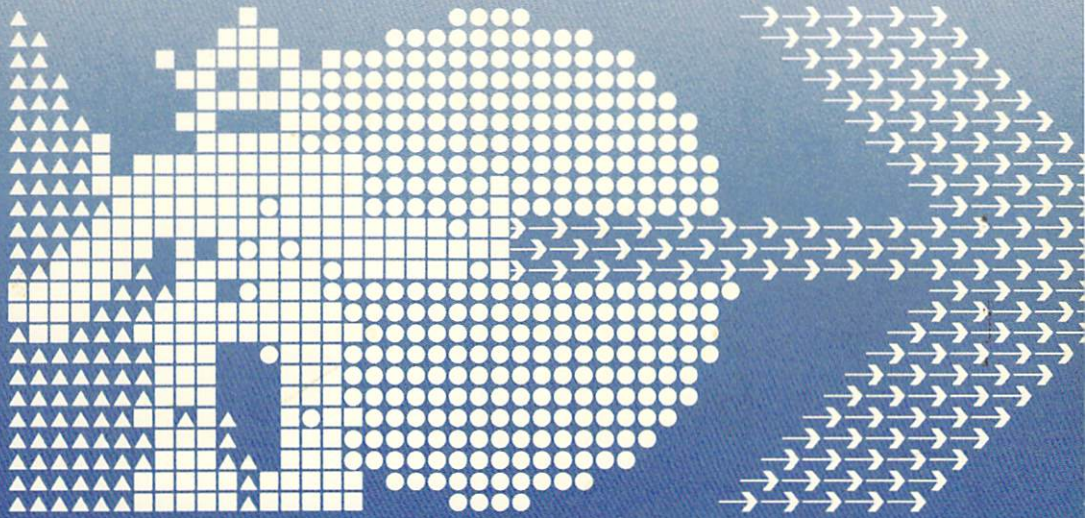
EPYX®

Programmers' BASIC Toolkit™

Assembly Language Graphics with BASIC Convenience



With Vorpal™ Fast Loader

For Commodore 64 and 128

EPYX®

Programmers' BASIC Toolkit

Assembly Language Graphics with BASIC Convenience

for Commodore 64® and 128™

TABLE OF CONTENTS

TABLES

FIGURES

PREFACE

Welcome to Programmers' BASIC Toolkit. This manual and the programs on the enclosed disk are your toolkit for creating colorful graphics with sound and action.

Programmers' BASIC Toolkit provides you with a complete package of programming tool enhancements for the built-in Commodore BASIC. The Programmers' BASIC Toolkit is a co-resident program: once it is loaded into your computer it becomes part of the Commodore BASIC. Suddenly you can add hi-speed assembly language power to your programs, using simple BASIC commands.

With the Programmers' BASIC Toolkit you get over a hundred new BASIC commands that can be applied to game design, computer animation, and business graphics.

You will find each new command fully documented in the Programmers' BASIC Toolkit manual, including demo and utility programs, and a convenient Command Reference Card.

Although the Programmers' BASIC Toolkit and manual are intended for more experienced programmers, if you are new to programming, or eager to learn, you'll find that the Toolkit commands are easy to use and the manual will guide you towards writing your own programs. The Toolkit even gives you onscreen editing of single characters or entire images.

To use the Programmers' BASIC Toolkit, you need

- Commodore 64 or 128
- Commodore 1541, 1571 or compatible disk drive
- Programmers' BASIC Toolkit disk
- Formatted disk to store your programs on
- TV or monitor (color, if possible)
- (Optional) Commodore compatible printer
- Joystick (to use the onscreen editor)
- (Optional) graphics tablet

So, turn to Section 1 — Introduction and take the first step towards the exciting new dimensions of Programmers' BASIC Toolkit.

Section 1 INTRODUCTION

The Programmers' BASIC Toolkit gives your Commodore 64 and 128 an array of new commands and features, while also supporting your existing sounds and graphics. Especially important is the Programmers' BASIC Toolkit onscreen editor that allows you to create and edit characters and sprites, in color, while you watch (see Section 6).

But before you leap on to that point, you need to know some things about this manual first, and, of course—but, not least-how to load the Programmers' BASIC Toolkit.

Boot and Run

The Programmers' BASIC Toolkit CREATE Command allows you to write "stand-alone" Boot and Run disks that automatically load Programmers' BASIC Toolkit ahead of any application. A Boot and Run program will work on any Commodore 64 or Commodore 128 in "64-Mode." It's important to keep in mind that only Boot and Run programs will work with the built-in Commodore 64 BASIC.

Programs written using Programmers' BASIC Toolkit commands will not run alone on Commodore 64 BASIC.

No license fee is required for applications written with the Programmers' BASIC Toolkit.

About This Manual

This manual contains six sections: introduction; descriptions of the operating system enhancements; descriptions of programming, graphics, and sound tools; and a reference summary.

This manual assumes that you are already familiar or have experimented with Commodore BASIC; however, if you are a beginner, you'll learn quickly and easily how to use the Programmers' BASIC Toolkit, and you'll be creating your own programs in no time.

Text Commands and Conventions

You're going to see various odd symbols and typefaces throughout this manual. The paragraphs below explain what these symbols and typefaces should mean to you.

Key names are shown exactly as they appear on your keyboard. For example, SHIFT means the Shift key; RUN STOP means the key labelled Run Stop, and so on. <RETURN> means to press the RETURN key.

Other angle brackets < > surround variables; but, don't type the angle brackets as part of the entry. For example, COPY SPRITE <sprite number> TO HIRES means that to use this command to copy sprite 4, you need to type **COPY SPRITE 4 TO HIRES**. Or, to copy sprite 6, you would type **COPY SPRITE 6 TO HIRES.**

Square brackets [ ] surround optional variables; but, don't type the square brackets either. For example, DISK [,<device number>] allows you to check a disk's error status, if your disk drive is device 8, you only need to type **DISK** to check it. However, if your disk drive is, say, device 9, then you would have to enter **DISK,9.**

Programmers' BASIC Toolkit commands appear in all capital letters. For example, COPY HIRES TO PRINTER makes your printer print whatever image is currently on your HIRES screen (screen modes are explained in Section 6).

Three dots (...) follow optional statement parameters that you can repeat as needed. For example, the SPRITE ANIMATE SPEED command allows you to choose the speed at which sprites move across the screen. You can use from 1 to 16 sprites, so the three dots (...) in the command syntax represent this fact. Entries and data that you must enter into the computer appear in boldface, and you must type them exactly as shown. For example, **LOAD "BOOTC",8,1** means to type those words in all capitals, include the quotations marks and commas, use spaces where shown, and don't include spaces where they aren't shown.

Function Keys

You can use the function keys to speed-up your programming. When you don't have a program running, issue the KEY ON command; then, when you press a function key, the command that corresponds to that key occurs. Table 1-1 lists the function keys and what they do.

Table 1-1. Function Keys

Key	Command Performed
f1	RUN plus RETURN
f2	BACKGROUND
f3	LIST plus RETURN
f4	SPRITE OFF
f5	DIR plus RETURN
f6	KEY LIST
f7	TEXT plus RETURN
f8	DISK

To use keys f2, f4, f6, and f8, hold down the SHIFT key while you press the function key.

Section 6 explains each command.

Loading the Programmers' BASIC Toolkit

Here's how to startup the Programmers' BASIC Toolkit:

1. Turn on the TV or monitor, the disk drive, and the Commodore 64.
2. Insert the Programmers' BASIC Toolkit disk into the disk drive. Make sure that the label faces up and enters the drive last.
3. Type **LOAD "*",8,1**
4. Press RETURN.

NOTE

Once you load the Programmers' BASIC Toolkit, 16K of memory remains in which you can create programs.

With the Epyx Fast Load Cartridge™

If you have an Epyx Fast Load Cartridge, the loading procedure is only slightly different:

Press and hold the **C=** key and the RUN STOP key.

NOTE

Programmers' BASIC Toolkit works on the Commodore 128 while it is operating in the 64 mode. Programmers' BASIC Toolkit does not work with BASIC 7.0 (128 mode BASIC).

When you finish for the day, be sure to remove the Programmers' BASIC Toolkit or program-storage disk before you turn off the disk drive and Computer.

Where To Go From Here?

If you're really, really an expert at BASIC programming, you can probably understand what's happening in the Programmers' BASIC Toolkit if you read "Editing" in Section 3, and look up all the commands you need in Section 6. But, it's better by far to go to Section 2 from here.

Section 2 OPERATING SYSTEM ENHANCEMENTS

Disk Drive

The disk drive, of course, is the device into which you place your 5 1/4-inch floppy disks. You load programs from the disk drive, and you save programs to the disk drive.

Normally, your disk drive is device number 8 on your system; all Programmers' BASIC Toolkit commands that require a disk drive number assume your disk drive is device 8 (default device). If your disk drive is device 9, you must enter the device number in these commands. For example, to see a disk's directory, type DIR if your disk drive is device 8, but, type DIR9, if your disk drive is device 9.

NOTE

The Programmers' BASIC Toolkit does not load from or save to cassettes.

Inputs and Outputs

Information must travel into the computer so you can create or run programs. Information must travel out of the computer so you can print copies or store your programs on disks.

When you type at the keyboard, use a joystick to move the cursor, or draw on a graphics tablet, you send information into the computer (input). The keyboard, joystick, and graphics tablet are input devices.

