently active table if there is one in memory, otherwise it will do nothing. Option 4 branches you back to the main menu.

While you are editing sounds, the screen display will be split up into three sections. The top part gives a summary of commands available, the middle part gives you a "window" onto part of the table you are editing, and the bottom part of the screen shows the sound you are currently working on, and is also used to prompt you for any input that is required.

Each sound is represented by a group of 11 numbers, which will usually fit onto one line of the screen. The sound in the center region which is shown in inverse is the one you are currently working on. At the start of an editing session, it will be set to the first sound in the table, which will also be copied into the working area at the bottom. You can move forwards and backwards through the table with the arrow keys. Each time you move to a new sound, the sound at the bottom of the screen will be replaced by the one you move to in the window, so be sure to add anything you're working on into the table before you try to work on another sound. The plus and minus keys will let you move through the table in steps of five to let you get to a particular sound more quickly. Anytime you move past the beginning or end of the group of sounds displayed in the window, it will scroll up or down accordingly.

So, having either found the sound you want to modify, or decided to create a new one, you'll need some idea of what all of those numbers mean. You can stick with the first five to begin with, and experiment with the rest later if you like. The first number is the pitch of tone 1, and the third number is the pitch of tone 3. The fifth number tells how long to play the note (maximum value of 65535). If you set all the numbers besides these three to zero, and set the 11th number to one, you'll just get a couple of tones mixed together. If you just want a single tone, the best thing to do is to use numbers for the frequencies that are just 1 or 2 apart, such as 198 and 200. This will give you a more vibrant note than a simple one tone routine would have anyway.

The second and fourth numbers are there to add a little spice to things. Instead of having a constant tone, you can use them to make the pitch change at a steady rate. Using a positive number will make the pitch get lower, a negative number will make it go higher. The bigger the number, the faster it will change. You can get some nice simple effects by keeping one of these two values at zero and changing the other one, or keeping them both the same so that both tones will change together. You can get quite an interesting range of results with just these five values.

So, how do you change them, anyway? Well, near the top of the screen it shows you a rectangle of 30 keys (I'm not going to repeat them here). Hitting the keys in the top row will increase the values of the sound in your work area, the second row will decrease them. If you hit one of the keys on the bottom row, then a question mark prompt will appear at the bottom of the screen, and you can type in a new value. It's a lot easier to change a number from 0 to 50 that way, certainly. By the way, the two symbols at the end are the greater than and less than symbols, typed by pressing shift-period and shift-comma respectively. *Don't* get them confused with the arrow keys, which are used to move you through the table!

Every time you change a value, the sound will be played again so you can hear how it sounds with the change. If you want to repeat the sound, just press the return key.

Now, about those other 6 numbers... The reason you have to set the 11th number to 1 is that it keeps track of how many times to repeat the sound given by the first five numbers. As long as it's set to one, the sixth through tenth values are totally ignored. But, if it's greater than one, then every time the sound is repeated, the second five numbers are added to the first five numbers! So the sixth number tells how much to change the first pitch value every time, the seventh tells how much to change the rate of change of the first pitch value, and so on. A little experimentation should clear things up.

When you're finished editing a sound, if you want to keep it, you should do one of two things before you move on to another or quit editing. If you press **control-R,** then the original sound that you started with will be replaced by the modified version you've made. However, if you liked it the way it was before, but have come up with an interesting variation, you can press **control-A** instead. Then the original sound will be left alone, and the new sound will be added on to the very end of the table you're editing.

There are just a few more commands to cover: **control-D** will delete the sound currently highlighted in the window from the table to save some space. **Control-S** will save the table you're editing to disk. **Control-L** will ask you for the name of a sound table on disk, and will add all of the sounds in it to the table you're working on. And lastly, **control-Q** will let you quit and return to the initial menu.

VI. The font converters

While this disk provides you with four fonts and an editor that you can use to design more, you'll probably want to use some of the numerous fonts that other people have already designed. Unfortunately, **Cat Graphics** uses a different font format than any other package on the market. One of the reasons for this is so that fonts can be any size. Some character generators assume that the characters are always going to be the same height and width, and just use a file with raw data in it, with no information about how the table is set up.

So, to use other fonts, you have to use the font converters to convert them to **Cat Graphics** format. These programs will work with any font from any of Penguin's other products, including the Additional Typesets disk. There are also some other programs available that use the same format, such as the Dos Toolkit character generator.

There are two font converters. The small font converter will convert Penguin small fonts, which are 7 x 8, the large font converter converts the large ones, which are 14 x 16. They take 4 times as much memory, and you can't print as many characters on the screen, but they look much nicer.

Both of the programs are pretty much self explanatory. First you will be asked for the name of the font you want to convert. This must be a file that ends in ".STS" or ".LTS" (depending on the size of the font), which will be added on to the name you type. If you want to get a font from the Dos toolkit or some other program that uses different naming conventions, use the Dos RENAME command to change its name.

Next you wait a while. The font converters are currently written in Basic, and are fairly slow. Small fonts take about three and a half minutes, large ones around thirteen or fourteen. You can watch the program scanning each character from top to bottom as it converts them, or go raid the refrigerator if you prefer. Then the program will ask you what name you want to use for the converted font, save it, and ask you if you want to do another. If not, you'll be branched back to the **Cat Graphics** menu again.

Appendix A.
Moving Cat Graphics to your work disks

If you want to use **Cat Graphics** from one of your programs, you will probably want to move the routines onto your disk along with the programs you are writing. Also, most of the utilities provided with **Cat Graphics** are easiest to use if the files to be edited are on the same disk as the utilities themselves. This can be somewhat inconvenient if you try to use a copy of the master disk, as it is almost full. The following section lists the files that you need to copy to use various features of **Cat Graphics,** and explains how to use them.

Single res **Cat Graphics** (48K version): Copy file GR1. To execute, BRUN GR1.

Single res **Cat Graphics** (64K version): Copy file GR2. To execute, BRUN GR2.

Double res **Cat Graphics:** Copy file GR5. To execute, BRUN GR5.

Shape/font editor: Copy files CHAR-EDIT 5, EDITGR, CHAREDIT, BDRAW-ED. To execute, RUN CHAR-EDIT 5.

Sound editor: Copy file SOUND EDIT. To execute, RUN SOUND EDIT with **Cat Graphics** loaded. If necessary, type BRUN GR1 or BRUN GR2 first.

Picture packer/unpacker: Copy file PACK/UNPACK. To execute, RUN PACK/UNPACK with **Cat Graphics** loaded. If necessary, BRUN GR1 or BRUN GR2 first.

Double-res shape editor: Copy files CHAR-EDIT 6, GR5, CHAREDIT DR. To execute, RUN CHAR-EDIT 6.

To transfer Basic programs, you can simply LOAD them from the **Cat Graphics** disk, insert your work disk, and SAVE them. To transfer binary files, though, you need to know where they start and how long they are. The following table lists the addresses and lengths of all the binary files mentioned in this appendix. To transfer file GR2, for example, you would type "BLOAD GR2", switch to your work disk, and type 'BSAVE GR2,A2048,L11755."

File name	Start Address	Length
GR1	24576	11625
GR2	2048	11755
GR5	2048	13552
EDITGR	2048	5888
CHAREDIT	16384	3687
BDRAW-ED	16384	3789
CHAREDIT DR	16384	3656

GR2 and GR5 load at 2048, and relocate themselves in the RAM card.

One thing worth noting about the shape/font editors: when you exit from them, whether by saving your work or just quitting without a save, they will attempt to run any program named HELLO, if there is one. On the **Cat Graphics** master disk, the HELLO program returns you to the menu. What your HELLO program does is up to you, but it's a good idea to make sure it's nothing that might wipe out any files...

Appendix B.
Memory management

This section is intended for more advanced programmers who want to re-arrange the default setup **Cat Graphics** will use. You may need to do this if you're running out of memory for tables, or if you want to use some other routines along with **Cat Graphics.** If you have some assembly language routines you've written yourself, you should be able to move them someplace where they won't cause any problems.

For all practical purposes, we can think of all available memory as being divided into four sections. Available memory is considered to be everything starting at $800 (decimal 2048) and going all the way up to where Dos starts. If you have a 64K Apple, then you can use some of that extra memory also. The Hello program on the **Cat Graphics** disk issues a "MAXFILES 1" command to make a little extra space available (about 1.2k). You may wish to do the same in your programs.

Anyway, the four sections are:
1) Space for Basic programs and variables
2) The hi-res screen (or screens)
3) The **Cat Graphics** routines
4) Space for shapes, fonts, etc.