When the computer displays anything on its screen, or when you ask the computer to print something on your printer, the information is sent out from the computer to the screen or printer (output). The screen and the printer are output devices.

When you LOAD something from the disk drive into the computer, you are putting information into the computer (input). When you SAVE something to a disk drive, you are sending information out of the computer (output). A disk drive is both an input and an output device.

Joystick

The Programmers' BASIC Toolkit onscreen editing (see Section 3) requires that you have a joystick attached to port 2 on your computer. The joystick allows you to move the editing cursor to create characters and screens, select options from the editing screen, and fill your characters or screens with color.

Directory Listing

When you need to see what files you have saved on a disk

1. Place your disk in the disk drive.
2. Type **DIR** (if your disk drive is device 8)
   or
   type **DIR9** (if your disk drive is device 9).
3. Press RETURN.

The screen then shows you the name of every file on that disk.

You can print the disk directory by using a PRINT command (see PRINT, below).

Load

To use a program or to retrieve an image, you must move it from the disk into the computer by using the LOAD command. The LOAD command format is LOAD "<file name>",<device number>.

For example, to LOAD an onscreen editor, you would type **LOAD "SPRITE",8,1** if your disk drive is device 8, or **LOAD "SPRITE",9,1** if your disk drive is device 9.

Save

To store a program or image, you must move it from the computer onto a formatted disk by using the SAVE command. The SAVE command format is SAVE "<file name>",<device number>.

For example, for the file named ANGST, you would type **SAVE "ANGST", 8** if your disk drive is device 8, or **SAVE "ANGST", 9** if your disk drive is device 9.

Copy

The Programmers' BASIC Toolkit allows you to do many kinds of copying. You can copy text into a high-resolution screen, high-resolution graphics into a sprite, multicolor graphics into a sprite, and so on.

Print

You can print a file directory, a program listing, text, or an image on your printer.

To print a file directory, a program listing, or text, use the command COPY TEXT TO PRINTER. Whatever is currently on your TEXT screen is printed.

To print an image from the high-resolution (HIRES) screen, use the command COPY HIRES TO PRINTER. Whatever is currently on your HIRES screen is printed.

To print an image from the multicolor (MULTI) screen, use the command COPY MULTI TO PRINTER. Whatever is currenly on your MULTI screen is printed.

Section 3 PROGRAMMING TOOLS

The Programmers' BASIC Toolkit allows you to write graphics programs just as you would write any other BASIC program.

Enter your program after the READY prompt on the TEXT screen. If you make a mistake while typing, just use the INST DEL key to back over the error and type in the correct information. If you receive a syntax error after you try to run a program, you'll also be shown the line where the error occurred. Then, at the READY prompt, just retype your line. When you type **RUN** again, the Programmers' BASIC Toolkit executes your program.

The text below describes some of the Programmers' BASIC Toolkit's features.

Editing Your Programs

Programmers' BASIC Toolkit provides three commands to speed your program editing tasks. The "FIND" command locates arbitrary strings in your BASIC programs. It even includes BASIC keywords in the search. The "CHANGE" command will find and replace text strings in your BASIC programs, including BASIC keywords. The "REN" command resequences your program, making modifications easier to perform.

Creating an Auto-Booting Disk

To save time and effort—especially if you share the program you've created with your friends—you can copy your programs to disks that will boot automatically.

The CREATE command creates the auto-booting disk which then contains your Programmers' BASIC Toolkit program along with all the support code.

Here's how to create an auto-booting disk:

1. Load your program into memory.
2. Insert a blank disk in the drive.
3. Type **CREATE**

Printing to the Printer

When you want to print something at the printer, use the LPRINT command. It works in the same manner as the PRINT command, but your output goes to the printer instead of to the screen.

Using a Graphics Tablet

The Programmers' BASIC Toolkit allows you to use a graphics tablet. The commands PADX, PADY, and PADB instruct the Toolkit to read x- and y-coordinates and the buttons. The graphics tablet plugs into port 1 on the Commodore.

Making Conversions

You can convert numerical arguments from decimal to hexadecimal, octal, or binary, or convert hexadecimal, octal, or binary to decimal. Use the HEX$, OCT$, or BIN$ commands to convert from decimal. Place &H (for hexadecimal), % (for binary), or @ (for octal) in front of an argument to convert it to decimal.

Sorting Arrays

You can sort arrays in ascending or descending order. The commands are SORT A$,[<direction>] or SORTA%,[<direction>] or SORTA,[<direction>]. <direction> can be A (for ascending) or D (for descending). Ascending is the default.

Getting Help

Whenever you see the READY prompt, you can type **HELP**, then press RETURN, for a list of all the commands along with a summary description.

Section 4 GRAPHICS TOOLS

When you use the Programmers' BASIC Toolkit, you can, of course, create all your backgrounds, fonts, and sprites by writing Programmers' BASIC programs. But, you don't have to: the Programmers' BASIC Toolkit includes two onscreen editors — a background/font editor and a sprite editor. By using a joystick to control the cursor, you can actually create backgrounds, fonts, and sprites as your watch. You can then save your images to disk and recall them as you need them in your programs.

The background/font and sprite editors are discussed in detail further on in this section; but, first, you should know a few more things about Programmers' BASIC Toolkit.

The Three Screen Modes

The Programmers' BASIC Toolkit operates in three screen modes: TEXT, HIRES, and MULTI modes. Which stand for:

TEXT	where you type text and enter programs.
HIRES	high-resolution screen.
MULTI	multicolor screen.

How the Hires and Multi Screens Are Constructed

Unlike television images, which consist of lines across the screen, graphic images are created on a screen that is made up of dots (which are made up of pixels). Each dot can be either on or off. When you create an image, that is precisely what you are doing: turning dots on and off.

Each dot has a definite location, and that location is defined by x and y coordinates, just as if the screen were covered by a grid.

HIRES Screen

The HIRES screen consists of 200 dots vertically and 320 dots horizontally.The screen's leftmost lower corner is position x = 0,y = 0, or (0,0). The center of the screen is at x = 160,y = 100, or (160,100). The screen's rightmost upper corner is at x = 320,y = 200, or (320,200). See Figure 4-1.

For example, if you wanted to draw a box whose leftmost lower corner was located at position (0,0), then you'd enter the BOX <corner x>,<corner y> command as **BOX 0,0**.

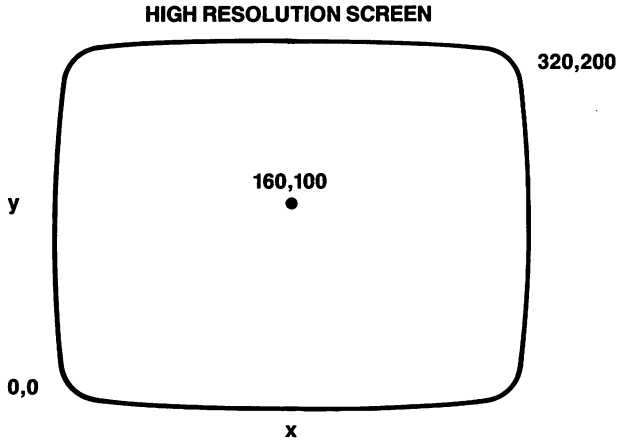


Figure 4-1. HIRES Screen

MULTI Screen

The MULTI screen consists of 200 dots vertically, but only 160 horizontally. Each dot on the MULTI screen is two pixels wide. So, center for the MULTI screen is at x = 80,y = 100, or (80,100). See Figure 4-2.

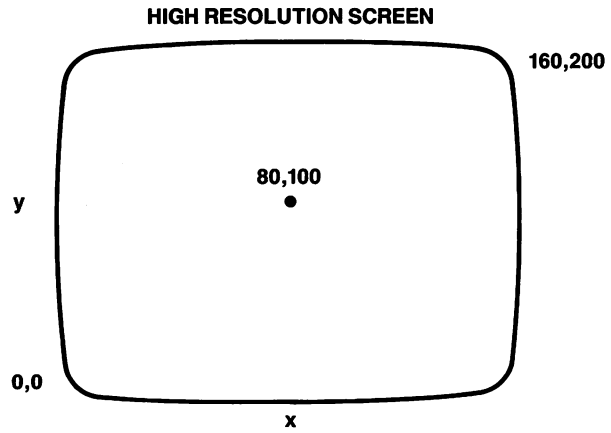


Figure 4-2. MULTI Screen

Color

The Programmers' BASIC Toolkit gives you access to 16 colors: black (0), white (1), red (2), cyan (3), purple (4), green (5), blue (6), yellow (7), peach (8), brown (9), pink (10), gray 1 (11), gray 2 (12), light green (13), sky (14), and gray 3 (15). (These color names and numbers are also listed in Appendix C.)

TEXT Screen

You can change the background color and the border color on the TEXT screen.

The first screen that you see after you load the Programmers' BASIC Toolkit has a black background with a gray border.

To change the background color, after the READY prompt, either type **BACKGROUND** <color number/name> then press RETURN; or, press the f2 key (SHIFT plus f1). When you press the f2 key, the word **BACKGROUND** appears.

Type the color number or name; then press RETURN.

To change the border color, after the READY prompt, type: **BORDER** <color number/name>, and press RETURN.

Give it a try:

Press f2 (SHIFT plus f1)
Type: **4**
Press RETURN
Type: **BORDER 6**
Press RETURN

You should now have a purple screen with a blue border.

To return to the black screen with the gray border:

Type: **BACKGROUND 0**
Press RETURN
Type: **BORDER 11**
Press RETURN.