Some of you may also wish to set aside space for your own machine language programs. For the sake of this discussion, they will be considered to be part of the area used for Basic, and it is assumed that you know how to allocate the Basic work space to keep your program from being overwritten.

The standard version of **Cat Graphics** allocates memory locations $800-$3FFF (decimal 2048-16383) for Basic, $4000-$5FFF (16384-24575) for hi-res page 2, $6000-$8DFF (24576-36351) for **Cat Graphics,** and $8F00 (36352) on up for the tables. When first initialized, though, it will check and see if you have a 64K Apple. If it finds the extra memory, it will store its tables up there, giving you 12K of table space. If you're using ProDos, Big Mac, or some other program that uses the top 16K, you may have to use the &START and &FINISH commands to move the tables elsewhere and avoid any conflicts. If you want to use both hi-res pages, you can issue a HIMEM:8192 command, but that will limit your space for Basic programs very much, giving you only 6K to work with.

File GR2 is the RAM card version of the routines. If you have a 64K Apple, then you can use this version, and the routines will be loaded into the bank switched memory that normally isn't used for much anyway. Using GR1 gives you 14K for Basic programs. This version gives you 22K, though you may have less space left for tables. GR2 allocates $800-$8FF (decimal 2048-2303) and $D000-$FFFF (53248-65535) for **Cat Graphics,** $900-$1FFF (2304-8191) for tables, $2000-$3FFF (8192-16383) for hi-res page 1, and $4000 (16384) on up for Basic.

If you use the reconfigure option from the main menu, it will create a new version of **Cat Graphics.** The way memory is allocated depends on how big the resulting file is. The **Cat Graphics** routines will start at $800 (decimal 2048), and end at some address. Let's call the ending address $XXXX. If $XXXX is less than $2000 (8192), then tables go at $XXXX-$1FFF (8191), hi-res page 1 at $2000-$3FFF (8192-16383), and $4000 (16384) on up is left for Basic. Otherwise, tables go at $XXXX-$3FFF (16383), hi-res page 2 at $4000-$5FFF (16384-24575), and $6000 (24576) on up is left for Basic. You are also given the option to change some of these default areas when creating a new version of **Cat Graphics,** but you should be careful to make sure that no two regions overlap. Also keep in mind that if you have a 64K Apple, the default area for tables will be ignored, and **Cat Graphics** will try to store tables in the top 12K of banked memory.

So, if you know where you want to re-allocate everything, here's how you can modify the addresses used for each of the four purposes listed above.

1. To change the address where the Basic work area begins, you should poke the start address plus one into memory locations 103 and 104 (the &POKE command is handy for this, since it works with 16 bit values), and then poke a 0 into the start address. For instance, if you want your program to start at 24576, you would type "POKE 103,1: POKE 104,64: POKE 24576,0". If **Cat Graphics** were loaded, you could use "&POKE 103,24577: POKE24576,0" instead. Note that this must be done *before* your program is loaded. You might want to have your hello program issue the pokes and then run your main program, or you could put them in an exec file.

To set the address of the end of the Basic work area, you simply do a "HIMEM:n", where n is the address you want to use.

2. If you only need to use one of the hi-res screens, you can pick whichever one will make it easier for you to arrange everything else in memory. Page one is at $2000-$3FFF (decimal 8192-16383), and page two is at $4000-$5FFF (16384-24575). Just make sure nothing else overlaps the page you want to use, and set things up with the commands "&PAGE,n: &DISPLAY,n", where n is the number of the page you want to use. If this is the default page, you don't even have to bother.

If you need to use both hi-res pages, then you'll have to leave the whole area from $2000-$5FFF (8192-24575) open. This will leave you 8K less available for everything else. With GR1 and GR2, the non-active page is part of the Basic work area, so it should be fairly easy to move the top (or bottom) 8K so you can use both pages. If you reconfigure **Cat Graphics,** you can only use both pages if the reconfigured version doesn't overlap the beginning of hi-res page 1.

3. The only real control you have over where **Cat Graphics** is in memory is which configuration you choose to load. If you want a shorter version of the package, you can use the reconfigure option to select only those routines you want to have available. There's more information on how to use the reconfigure program in Appendix E.