HIRES Screen

You can use two colors on the HIRES screen: one for the background and one for the foreground.

To set colors on the HIRES screen, use the HIRES COLOR <foregroundcolor> ON <backgroundcolor> command.

Try this, enter:

```
10      HIRES
20      HIRES COLOR BLACK ON PINK
30      BOX 100,40 XYSIZE 100,100
RUN
```

You should have a black box on a pink background. Type **CLEAR** to clear the screen.

To return to the TEXT screen, type **TEXT**.

MULTI Screen

You can use any color on the MULTI screen: one for the background and the rest for the foreground.

```
10      MULTI
20      BACKGROUND BROWN
30      COLOR WHITE
40      LINE 100,100 TO 200,200
50      COLOR RED
60      LINE 10,180 TO 200,30
70      COLOR GREEN
80      LINE 160,200 TO 160,0
RUN
```

Graphics Primitives

Graphics primitives are the most basic components from which you can build images. The primitives include: lines, circles, boxes, dots, and filling in with color. You can use most primitives (except FILL) in HIRES or MULTI mode.

Lines

If you want to draw a line, use the LINE <x coordinate>,<y coordinate> TO <x coordinate>,<y coordinate> command.

For example, try this, enter:

```
10      HIRES
20      HIRES COLOR BLUE ON PURPLE
30      LINE 0,0 TO 320,320
RUN
```

You should see a diagonal blue line on a purple background. Type **CLEAR** to clear the screen.

Here's another example. Enter:

```
10      MULTI
20      BACKGROUND BLUE
30      COLOR RED
40      LINE 0,100 TO 200,200
RUN
```

You should see a red line on a blue background. Type **CLEAR** to clear the screen.

Dots

If you need dots on the screen, use the DOT <x>,<y>;<x>,<y>;<x>,<y>... command.

For example, enter:

```
10      HIRES
20      HIRES COLOR BLACK ON PURPLE
30      DOT 100,160;110,150;120,140;130,130;140,120
RUN
```

You should see five black dots on a purple background. Type **CLEAR** to clear the screen.

Circles

If you want to draw circles, use the CIRCLE <center x>,<center y> XYSIZE <x size>,<y size> command.

For example, enter:

```
10      HIRES
20      HIRES COLOR RED ON WHITE
30      CIRCLE 50,60 XYSIZE 10,30
RUN
```

You should see a red circle on a white background. Type **CLEAR** to clear the screen.

Fill

To fill in a circle, box, or other geometric shape with color, use the FILL <x>,<y> command. FILL only works on the HIRES screen. If you specify a point that lies outside the shape that you want to fill, FILL colors the area surrounding the shape.

For example, enter:

```
10      HIRES
20      HIRES COLOR BLUE ON PINK
30      BOX 100,100 XYSIZE 50,50
40      COLOR BLUE
50      FILL 101,101
RUN
```

The box should be filled with the color blue. Type **CLEAR** to clear the screen.

On the other hand, enter:

```
10      HIRES
20      HIRES COLOR WHITE ON BROWN
30      BOX 100,100 XYSIZE 50,50
40      BOX 75,75 XYSIZE 100,100
50      FILL 76,76
RUN
```

You should see two boxes, one inside the other. The inside box remains empty, while the outside box fills in like a border around it. Type **CLEAR** to clear the screen.

If you change line 50 to read:

```
50      FILL 101,101
```

The center box fills in. Type **CLEAR** to clear the screen.

The Sprite Editor

Sprites are designs that you can animate and move around on the screen. You can create nearly any shape that you have in mind, and you can use six sprites on the screen at one time. You can also store 32 sprite shapes. Each sprite can accept any one of those 32 shapes, or all six — or fewer — can have the same shape.

To help you create sprite shapes, the Programmers' BASIC Toolkit has an onscreen sprite editor.

Loading the Sprite Editor

To load the sprite editor, make sure that you have the Programmers' BASIC Toolkit disk in your disk drive. Then, from the BASIC version 2 prompt,

1. Type: **LOAD "SPRITE",8,1**
2. After a while, the sprite editor screen appears (Figure 4-3).

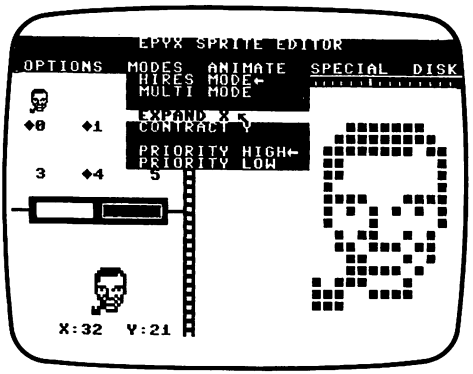


Figure 4-3. Sprite Editor Screen

What's on the Screen

The title EPYX SPRITE EDITOR appears at the top of the screen. Underneath are five headings: OPTIONS, MODE, ANIMATE, SPECIAL, and DISK. Notice that the cursor has become a flashing arrow.

Six blank sprites (0-5) appear on the left half of the screen below the headings.

Below the sprites is the color well. It displays two colors when you use HIRES mode and four when you use MULTI mode.

The display box occupies the screen's lower left half. The x and y indicate x- and y-coordinates relative to the upper left corner of the display box.

The space to the right is the editing box.

The cursor is a blinking arrow everywhere except when you move it inside the editing box: then it becomes a box with the current color in its middle.

The descriptions that follow explain the fine points of the sprite editor. Take time to read them, and especially, take time to experiment with the system!

Editing Sprites

The six blank sprites in the upper left corner are the sprites that you are going to edit.

When you load the sprite editor, these sprites are numbered 0 through 5. To change a sprite number, move the cursor to the number, click the button, and press the joystick to the left or right: the numbers cycle from 0 to 31. Stop pressing the joystick when you reach the number you want.

To start editing a sprite, move the cursor to the sprite number, and click the button: the selected sprite number flashes. Move the cursor into the editing box (notice that the cursor becomes a small box itself with the current color in its middle). To draw, move the joystick and press the button wherever you want to plot a mark. When you draw in the editing box, your creation also appears above its sprite number and in the display box, if it has been added.

The Color Well

The color well appears to the left between the numbered sprites and the display box.

In HIRES mode, the color well shows two areas: the left area changes the background color, the right changes the sprite's color.

In MULTI mode, the color well shows four areas: the leftmost area changes the background color, the others change the sprite's colors.

To change the background color, move the cursor to the leftmost area, hold the button down, and press the joystick left or right: the colors cycle.

To change a sprite color, move the cursor to the rightmost area (in HIRES mode) or to one of the three remaining areas (in MULTI mode), hold the button down, and press the joystick left or right: the colors cycle. When you have the color you want, move the cursor into the editing area and draw a sprite or go over the current one.

The Display Box

The sprite you're editing can be placed in the display box along with any other sprites you've placed there.

Although the division is invisible, the display box contains two areas—left and right—that allow you to move a sprite or a group of sprites around inside the box.

To move the current sprite, place the cursor in the left side of the display box, and hold down the button. Now you can move the sprite around, and the x and y at the bottom of the screen show you the sprite's coordinates.

To move a group of sprites, place the cursor in the right side of the display box, and hold down the button.

The OPTIONS Column

Place the cursor on the word OPTIONS, and hold down the button: you'll see the option selections. They are:

SPRITE X,Y	Moves the sprites from the display box onto the user screen. Use the joystick to move the sprites around, then click the button when the sprites are located where you want them to be. The x- and y-coordinates for the sprites appear at the top of the screen (in decimal or hex). Click the button again to resume normal operations.
SOUND OFF/ON	Toggles the sound effects off and on.
OOPS	Restores whatever you changed last to its previous state.
DECIMAL	Numbers printed or entered are in decimal.
HEX	Numbers printed or entered are in hexadecimal.
QUIT	Returns you to Commodore BASIC version 2.

The backarrows point to the options currently in effect. When you first load the sprite editor, DECIMAL is the current option.

To select an option, move the cursor to the option you want, and release the button.

For example, move the cursor to HEX and let go of the button. Any numbers printed or entered are now in hexadecimal.

NOTE

The x,y coordinate system for positioning sprites on the screen starts at the lower lefthand corner in Toolkit BASIC and at the upper left-hand corner in the SPRITE EDITOR.

The MODE Column

Place the cursor on the word MODE, and hold down the button. The MODE selections are:

HIRES MODE	Places the current sprite in high-resolution mode.
MULTI MODE	Places the current sprite in multi-color mode.
EXPAND/CONTRACT X	Expands or contracts the current sprite in the x direction, but only if the current sprite appears in the display box.
EXPAND/CONTRACT Y	Expands or contracts the current sprite in the y direction, but only if the current sprite appears in the display box.
PRIORITY HIGH	Marks the current sprite as having higher priority than the background, when the sprite is on the user screen.
PRIORITY LOW	Marks the current sprite as having lower priority than the background, when the sprite is on the user screen.

To change the MODE selection, put your cursor on the selection you want to use, and let go of the button. For example, to change from HIRES to MULTI mode, place the cursor on MULTI and release the button. The sprite editor screen reappears in multi-color mode.

The ANIMATE Column

Place the cursor on the word ANIMATE, and hold down the button. The ANIMATE selections are:

IN MULTIPLE BOX	Sets up animation in the multiple box. The program prompts for animation speed and animation steps. To terminate the selection, press f7 before making an entry. To end a list, enter a period (.) when the program asks for another animation step. To stop the sprite animation, press the button (see SPRITE ANIMATION).

IN USER SCREEN	Sets up animation for the user screen. The program prompts for starting x- and y- coordinates, movement speed, animation speed, and animation steps. To terminate the selection, press f7 before making an entry. To end a list, enter a period (.) when the program asks for another animation step. To stop the sprite animation, press the button: the program returns you to the sprite editor screen (see SPRITE ANIMATION).
REPEAT LAST	Repeats or edits the last animation set up for the user screen.
ADD SPRITE	Places the current sprite in the display box (see SPRITE ANIMATION). A diamond (♦) marks sprites that have been added.
REMOVE SPRITE	Removes the current sprite from the display box.
REMOVE ALL	Removes all the sprites in the display box.
STORE MULTIPLE	Stores the shape numbers and the relative positions of the sprites in the display box. You can store up to 16 multiple definitions.
FETCH MULTIPLE	Recalls a multiple shape definition. To cycle through the multiple definitions, press the joystick left or right. When you come to the multiple that you want to recall, click the button. To leave this selection, move the cursor to the left until the message CLICK TO ABORT appears at the top of the screen, and click the button.
COPY MULTIPLE	Copies a multiple definition into other sprite shapes. To cycle through the definitions, press the joystick left or right. When you come to the multiple that you want to to copy, click the button. To leave this selection, move the cursor to the left until the message CLICK TO ABORT appears at the top of the screen, and click the button.

REMOVE MULTIPLE — Erases a multiple definition from storage. To cycle through the definitions, press the joystick left or right. When you come to the multiple that you want to erase, click the button. To leave this selection, move the cursor left until the message CLICK TO ABORT appears at the top of the screen, and click the button.

To make a selection from the ANIMATE column, put your cursor on the selection you want to use, and let go of the button. For example, to add the current sprite to the display box, move the cursor to ADD SPRITE, and release the button.

The SPECIAL Column

Place the cursor on the word SPECIAL, and hold down the button. The SPECIAL selections are:

OR MASK — Logically ORs the choosen shape onto the current sprite. The message OR FROM SHAPE #0 appears at the top of the screen. Press the joystick right or left to select the shape number. To leave this selection, press the joystick to the left until the message CLICK TO ABORT appears.

AND MASK — Logically ANDs the choosen shape onto the current sprite. The message AND FROM SHAPE #0 appears at the top of the screen. Press the joystick right or left to select the shape number. To leave this selection, press the joystick to the left until the message CLICK TO ABORT appears.

XOR MASK — Logically XORs the choosen shape onto the current sprite. The message XOR FROM SHAPE #0 appears at the top of the screen. Press the joystick right or left to select the shape number. To leave this selection, press the joystick to the left until the message CLICK TO ABORT appears.

COPY — Copies shape data onto the current sprite. The message COPY FROM SHAPE #0 appears at the top of the screen. Press the joystick right or left to select the shape number. To leave this selection, press the joystick to the left until the message CLICK TO ABORT appears.

CLEAR	Clears the current sprite to all zeroes.
CLEAR SET	Clears all 32 sprites and the multiple shape list.
SCROLL	Scrolls the current sprite in any direction. Use the joystick to move the sprite, and click the button when the sprite appears as you want it to.
FLIP HORIZ	Flips the current sprite horizontally around its center.
FLIP VERT	Flips the current sprite vertically around its center.

To use a SPECIAL selection, put your cursor on the selection you want to use, and let go of the button. For example, to flip the current sprite vertically, move the cursor to FLIP VERT, and release the button.

The DISK Column

Place the cursor on the word DISK, and hold down the button: the DISK screen appears. The items on the DISK screen are:

LOAD	To pick a file to load, point the cursor at a name in the directory box, and click the button. The name then appears in the blank bar in the command box, and also in reverse in the directory box. The SPRITE or CHRSET box turns blue depending on the file type. If the CHRSET box turns blue, it also indicates SCREEN or CHRSET depending on the file type (screen data or character set data). The load box turns blue, meaning that it is now active. To load the selected file, place the cursor on the load box, and click the button.
SAVE	Saves a file. Place the cursor on the save box, and click the button. The cursor appears in the blank bar at the top of the command box. Enter the filename, and press RETURN. The file is saved out to the disk. If you decide not to save the file after all, press RETURN when the blank bar is empty.
FORMAT	Formats a disk. Place the cursor on the format box, and click the button. The screen turns orange, and the program asks you to enter a disk name and ID. Place a blank disk in the drive, then

enter the disk name, a comma, and a two-character ID (for example, **TKBASIC,F0**), then press RETURN. If you change your mind about formatting a disk, press RETURN when no characters are entered.

DEVICE/ DRIVE NUMBER — The 08:0 box tells you that the current device number is 8 and the current drive number is 0.

To change the device number, place the cursor on the device number, hold down the button, and move the joystick left or right. The device number cycles between 8 and 11. To change the drive number, place the cursor on the drive number, hold down the button, and move the joystick left or right. The drive number cycles between 0 and 1.

EDITOR — Returns you to the background/font editor

DIR — Reads the directory from a new disk. Insert a new disk in the disk drive, and click the button.

Sprite Editor Fast Keys

Fast keys allow you to edit sprite shapes more quickly than you can by moving the cursor around the sprite editor screen. After some experience with the sprite editor, you'll feel as at home with these keys as with the screen. Table 4-1 lists the fast keys.

Table 4-1. Sprite Editor Fast Keys

Key	Description
	UNSHIFTED KEYS
1	The current editing color is the screen color. Editing plots 0 in HIRES mode, and 00 in MULTI mode. This action is like clicking the leftmost color well.
2	The current editing color is the sprite color. Editing plots 1 in HIRES mode, and 10 in MULTI mode. This action is like clicking the second color well.

3	If the current sprite is in MULTI mode, the current bitpair is 01. This action is like clicking the third color well.
4	If the current sprite is in MULTI mode, the current bitpair is 11. This action is like clicking the fourth color well.

SHIFTED KEYS (press SHIFT and the key)

1	Increments screen color.
2	Increments sprite color.
3	Increments multi-color 1.
4	Increments multi-color 2.

CONTROL KEYS (press CTRL and the key)

1	Decrements the screen color.
2	Decrements the sprite color.
3	Decrements multicolor 1.
4	Decrements multicolor 2.

FUNCTION KEYS

f1	When you select IN USER SCREEN from the ANIMATE column, the program asks for starting and ending x- and y coordinates. f1 allows you to jump to the user screen, position the sprite by using the joystick, and click the button to enter the x- and y-coordinates.
f3	Toggles between the user screen and the sprite editor screen. When you select the user screen, you can use the joystick to move the sprites.
f5	Increments the border/menu color.
f7	Adds the current sprite to the display box.
f8	Removes the current sprite from the display box.

OTHER KEYS

CRSR	Scroll the current sprite in the direction pointed to by the key.
H	Flips the current sprite horizontally around its center.
V	Flips the current sprite vertically around its center.

Sprite Animation

How to Enter an Animation Sequence

There are two types of animations you can do, animating in the display box or animating in the user screen. You can specify an animation sequence of single sprite names if you wish to animate a single sprite, or a sequence of stored multiple definitions if you wish to animate a multiple sprite. In addition, if you choose to animate your sprite(s) on the user screen (against your loaded background created with the background/font editor), you can make your sprite(s) move across the screen. Use the *disk* menu to load a background screen.

When you select ANIMATE IN DISPLAY BOX, you are prompted for an animation speed and a series of animation steps (up to 15). The speed is the number of "jiffies" (one sixtieth of a second) between animation step changes. The fastest speed is 1 and the slowest speed is 255. The animation steps you choose will either be single shape names (0 to 31) or multiple shape definitions (0 to 15), depending on whether you wish to animate a single sprite or a multiple sprite combination. The animator will decide which you want depending on the number of sprites you have in the display box. If you have no sprites in the display box, or one sprite, it assumes that you want to animate just a single sprite. If you have two or more sprites in the display box, it assumes you want to animate a sprite combination. Keep entering steps until you have entered 15 (the maximum), or you have entered a period (.) to signal the end of the animation sequence.

If you want to edit your animation table before you enter the last step (or terminating period) you may do so by using the cursor up and down keys to go from one entry in the table to the next. Once you position the box cursor on the entry you wish to change, simply type in a new number and press return. If you accidentally type over a value you have moved to, you can restore the old value by moving the cursor away from that entry. However, if you have entered that new number by pressing return, it is there forever unless you position the cursor over it again and re-enter the number you want. Once you are finished editing, you can go to the actual animation by moving the cursor to the last unentered line and entering a terminating period (or entering the 15th step in the table).

If you do not wish to animate a sprite, you may escape by typing f7.

The values that you enter into the animation table will be assumed to be in the current RADIX (decimal or hexadecimal) that is selected in the OPTIONS menu. If you wish to force the number you are entering to a specific radix no matter what the default, you can type a $ in front of the number (to force it to be hexadecimal) or a # in front of the number (to force it to be in decimal).

When you select ANIMATE IN USER SCREEN, you are prompted for a starting X and Y location, an ending X and Y location, the number of steps to take between those locations, the speed at which those steps are taken, and the animation speed and animation steps as indicated above in ANIMATE IN DISPLAY BOX. However, unlike the on-screen animation which only allows one animation sequence to be entered, you may enter up to eight "paths" (or sequences) for your sprite(s) to take in the user screen, each one with its own animation sequence.