4. The ampersand commands &START and &FINISH are provided to let you specify where the table space starts and stops. You could also use these commands to store tables in two different parts of memory. For instance, if you're using GR1 on a 64k Apple, normally the space from $8F00 (decimal 36352) to the start of DOS would be unused. If you want, though, you can set this area as the active table space, load some things there, and then switch back to the top 12K and load the rest of your tables. One other thing that you may find convenient with GR1, particularly if you're using a 48K Apple - the curve routine was deliberately put at the very end of **Cat Graphics.** So, if your programs don't use the &CURVE command at all, you can make an extra 1.5K available for tables by doing an &START,35072.

A note is in order about just how much space **Cat Graphics** tables actually do take up. In the following formulas, N stands for number of items in the table, W stands for width of the shapes or characters, and H stands for height of the shapes or the characters.

Fonts: $96*(2+INT((H*W/8)+.9))+3$
Shapes: $N*(2+INT((H*W/8)+.9))+3$
Sounds: $21*N+1$
Bshapes: $86*N+1$

The following line of Basic will let your programs determine exactly how much room is left for tables at any given time:

&PEEK 792,A: &PEEK 794,B: S = B-A:PRINT "Free space = ";S

Program SPACE CHECK gives a good example of how to use this formula.

Appendix C.
Using the routines from assembly language

To make it easier for you to call the routines in the package from your own assembly language programs, all of the different configurations set up a jump table in page three through which you can call many of the major routines. Most of the important flags and variables are also stored in page 3 or in page 0. In addition, the character input and output routines are patched in through the standard vectors, so you can call them just as you would the normal Apple routines.

There are a number of differences in the internal functioning of the double res version of **Cat Graphics,** so some things may not work as described in this appendix. These differences are not currently documented.

A complete description of how the routines are set up and how to use them is beyond the scope of this manual. The following list of variables and entry points should hopefully be sufficient to show you how to access some of the features of the package from your own programs.

The variables:

```
$0002 WHICH - which shape or sound to use
$001B RADIUS - radius for circle routine
$001C MODE - specifies plot, erase or flip
$001E COLOR - color to draw things in
$00E8 TABSTART - address of table to use
$00F9 X1 - x coordinate to plot at (2 bytes)
$00FB Y1 - y coordinate to plot at
$00FC X2 - second x coord. where needed (2 bytes)
$00FE Y2 - second y coord. where needed
$00FF MODE 2 - special flag for some routines
$0300 DOTVAL - color returned by CHKDOT
$0301 MMODE - graphics mode flags (see below)
$0302 WINDX - left edge of magnified region
$0303 WINDY - top of magnified region
$0305 SHMODE - shape mode (see &SHAPEMODE command)
$0306 MDFLAG - 1=destructive, 0=non-destructive characters
$0307 TXMODE - value set by &CHARMODE command
$0308 XX - x coordinate of cursor (2 bytes)
$030A YY - y coordinate of cursor
$030C TXCOLOR - color for hi-res text
$030D HSPACE - horizontal spacing for text
$030E VSPACE - vertical spacing for text
$0310 FNTADDR - address of current font
$0312 SHPADDR - address of current shape table
$0314 SNDADDR - address of current sound table
$0316 BSHADDR - address of current bshape table
$0318 NFREE - next free address to load table at
$031A MEMTOP - top of area reserved for tables
$031C NMFNTS - number of fonts in memory
$031D NMSHPS - number of shape tables in memory
```

$031E NMSNDS - number of sound tables in memory
$031F NMBSHS - number of bshape tables in memory
$0320 FNTTAB - addresses of all font tables in memory
$0330 SHPTAB - addresses of shape tables in memory
$0340 SNDTAB - addresses of sound tables in memory
$0350 BSHTAB - addresses of bshape tables in memory
$0360 START - beginning of area reserved for tables
$0362 PAGE - high byte of active hi-res page, $20 or $40

There now follows a list of all the routines in the jump table and the parameters that need to be set before calling them. Note that the COLOR and MODE values will affect DOT, PENGUIN, LINE, CURVE, BOX, CHARSET, and CIRCLE, but are not listed under these commands. This is because you usually won't want to change their values every time you draw something, but only occasionally.

$0380 DOT :X1, Y1. plots a dot at (X1,Y1).

$0383 PENGUIN :X1,Y1. plots a penguin at (X1,Y1).

$0386 CHKDOT :X1,Y1. checks dot at (X1,Y1), and returns it's color in DOTVAL.

$0389 LINE :X1, Y1, X2, Y2. draws a line from (X1,Y1) to (X2,Y2).