When prompted for the X and Y starting and ending locations, you should enter the true sprite locations, with X being between 0 and 343, and Y being between 0 and 249. If you don't know the true sprite location, or want to position a sprite perfectly within your user screen, you may enter f1 at the prompt. The screen will then change to the user screen, and your sprite(s) will appear in the upper left hand corner. You may move your sprite(s) around with the joystick until you have them positioned properly, and then click the trigger. The screen will then go back to the editing screen, and your chosen location will be entered for you.

When you are prompted for the number of steps, enter a number between 1 and 255. This is the total number of steps that will be taken between the starting and ending locations you have entered. For example, if your starting X location is 100, and your ending X location is 200, and you enter 50 steps, the sprite will move in the x direction by 2 pixels at a time (200 minus 100 divided by 50). The same number of steps will affect the Y direction movement as the X. Your number of steps needn't be evenly divisible into your total movement, either. You can have 37 steps between 112 pixels if you like.

When you are prompted for the movement speed, you can enter a number between 1 and 255. Like the animation speed, this number reflects the number of "jiffies" between step movements of the sprites.

NOTE

Sprite shape numbering starts at "0" in the SPRITE EDITOR and at "1" in Toolkit BASIC.

After you have entered your path, you will then be prompted for the animation speed and animation steps as in the display box animation. You may terminate your sequence and initiate animation by typing a period as before. However, you have a few more options when you are animating in the user screen. If you type a plus sign ("+"), it will terminate the current entry table, clear the slate, and allow you to enter an additional path which will be followed once the first one is done. All in all you may enter up to eight paths. If you type an "R", it will terminate entry as if you typed a period, but will make the animation start over from the beginning once it has traced through its path(s). As in animating in the display box, you can stop the animation and return to the editing screen by clicking the joystick trigger.

If you want to specify a number of paths for your sprite(s) to take, but you want them to use the same animation sequence without having to type it in at every table, you can use the f2 key to copy the previous table's animation sequence. The f2 key will only work from the ANIMATION SPEED prompt, however, if you have already passed it you can back up to it by using the cursor keys.

You may edit your table before sending it off to be executed just like in the display box animation, except that you can also back up and go forward between your eight tables by using the f3 and f5 keys. f3 will jump to your previous table (unless you are already on the first table) and f5 will jump to the next table (unless you are already on your last table). In order to start the animation, however, you must go back to your last table when you are through editing and enter a terminating character (either a period or an "R", or a plus sign to create a new table) at the end of your last animation sequence.

As in the display box animation, you may abort this feature by typing the f7 key at any prompt.

Repeat Last

Lets you repeat the last animation you did, whether it was in the display box or in the user screen. You may also edit your animation table(s) before you re-start the animation. To initiate animation from REPEAT LAST you can go to the last entry in the last table and type a terminating character, or you may simply type f8 at any prompt to start the new animation going.

NOTE

The "+" and "–" keys can be used in the SPRITE EDITOR to increase and decrease animation speed. Also the shifted "+" and "–" will increase and decrease movement speed. Use these keys while the animation is running to fine-tune it. "Repeat last" will then contain the modified speeds.

Table 4-2. Sprite Animation Keys

Key	Description
f1	If in "starting x,y" or "ending x,y," will jump to user screen, sprite(s) will be displayed and be under joystick control. Position sprite and click button. Position will be entered into table.
f2	At the "ANIMATION SPEED" PROMPT, copies the previous PATH'S animation [shape] sequence (speed & names).
f3	Edit next page (in user screen only).
f5	Edit previous page (in user screen only).
f7	Quit
f8	In "REPEAT LAST", will initiate animation.
"."	Terminates entry (after at least 1 sprite shape name has been entered).
"+"	Terminates current page, starts new one (up to 8 maximum).
"R"	Terminates entry and sets repeat animation flag.
"$"	Force hex radix.
"#"	Force decimal radix.
crsr dwn	Edit next element (with wrap around)
crsr up	Edit previous element (with wrap around)

The Background/Font Editor

When you need to create backgrounds for your sprites to move around on, or when you want to create a set of new fonts (letter shapes), the Programmers' BASIC Toolkit provides an onscreen background/font editor.

NOTE

To use the background/font editor, you must have a joystick plugged in to port 2 on your Commodore computer.

Loading the Background/Font Editor

To load the background/font editor, make sure that you have the Programmers' BASIC Toolkit disk in your disk drive. Then, from the BASIC version 2 prompt,

1. Type: **LOAD "BGFONT",8,1**
2. After a while, the background/font editor screen appears (Figure 4-4).

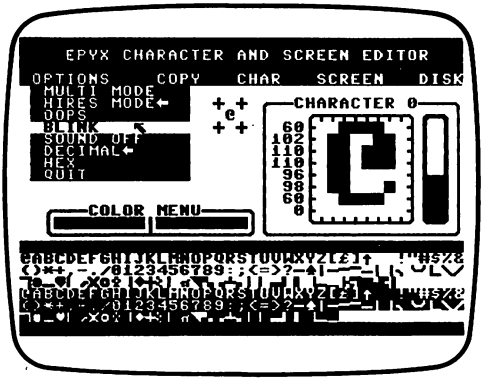


Figure 4-4. Background/Font Editor

Take your time and read the descriptions that follow. Then start experimenting with the background/font editor on your own. You'll have a terrific time.

What's on the Screen

The screen's title appears at the top: EPYX BACKGROUND/FONT EDITOR. Just below the title, then, you'll see five column headings, which are: OPTIONS, COPY, CHAR, SCREEN, and DISK. Also notice that the cursor has become a flashing arrow.

Three Kinds of Cursors and What They Do

Try out the joystick. Everything you need to do on the background/font editor screen, you can get to by moving the cursor and clicking the button.

But, you'll also notice that the cursor changes shape.

Try this: move the cursor into the middle left area that shows a line of color squares (these are the Programmers' BASIC Toolkit 16 color selections). When the cursor is over one of these colors, the cursor turns into an empty box. However, when you move the cursor into the large character display on the right side of the screen, the cursor becomes a box filled with the current drawing color.

Move the cursor back into the line of color squares. Place the cursor over a color, and click the button. Now, move the cursor down into the blank area below the character strip. The cursor changes into an empty box with four little boxes attached to its corners. This cursor carries color for you.

Now move the cursor into the first line of the character strip, and place it over the A. Click the button. The large character display becomes A with the display code number (in decimal) shown right below it. You can edit any character by placing the cursor over that character in the character strip and clicking.

When you select a character, something else happens, too. Move the cursor down into the blank screen area. You'll notice that the cursor is carrying the character you selected. If you want to deposit this character anywhere in the blank area, click the button.

After you've deposited a few characters. Go back to the color strip, pick up a color (place the cursor over it and click the button). Then go back down to the blank area, place your cursor over one of the characters you deposited there, and click. The character turns into the color you selected.

The Character Editing Box (The Large Character Display)

Move the cursor into the character editing box (the cursor becomes an empty square). Whenever you click the button, the character in the editing box changes, and its image in the character strip changes along with its image in the small display box to the left of the character editing box.

The small display box is there to show you how your edited character looks, by itself, in its normal size.

The number at the top of the character-editing box is the display code for the character you're editing. You can have this number in decimal or hexadecimal. (Toggling the number is discussed under "The OPTIONS Column," below).

The numbers to the left of each "line" in the character-editing box are the values for each of the eight bytes that make up a character (see the Commodore Programmers' Reference Guide for further details).

Colors

To the right of the character-editing box, and really a part of it, is a vertical strip (the vertical color well), that shows two colors (in HIRES mode) or four colors (in MULTI mode).

The top color on the color well controls background color. To cycle through the colors, move the cursor into the top color well, hold down the button, and move the joystick left or right. Whatever color you stop at is the background color.

The bottom half of the color well controls foreground color (in HIRES mode) or color 1, color 2, and color 3 (in MULTI mode). This half of the color well operates in the same manner as the top: move the cursor into the bottom color well, hold down the button, and move the joystick left or right. Whatever color you stop at is selected.

The OPTIONS Column

Place the cursor on the word OPTIONS, and hold down the button: you'll see the option selections. They are:

MULTI MODE	Places the editor in MULTI mode
HIRES MODE	Places the editor in HIRES mode
OOPS	Restores the last item edited to its previous state
BLINK	Choose the character that you want to blink from the character strip. That character then blinks everywhere it exists on the work-screen until you select BLINK again to turn it off. Note that if the characters you've choosen to blink are the same color as the background, you won't see them.

SOUND ON/OFF	Toggles any sound on or off
DECIMAL	Shows the character display code in decimal
HEX	Shows the character display code in hexadecimal
QUIT	Returns you to BASIC version 2

The backarrows point to the options currently in effect. When you first load the background/font editor, HIRES MODE and DECIMAL are the current options.

To change options, move the cursor to the option you want, and let go of the button.

For example, move the cursor to MULTI MODE and let go of the button. The character in the character editing box changes color, and the vertical strip next to the character editing box shows four color bars.

The COPY Column

Place the cursor on the word COPY, and hold down the button. The COPY selections are:

OR MASK	Lets you select another character to OR with the character that you're currently editing
AND MASK	Lets you select another character to AND with the character that you're currently editing
XOR MASK	Lets you select another character to XOR with the character that you're currently editing
CHARACTER	Copies a character from the character strip onto the current character
UPPERCASE FONT	Puts the uppercase ROM font in the character strip
LOWERCASE FONT	Puts the lowercase ROM font in the character strip

To change the COPY selection, put your cursor on the selection you wish to use, and let go of the button.

For example, to place the lowercase font in the character strip, move the cursor to FONT B, and let go of the button.

The CHAR Column

Place the cursor on the word CHAR, and hold the button down. The CHAR selections are:

FLIP VERT	The character you're currently editing flips over along the vertical axis
FLIP HORIZ	The character you're currently editing flips over along the horizontal axis
SCROLL	You can use the joystick to scroll the character that you're currently editing.
CLEAR CHAR	Clears the current character
CLEAR ALL	Clears all the characters in the set
FILL COLOR	Makes all characters in the character strip the same color.

For example, place the cursor on FLIP HORIZ, and let go of the button. The character in the character editing box turns upside down.

The SCREEN Column

The SCREEN is the blank area under the character strip.

Place the cursor on the word SCREEN, and hold the button down. The SCREEN selections are:

FLIP VERT	The image in the SCREEN flips along the vertical axis
FLIP HORIZ	The image in the SCREEN flips along the horizontal axis
SCROLL	The image in the SCREEN scrolls
FILL COLOR	The cursor jumps down to the color strip, and once you've selected a color, the SCREEN fills with that color.

For example, move the cursor to FILL COLOR, and let go of the button. The empty square cursor appears at the color strip. Place the cursor over any color, then click the button. Next move the cursor down to the SCREEN. The SCREEN has filled with the color you selected. Note that the color you selected from FILL COLOR did not affect the character in the character editing box.

The DISK Column

Place the cursor on the word DISK, and click the button. You now have the DISK display on your screen.

The box to the left shows the current disk's directory. If there are more entries than fit in the box, you can scroll to see the other entries. Place the cursor on the top bar and click the button: the directory scrolls up. Place the cursor on the bottom bar and click the button: the directory scrolls down.

To the right you'll see a blank bar, and underneath boxes labelled LOAD, SAVE, 08:0, FORMAT, CHRSET, SCREEN, EDITOR, and DIR.

LOAD — To choose a file to load, place the cursor at a name in the directory box, and click the button. The name then appears in the blank bar in the command box, and also in reverse in the directory box. The SCREEN or CHRSET box turns blue to tell you what the file type is. The LOAD box also turns blue to indicate that it is now active. To load the file you've selected, click the button.

SAVE — To save a file, place the cursor on the SAVE box, and click the button. Both SCREEN and CHRSET turn blue: you need to indicate whether you are saving a screen or character set. Place the cursor on SCREEN or CHRSET, and click the button: a cursor appears in the blank bar at the top of the command box. Enter the filename, and press RETURN. The file is saved out to the disk. If you change your mind about saving a file, press RETURN when the blank bar is empty.

FORMAT — To format a disk, place the cursor on the FORMAT box, and click the button. The screen turns orange. The program asks for the disk name and ID. Insert a blank disk in the drive, and enter the disk name, a comma, and the two-character disk ID (for example, TKBASIC,f1), then press RETURN. If you decide not to format a disk, press RETURN when there are no characters entered.

DEVICE/ DRIVE NUMBER	The 08:0 box tells you that the current device number is 8 and the current drive number is 0. To change the device number, place the cursor on the device number, hold down the button, and move the joystick left or right. The device number cycles between 8 and 11. To change the drive number, place the cursor on the drive number, hold down the button, and move the joystick left or right. The drive number cycles between 0 and 1.
EDITOR	Returns you to the background/font editor
NEW DISK	Reads the directory from a new disk. Insert a new disk in the disk drive, and click the button.

Type-in-Mode

When the cursor is in the work-screen and character-plotting mode (that is, the cursor is carrying a character, not a color), you can type characters onto the screen. All the printing keys work, as well as the cursor keys, RETURN, and RVS-ON and RVS-OFF.

Background/Font Function Keys

Table 4-3 lists the function keys available with the background/font editor and describes what they do.

Table 4-3. Background/Font Editor Function Keys

Key	Description
f1	(Only operates in the work-screen) In character-plotting mode, the cursor picks up the character under it. In color-plotting mode, the cursor picks up the color under it.
f2	(Only operates in the work-screen) Toggles between the character-plotting mode and the color-plotting mode.
f3	Jumps between the work-screen and the character-editing screen.
f5	Cycles the border and menu colors.
f6	(Only operates in the work-screen) In color-plotting mode, cycles the cursor color.
f8	(Only operates in the work-screen) Acts as a blink toggle. Blink affects the character under the cursor.

Section 5 SOUND TOOLS

The Programmers' BASIC Toolkit gives you three voices that you can arrange in complex ways to create an array of sounds from three-part harmonies to cannon fire. You can control each voice independently.

Wave Forms and ADSR

Each voice is defined by its wave form. The types of waves are triangle, sawtooth, pluse, and noise.

Each sound is defined by the settings you make for its attack, decay, sustain, and release. These four variables make up the volume envelope. A short attack makes a note start at once. A long attack produces a note that starts slowly and rises gradually in volume. A short decay makes a note end abruptly. A long decay makes the volume fade down. The volume fades to the sustain level until the program encounters a VOICE FREEZE command. Then the volume falls away at the release rate.

You select the settings for each voice when you use the VOICE <voice number> WAVE <wave number> [,<pulse width>] command and the VOICE <voice number> ADSR commands.

Automatic Sound

Tones are to voices as shapes are to sprites. That is, you can define a sequence of tones, then set the sequence up to play automatically, much as you might animate a sprite.

For example, enter:

```
10    VOLUME 15
20    VOICE 1 WAVE 2 ADSR 10,15,10,10
30    VOICE 1 PLAY 9854,14764,16572,9854,6574,13253 SPEED 25
40    VOICE 1 GO
50    SOUND GO
RUN
```

To make the tune repeat ceaselessly, edit line 30 to read:

```
30    VOICE 1 PLAY 9854,14764,16572,9854,6574,13253,←SPEED 25
```

To stop the tune, type **SOUND OFF**.

Section 6 REFERENCE SUMMARY

This section lists all Programmers' BASIC Toolkit commands alphabetically. As you write your programs, you'll probably use this section more than any other, so we've tried to make it as simple as possible for you to get the information you need.

Table 6-1 gives a brief summary of the command names and what the commands do.

Detailed explanations of each command begin right after the table. You'll see that the command name appears at the top of the page. Right below that you'll find the format that you must type the command in. Next, follows a description, usually an example or two, and, finally, an error message (when appropriate).

Table 6-1. Command Summary

Command Name	**Description**
BACKGROUND	Changes background color.
BACKGROUND (<SPRITE Number>.)	Checks for sprite-to-background collision.
BACKUP	Creates a backup copy of Programmers' BASIC Toolkit.
BORDER	Changes border color.
BOX	Draws rectangular boxes.
CHANGE	Replaces text string.
CHAR (<ascii>)	Redefines characters into a RAM character set.
CHAR (<ascii>,n)	Visual representation to redefine characters.
CHAR LOAD	Loads the characters redefined by CHAR (<ascii>), CHAR (<ascii>,n), and Background/ Font Editor.

Table 6-1. Command Summary (continued)

CHAR RAM	Switches to the redefined character set.
CHAR RESET MEMORY	Removes protection from the RAM character set.
CHAR ROM	Displays the standard Commodore character set.
CHAR SAVE	Saves the redefined character set to RAM.
CHAR SET MEMORY	Protects the RAM character set.
CIRCLE	Draws circles, ellipses, arcs, and regular polygons.
CLEAR [<byte>]	Fills HIRES and MULTI screen memory with <byte>.
COLOR	Selects the color for the DOT and LINE commands.
COPY HIRES TO PRINTER	Prints the present image at the printer.
COPY HIRES TO SPRITE	Transfers the image under the sprite.
COPY LOWERCASE TO RAM	Copies character-definition data from ROM to RAM.
COPY MULTI TO PRINTER	Prints the current screen image.
COPY MULTI TO SPRITE	Transfers the image under the sprite.
COPY SPRITE TO HIRES	Displays predefined sprite shapes.
COPY SPRITE TO MULTI	Displays predefined sprite shapes.

Table 6-1. Command Summary (continued)

COPY UPPERCASE TO RAM	Copies character-definition data from ROM to RAM.
COPY TEXT TO HIRES	Copies the current TEXT screen to the HIRES screen.
COPY TEXT TO PRINTER	Prints the current TEXT screen.
CREATE	Creates an auto-booting disk from a Programmers' BASIC Toolkit program.
DIR	Prints out a device's disk directory.
DISK	Shows a disk drive's error status.
DISK "<Command>"	Sends disk drive commands to the disk drive.
DO	Executes a procedure and passes variables.
DOT	Plots a dot or a series of dots.
ELSE	Transfers control to the ELSE statement.
FILL	Fills an enclosed object with the current color.
FIND	Finds a string.
GOTO	Branches to line numbers.
GPRINT	Prints letters and numbers.
HELP	Displays a list of Programmers' BASIC Toolkit Commands with descriptions.
HIRES	Switches to hires mode.
HIRES FROM TO	Splits the screen into two sections.