$038C CURVE :X1, Y1, X2, Y2. draws a curve from (X1,Y1) to (X2,Y2). $19 and $1A should contain 8 bit signed values to determine the shape of the curve.

$038F CLEAR :MODE 2, clears the screen to color given by MODE2.

$0392 FLIP :MODE 2. flips the screen with color given by MODE2.

$0395 HISCROLL :MODE2. scrolls screen vertically by the number of lines in MODE2.

$0398 BOX :X1, Y1, X2, Y2, MODE2. draws a box with corners at (X1,Y1) and (X2,Y2). MODE2 specifies outline (0) or solid (1).

$039B SSCROLL :MODE2, scrolls screen sideways by the number of bytes in MODE2.

$039E CHARSET :X1, Y1, WHICH. draws the indicated shape out of the current table at (X1, Y1).

$03A1 BDRAW :X1, Y1, WHICH. draws the indicated bshape out of the current table at (7*X1,Y1).

$03A4 CIRCLE :X1, Y1, RADIUS, MODE2. draws a circle with center at (X1,Y1) and indicated radius. MODE2 specifies outline (0) or solid (1).

$03BB FILL :X1,Y1,COLOR. fills region containing point (X1,Y1) with the specified color.

$03C1 SOUND :WHICH. plays the indicated sound out of the current table.

A couple of notes: The variable MMODE has the format XXXABCDD. The three X's represent unused bits. A is the smooth scroll flag, when it is set to 1, smooth scrolling is enabled. B is a flag to enable four way reflection. C enables "thick mode", as described under the &THICK command. DD specifies the magnification mode: 00 is no magnification, 01 is x2, 10 is x4, and 11 is x8.

When using any of the table based routines (CHARSET, BDRAW or SOUND), you have to put the address of the table you want to use into TABSTART ($E8 and $E9). To use the currently active table, you can just copy the appropriate address from page three into TABSTART.

One last thing - if you're using GR2, the language card version of the package, you should make sure the RAM card is read enabled before calling any of the routines. Of course, if you want to return to Basic, you should disable it before you return. There are two subroutines that can be used for this purpose. RAMREAD, located at $84F, enables the RAM card. ROMREAD, at $848, re-enables the motherboard ROM.

Appendix D.
Ampersand command list

&ADDR,var
&AT,x,y
&BOX,x1,y1,x2,y2
&BSHAPE,n,x,y
&BTAB,n
&CHARMODE,n
&CHKDOT,x,y,var
&CIRCLE,x,y,radius
&CLR,color
&COL,color
&COPY,mode
&CURVE,x1,y1,x2,y2,a,b
&DISPLAY,n
&DOT,x,y
&DRIVE,n
&FILL,x,y,color
&FINISH,add
&FLIP,color
&FTAB,n
&FULL
&HSPACE,n
&HSCROLL,n
&LINE,x1,y1,x2,y2
&LMARGIN,n
&LOAD,n,"filename"
&MAG,n,x,y
&MODE,n
&NOISE,f1,s1,f2,s2,d,i1,i2,i3,i4,i5,r
&OUTLINE
&PACK,loc,var
&PAGE,n
&PDL n,var
&PEEK addr,var
&PENGUIN,x,y
&POKE addr,n
&REFLECT,n
&REHOOK
&RESCAN
&REWRITE
&SHAPE,n,x,y
&SHAPEMODE,n
&SMOOTH,n
&SNTAB,n
&SOUND,n
&STAB,n

```
&START,addr
&THICK
&THIN
&TXTCOL,color
&UNHOOK
&UNPACK,loc
&VSCROLL,n
&VSPACE,n
&WIPEMEM
&WRITE
```

Appendix E.
Reconfiguring Cat Graphics
(single res only)

The reconfigure program is accessed as option 8 from the **Cat Graphics** menu. Its purpose is to allow you to save memory by creating a version of the package containing only the routines you want. The version of **Cat Graphics** created will work in single res graphics mode only.

When you run the program, some instructions will be printed. Press any key, and a menu of 24 different commands will appear. Notice that some of the **Cat Graphics** commands are not on this list. These are very short routines, and are built in to the package. The "text commands" and the hi-res text generator are all grouped together as one entry - you must either load them all, or none of them.

Each command on the list should have a number and either a "Y" or an "N" listed on the same line. This indicates whether it is currently selected to be loaded. All the routines start out with the letter "N" for "NO". The numbers show roughly how much memory each routine requires to load. A running total is kept as you choose which routines you want to load.