Table 6-1. Command Summary (continued)

HIRES COLOR	Defines fore- and background colors.
HIRES LOAD	Load a high-resolution image.
HIRES SAVE	Saves a high-resolution image.
JOY	Returns the present position of the joystick.
KEY	Assigns a string to a function key.
KEY LIST	Lists the operations in the current function keys.
KEY LOAD	Loads the current function-key assignments.
KEY OFF	Turns off Programmers' BASIC Toolkit function keys.
KEY ON	Turns on Programmers' BASIC Toolkit function keys.
KEY SAVE	Saves the current function-key assignments.
LINE	Draws lines.
LIST	Lists the program.
LLIST	Lists the program on the printer.
LPRINT	Prints whatever you've selected on the printer.
MULTI	Switches to multicolor modes.
MULTI COLOR	Selects three principal colors.
MULTI FROM TO	Splits the screen into two sections.
MULTI LOAD	Loads a MULTI screen image.
MULTI SAVE	Saves a MULTI screen image.

Table 6-1. Command Summary (continued)

ON ERROR GOTO	When an error occurs, jumps to the selected line number.
ON ERROR OFF	Turns off error trapping.
ON ERROR ON	Turns on error trapping.
PADx	Reads bitpad inputs x, y, and buttons.
PRINT AT	Prints specified data at the selected cursor position.
PROCEDURE	Defines the procedure for a DO command.
REN	Resequences the program's line numbers.
RESET	Returns sound, graphics, and sprites to normal.
RESTORE	Resets the pointer to the next DATA element.
ROLL	Rolls the screen in the selected direction.
SCALE	Changes the scale.
SCROLL	Scrolls the screen in the selected direction.
SETORIGIN	Sets the origin.
SORT	Sorts an array.
SOUND CLEAR	Resets all three voices.
SOUND FREEZE	Stops automatic sound sequences.
SOUND GO	Starts automatic sound sequences.

Table 6-1. Command Summary (continued)

SOUND OFF	Turns the volume to zero.
SOUND ON	Restores the volume.
SPRITE	Checks sprite-to-sprite collision.
SPRITE ANIMATE OFF	Turns off sprite animation.
SPRITE ANIMATE ON	Turns on sprite animation.
SPRITE ANIMATE SPEED	Selects the shapes to slip between.
SPRITE AT	Positions sprites.
SPRITE CLEAR HIT	Clears sprite collision flags.
SPRITE COLOR	Sets a sprite's color.
SPRITE FREEZE	Stops sprite animation or movement.
SPRITE HIRES	Puts sprite in HIRES mode.
SPRITE LOAD	Loads sprite shapes.
SPRITE MOVE	Turns on sprite movement or animation.
SPRITE MULTI	Puts a sprite into MULTI mode.
SPRITE MULTICOLOR	Defines two sprite colors.
SPRITE OFF	Turns the sprites off.
SPRITE ON	Turns the sprites on.
SPRITE ON BACKGROUND	Places a sprite on top of the background.
SPRITE UNDER BACKGROUND	Places a sprite behind the background.
SPRITE SAVE	Saves the sprite shape.
SPRITE SHAPE	Assigns a shape to a sprite.
SPRITE SPEED	Defines the speed of the sprite.

Table 6-1. Command Summary (continued)

SPRITE XYSIZE	Sets a sprite's horizontal and vertical size.
TEXT	Switches to TEXT mode.
TEXT FROM TO	Splits the screen into two sections.
TEXT LOAD	Loads text.
TEXT SAVE	Saves text.
UNNEW	Reverses the action of NEW.
VOICE ADSR	Determines the volume envelope.
VOICE FREEZE	Stops the automatic voice sequence.
VOICE GO	Starts the automatic voice sequence.
VOICE OFF	Turns a voice off.
VOICE ON	Turns a voice on.
VOICE PLAY	Defines a voice's tone sequence.
VOICE TONE	Sets the voice tone.
VOICE WAVE	Selects a voice's waveform.
VOLUME	Master volume control.
WINDOW	Sets up a window.
XPOS	Returns the selected sprite's x-coordinate.
YPOS	Returns the selected sprites y-coordiante.

BACKGROUND
(changes background color)

BACKGROUND <color>

Description

BACKGROUND changes the TEXT screen's background color. You can use the color number of the screen color names (see Appendix C). You can also use this command to select the background color in MULTI mode. The default f2 key prints out the word: **BACKGROUND**. Use this command in either direct or program mode.

Example

```
BACKGROUND GREEN
20 BACKGROUND RED
20 BACKGROUND 1+5*5
```

Error Message

ILLEGAL QUANTITY ERROR

The color number is not in the range 0 to 15.

BACKGROUND
(checks for sprite-to-background collision)

BACKGROUND (<sprite number>)

see also SPRITE CLEAR HIT

Description

BACKGROUND checks for sprite-to-background collision. The function returns true if the sprite has been in contact with any character or graphic.

The sprite collision flag clears after each use of the function, or when you use SPRITE CLEAR HIT.

Sprite-to-background collisions and sprite-to-sprite collisions work in the same manner.

Use the BACKGROUND (<sprite number>) command to see if a sprite has touched or is touching a background.

Use the SPRITE CLEAR HIT command to check whether a sprite is in contact with a background at an instant.

Example

```
10 IF BACKGROUND (2) THEN PRINT "WOW!"
```

Error Message

ILLEGAL QUANTITY ERROR

The sprite number is not in the range 1 to 8.

BACKUP

Description

Creates a backup copy of Programmers' BASIC Toolkit. The backup is for archival purposes only. This command will prompt you through the backup process.

BORDER

(changes border color)

BORDER <color>

Description

BORDER changes the border color. Use with the color numbers or the color names (see Appendix C). You can use this command in either direct or program mode.

Example

```
BORDER PURPLE
    10      BORDER RED
    10      BORDER 8/2+1
```

Error Message

ILLEGAL QUANTITY ERROR

The color number is not in the range 0 to 15.

BOX

(draws rectangular boxes)

BOX <corner x>,<corner y> [XYSIZE <x>,<y>] [TO <corner x>,<corner y>]

Description

BOX draws rectangular boxes on the HIRES and MULTI screens. <corner x>,<corner y> locates the corner where the box will be drawn. If both values are positive, (x,y) is the lower left corner. If both values are negative, the box is drawn down and to the left of the point.

XYSIZE determines the box's size and shape.

TO specifies the box's upper right corner. TO does not specify the length of the sides.

Example

```
    10      HIRES: BOX 10,10 XYSIZE 30,40
    RUN

    10      HIRES
    20      FOR A=0 TO 360 STEP 10
    30      BOX A,A XYSIZE 30,30
    40      NEXT A
    RUN
```

CHANGE

(replaces <old string> with <new string>)

CHANGE "<old string>" TO "<new string>"

Description

CHANGE finds all occurrences of the old string and replaces them with the new string.

Example

```
CHANGE "FLOW" TO "SLOW"
CHANGE /MOVE/ TO /FIND/
CHANGE 'RIGHT$(' TO 'MID$('
```

CHAR (<ascii>)

(redefines characters into a RAM character set)

CHAR <ascii> = <n>,<n>,<n>,<n>,<n>,<n>,<n>,<n>

also see CHAR (<ascii>,n) (visual representation to redefine characters)
CHAR SAVE
CHAR LOAD

Description

<ascii> is the display code (in decimal) of the character that you wish to redefine (see Appendix G for the screen display codes).

<n>...<n> are the values (in decimal) that represent the redefined character.

A character is made up of a square that contains 64 dots (bit positions). In Figure 6-1, a bit position that is on equals a binary 1; a bit position that is off equals a binary 0.

To translate a bit position's binary value to a decimal value, raise 2 to the power of that bit position's location. For example, in Figure 6-1, the next-to-leftmost lower bit has a location value of 6. Two to the sixth power is 64.

To calculate the values for <n>...<n>, look at each row: first, find the value for each bit position that is on, then add those values together. The result is the value of <n> for that row. For example, in the second row (from the top), bit positions 7, 6, and 0 are on. The values for these bit positions are: 128 for bit position 7 (2 to the seventh power), 64 for bit position 6, and 1 for bit position 0. Added together these bit-position values equal 193 (128+64+1). The values that you calculate for each row will fall between 0 and 255.

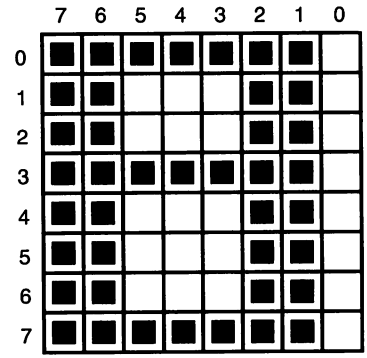


Figure. 6-1. New Character

Example

To redefine B as the character shown in Figure 6-1, write the CHAR command as follows:

```
CHAR(2) = 254,198,198,254,198,198,198,254
```

In a program to replace every occurrence of B with the new character, this command would appear as:

```
10      COPY UPPERCASE TO RAM
20      CHAR RAM
30      CHAR(2) = 254,198,254,198,198,198,254
40      PRINT "B"
RUN
```

CHAR <ascii>,n)

(visual representation to redefine characters)

CHAR <ascii>,n) = "<an 8character string>"

also see CHAR (redefines characters into a RAM character set)
CHAR LOAD
CHAR SAVE

Description

CHAR (<ascii>,n) allows you to use a visual representation of the character you want to redefine. Use the CHAR (<ascii>,n) command eight times--once for each row in the character. (A character is made up of a square that contains 64 bit positions.) Then type any character (except space) to indicate which bits you want to have on. See Figure 6-2.