The first routine should initially be displayed in inverse. You can move backwards and forwards through the list with the **arrow** keys. Pressing the **space** bar will change the status of a routine. When you select a routine that requires another routine in order to work properly, that other routine will be selected automatically. For example, choosing to load LINE or SHAPE would also automatically select DOT. Choosing BOX would select both DOT and LINE. Conversely, if you try to deselect a routine that is needed by another, a buzzer will sound. The only way to deselect the routine is to first get rid of all the routines that depend on it.

Once you've chosen which routines you want, you can hit the **escape** key. Some defaults will be displayed, which you can either accept, or type in new values. The new configuration will then be created, and you will be asked to put in the disk you want to save it on. The new version is saved as file "CAT GRAPHICS 3.0". Note that when you use this version, which can be initialized with a BRUN command like the others, it will not automatically display the hi-res screen and activate the hi-res text option. To make it more flexible, the reconfigured version does not include a default font, so that if you don't want to use the standard character set, it won't waste any memory. This does mean, however, that if you want to use hi-res text at all you'll have to load a font with the &LOAD command, and then do an &REHOOK. You'll also have to switch to the graphics screen if the text screen was active when you started.

Appendix F.
Electric Fire

Electric Fire is sort of like an electronic kaleidoscope. Though individual preferences may vary, I recommend that Electric Fire be viewed in a dark room, with music playing and someone giving you a backrub. In any case, this appendix contains some technical information on the workings of Electric Fire, for those who are interested.

If you press the "**?**" key while the program is running, it will give you a list of keys that are active, along with one line summaries of what each one does. While some of these descriptions may seem rather cryptic, a little experimentation should clear most of them up. For the more inquisitive, however, a description follows of roughly how the program works and what the various options do.

The basic idea of the program is to keep track of a couple of moving points. Let's call them (x1,y1) and (x2,y2). Each of these coordinates has a set amount it moves each time, lets call these values dx1, dx2, dy1, and dy2. Every time that the points are to be moved, dx1 is added to x1, dy1 is added to y1, and so on. Lastly, we keep track of the points we've drawn things at in an array so that we can erase them later. Every time something new is drawn, the oldest thing on the screen is erased, so the pattern will leave a trail behind it of a set number of items. The **G, H, J**, and **K** keys, as well as the greater-than and less-than symbols control the length of the trail.

Depending on what the program is doing, sometimes only one of the two points will be used. In the line mode, for instance, the two points are used as the endpoints of the line, but in circle mode, one point is taken as the center of the circle, and the other is ignored.

The last major thing that's kept track of is a counter that tells how often to flip the colors on the screen. Pressing the "**1**" key makes them flip every time, giving a shimmering effect. The 2,3 and parentheses keys will also affect the flip rate.

Most of the other keys will either control various parameters of the movement of the points, or turn on and off some special options. The "gravity" option (keys **7** and **8**), for instance, will pull the two points towards each other. Every time through the main loop, the values of dx1, dy1, dx2, and dy2 are either increased or decreased by one to make the point move closer to the current position of the other point. Usually it will overshoot when it finally gets close, though. This option is active when you start up the program, and that's why the points go in nice curved paths.

The "delta swapper" (keys **Q** and **W**) simply switches the values of dx1 and dy1 with dx2 and dy2 every time the points move. This sometimes creates interesting zig-zagging patterns.

The **E, R,** and **T** keys control the maximum size allowed when new random values are chosen for dx1, dy1, dx2 and dy2. **E** allows only very small values, **T** allows much larger values.

The **Y** and **U** keys control an option that I originally put in to limit the size of lines, though it can do some other interesting things in conjunction with various options. What it does is to replace the value of (x2,y2) with (x1+dx2,y1+dy2) every time new positions are calculated.

The **A,S,D** and **F** keys control options to freeze and unfreeze the values of x2 and y2. When a value is frozen, it will change no matter what happens.

The **Z** and **X** keys toggle a couple of options I meant to be used when you're plotting lines, to let you plot horizontal or vertical lines only. If you enable both options at the same time, you'll just get dots. The options are basically to always set x2 to the same value as x1, or to always set y2 to the value of y1.

There are several other options listed on the four screens of instructions, such as the space bar to freeze the picture, but most of them should be pretty much self-explanatory.

Appendix G.
Double resolution graphics

To use double res graphics, you need either a //e with an extended 80 column card, or an Apple //c. If you're not sure if your machine is set up properly for double res, consult your dealer.

The double res mode can be used two ways. Just as standard Apple graphics can be treated as 280*192 black and white, or 140*192 with 6 colors, double res can be treated as 560*192 black and white, or 140*192 with 16 colors. For 560*192 graphics, you may want to work with a black and white monitor for greater precision, or possibly an RGB monitor. 16 color graphics look better on an RGB monitor as well.

Three things on the **Cat Graphics** disk will work with double res. The double res version of Electric fire works the same way as the single res version, except that there are no color flips.

The double res shape/font editor works with the same file formats as the single res version. The only difference here is that since double res gives you more control over the width of your shapes, the double res shape editor shows you how they will look at half the width and twice the width of shapes on the single res screen. You can also display shapes on the double res screen that look the way they do in the single res editor, or four times as wide, if you wish.

Lastly, there is the double res version of the ampersand routines. They are contained in GR5, which you can BRUN to initialize. The memory allocation is the same as the language card version, GR2. Every effort was made to make all of the commands work identically in the new version, however there are a few differences. There are also some new commands available. Explanations of all the changes follow.

One of the most important differences is that many of the commands will not work unless the double res graphics screen is actually being displayed. The display switches affect the memory mapping of the auxiliary 64k, so this can cause problems... Also, there is only 1 page of double res graphics that can be displayed, so don't try to display page 2.

Some more oddities - the pack, unpack, and scroll commands will only work with page 1. All the other commands will work on the second page, you just can't see it without swapping it with page 1.

The magnification option only works with factors of 2 and 4. You can no longer magnify 8 times. Also shapes and text will not magnify properly in the current version. They will magnify if you do an &RESCAN after they've been plotted, the only problem is if you plot them within the magnified region while the magnify option is active.

Bshapes are no longer available. The &BSHAPE and &BTAB commands will be ignored. If you want multicolored shapes, you should edit your shape tables with the double res shape editor.

The &THICK and &THIN commands now select between plotting with a horizontal resolution of 140 and 16 colors (selected by &THICK, the default) and a resolution of 560 and 2 colors (&THIN). In &THIN mode, you can use 0 and 1 for black and white with the &COL command. Also note that the &CHKDOT command will return 0 or 1 if &THIN mode is active, or a color 0 through 15 otherwise. Also, color fills and the magnification option only work properly in &THICK mode.

&TRES,n will select &THIN (n=0) or &THICK (n=1) mode for text independently of all other graphics. To illustrate, here are the commands to get 20, 40 or 80 column text:

```
20 columns  &CHARMODE,0:&TWIDTH,1
40 columns  &CHARMODE,1:&TWIDTH,0
80 columns  &CHARMODE,0:&TWIDTH,0
```

&RES,n selects the range of coordinates that you use to specify locations on the screen. X coordinates range as follows:

&RES,0 0-559
&RES,1 0-279
&RES,2 0-139

Note that these commands only affect how you choose a spot on the screen - not what is drawn there. Thus, typing &RES,0:&THICK would let you specify 560 horizontal positions, but things would still be drawn with 140 dots of horizontal resolution.

&MIXCOL,n allows you to make new colors by blending four base colors together. The acceptable values for n are 0-65535, and can be chosen by the formula: n=color1 + 16*color2 + 256*color3 + 4096*color4.

&WIDTH,n will set the width of lines, boxes and circles. The default value is 1.

&STYLE, mask will let you set up a pattern for dotted or dashed lines and boxes. The mask can be whatever length you choose (within reason), and should consist of a series of spaces and asterisks enclosed in quotation marks. Some examples:

&STYLE," * * *" dotted line, every fourth dot plotted
&STYLE," *" dotted line, every fourth dot plotted
&STYLE," *** **** " dashed line

Other Penguin Software products:

The Graphics Magician
The Complete Graphics System
Short Cuts
Paper Graphics
Transitions
Magic Paintbrush
Additional Type Sets
Map Pack
The Data Analyzer
Home Data Manager
DISK arRANGER

Expedition Amazon
Ring Quest
Xyphus
Bouncing Kamungas
The Coveted Mirror
Pensate
The Spy Strikes Back
Minit Man
The Quest
Spy's Demise
Transylvania
Pie Man

penguin software ™

the graphics people