<ascii> is the code number (in decimal) of the character that you wish to redefine (see Appendix G for the screen display codes).

n is the row number in the visual character.

```
CHAR(2,1)="rrrrrrr "
CHAR(2,2)="  rr rrr"
CHAR(2,3)="rrrr rr "
CHAR(2,4)="rrrrrrrr"
CHAR(2,5)="    rrrr"
CHAR(2,6)="rrrr rrr"
CHAR(2,7)="rrrrrrrr"
CHAR(2,8)="rr    rr"
```

Figure 6-2. Visual Redefinition

Example

The new character replaces the letter B everywhere that B appears:

```
10      COPY UPPERCASE TO
        RAM
20      CHAR RAM
30      CHAR(2,1)  = "rrrrrrr "
40      CHAR(2,2)  = "  rr rrr"
50      CHAR(2,3)  = "rrrr rr "
60      CHAR(2,4)  = "rrrrrrrr"
70      CHAR(2,5)  = "    rrrr"
80      CHAR(2,6)  = "rrrr rrr"
90      CHAR(2,7)  = "rrrrrrrr"
100     CHAR(2,8)  = "rr    rr"
RUN
```

CHAR LOAD

(loads the characters redefined by CHAR (<ascii>) and CHAR (<ascii>,n) or created through the background/font editor)

CHAR LOAD"<file name>"[,<device>]

also see CHAR RAM
CHAR ROM
CHAR (<ascii>)
CHAR (<ascii>,n)
CHAR SAVE
COPY UPPERCASE TO RAM
COPY LOWERCASE TO RAM

Description

CHAR LOAD loads the entire character set, even when you only redefined one character. It doesn't matter if you redefined the character in a program or input it directly. The device default is 8.

To load the character set, type

```
COPY UPPERCASE TO RAM
CHAR RAM
CHAR LOAD"<file name>" [,<device>]
```

After you've loaded it, use CHAR RAM to switch into the character set. To return to the standard Commodore character set, use CHAR ROM.

NOTE

The data for the RAM character set uses memory inside the BASIC workspace. If your BASIC program is too large, it may be destroyed when you use CHAR LOAD.

CHAR RAM

(switches to the redefined character set)

CHAR RAM

also see CHAR (<ascii>)
CHAR (<ascii>,n)
CHAR LOAD
CHAR ROM
CHAR SAVE
COPY UPPERCASE TO RAM
COPY LOWERCASE TO RAM
CHAR SET MEMORY
CHAR RESET MEMORY

Description

CHAR RAM switches to the character set redefined under the CHAR (<ascii>) or CHAR (<ascii>,n) commands. These characters reside in locations $3800 through $4000. If you haven't loaded any characters to these locations, you'll get garbage when you use CHAR RAM. (Use CHAR LOAD to load the character set.)

CHAR ROM returns you to the standard Commodore character set, and does not destroy the character set in RAM.

COPY UPPERCASE TO RAM clears the character set from RAM.

CHAR RESET MEMORY

(removes protection from the RAM character set)

CHAR RESET MEMORY

also see CHAR SET MEMORY
RESET

Description

CHAR SET MEMORY protects a RAM character set from damage by BASIC variables or program storage. CHAR RESET MEMORY removes that protection.

CHAR ROM

(displays the standard Commodore character set)

CHAR ROM

also see CHAR LOAD
CHAR SAVE
CHAR RAM

Description

Use CHAR ROM to return to the standand Commodore character set after you've used a character set from CHAR RAM. CHAR ROM does not destroy the character set in RAM.

CHAR SAVE

(saves the redefined character set to RAM)

CHAR SAVE "<file name>" [,device]

also see CHAR LOAD
CHAR RAM
CHAR ROM
CHAR (<ascii>)
CHAR (<ascii>,n)
COPY UPPERCASE TO RAM
COPY LOWERCASE TO RAM

Description

CHAR SAVE saves your redefined character to RAM. It doesn't matter whether you created the character in a program or input it directly. The device default is 8.

Example

```
10 CHAR SAVE "GARDEN",9
10 CHAR SAVE "CACTUS"
```

CHAR SET MEMORY

(protects the RAM character set)

CHAR SET MEMORY

also see CHAR RESET MEMORY
RESET

Description

CHAR SET MEMORY reserves the memory you'll need for redefined characters. The memory area is protected from BASIC actions, variable storage, and program storage. Use CHAR RESET MEMORY or RESET to remove the protection.

CIRCLE

(draws circles, ellipses, arcs, regular polygons)

CIRCLE <center x>,<center y> XYSIZE <x size>,<y size> [FROM <starting angle>] [TO <ending angle>] [STEP <angle>]

Description

CIRCLE draws circles, ellipses, arcs, and regular polygons.

<center x>,<center y> locates the circle's center.

XYSIZE determines the circle's size and shape.

<size x>,<size y> draws larger circles and makes arcs, ellipses, and elliptical arcs.

To draw a full circle, omit FROM...TO. To draw an arc, include FROM...TO. All angles are in degrees and may take any value. Angle values for the FROM...TO command are shown in Figure 6-3.

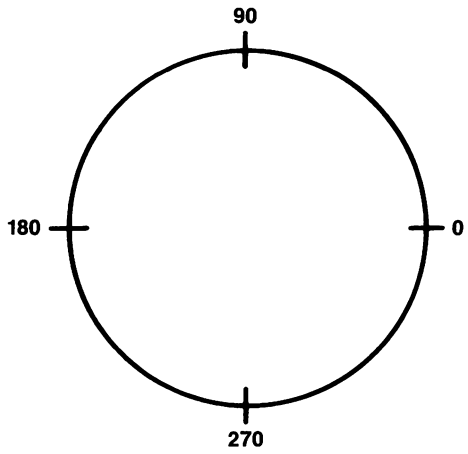


Figure 6-3. Angle Values in CIRCLE

<angle> makes large polygons. All angles are in degrees and may take any value.

Example

```
10      HIRES
20      CIRCLE 50,60 XYSIZE 10,30
RUN

20      CIRCLE 100,100 XYSIZE 28,20 STEP 360/5
RUN

20      CIRCLE 100,120 XYSIZE 50,40 FROM 10 TO 180
RUN

10      CLEAR:BORDER BLACK
20      X=5:Y=5:N=60
30      HIRES
40      HIRES COLOR GREEN ON BLACK
50      CIRCLE160,100XYSIZEX,YSTEPN
60      X=X+5:Y=Y+5:N=N+1
70      GOTO50
RUN

10      CLEAR:BORDER BLACK
20      X=160:Y=100:N=60
30      HIRES
40      HIRES COLOR GREEN ON BLACK
50      CIRCLE160,100XYSIZEX,YSTEPN
60      X=X-5:Y=Y-5:N=N+1
70      GOTO50
RUN
```

CLEAR

(fills HIRES and MULTI screen memory with <BYTE>)

CLEAR [<BYTE>]

CLEAR <number>

Description

CLEAR fills the HIRES and MULTI screen memory with <byte>. Use CLEAR either before or after the HIRES COLOR <foregroundcolor> ON <background-color> or MULTI COLOR <color1>,<color2>,< color3> commands. The screen becomes the designated HIRES color, however, only after you issue both commands.

CLEAR or CLEAR 0	The screen becomes the background color.
CLEAR 255	The screen becomes the foreground color in HIRES mode, <color 3> in MULTI mode with band of <color 1> at the bottom.
CLEAR 85	The screen becomes <color 1> in MULTI mode.
CLEAR 170	The screen becomes <color 2> in MULTI mode.

Example 1

To turn the MULTI screen purple:

```
10      MULTI:BACKGROUND PURPLE
20      MULTICOLOR RED,GREEN,BLUE
30      CLEAR
RUN
```

Example 2

To turn the HIRES screen blue:

```
10      HIRES
20      HIRES COLOR BLUE ON RED
30      CLEAR 255
RUN
```

Error Message

ILLEGAL QUANTITY ERROR

The number after the CLEAR command is not in the range 0 to 255.

COLOR

(selects the color for the DOT and LINE commands)

COLOR <color>

COLOR HIRES <color>
COLOR MULTI <color>

Description

COLOR selects the color to be used with the DOT and LINE commands. You can use the color number or the color name (see Appendix C).

In HIRES mode, use the COLOR HIRES <color> command.

In MULTI mode, use the COLOR MULTI <color> command.

If you omit HIRES or MULTI after the word COLOR, Programmers' BASIC Toolkit assumes you want to use whatever mode the computer is currently in.

NOTE

If you use a <color> that is not a principal color in the HIRES COLOR <foregroundcolor> ON <backgroundcolor> or in the MULTI COLOR <color 1>,<color 2>,<color 3> command, the next line plotted appears in the selected color. However, where two plotted lines cross, their colors will bleed into one another.

Example 1

The HIRES screen turns black and the box plotted switches from pink to blue. Press the f1 key several times to view the effects.

```
RESET
10      HIRES
20      HIRES COLOR BLUE ON BLACK
30      FOR D = 1 TO 300: NEXT D
40      COLOR HIRES PINK
50      BOX 100,100 XYSIZE 30,40
```

Example 2

The MULTI screen turns black and the next line plotted turns green.

```
10      MULTI: BACKGROUND 0: CLEAR
20      MULTICOLOR RED,BLUE,GREEN
30      COLOR GREEN
40      LINE TO 100,100
RUN
```

Error Message

ILLEGAL QUANTITY ERROR

<color> is not in the range 0 to 15.

COPY HIRES TO PRINTER

(prints the present image at the printer)

COPY HIRES TO PRINTER

Description

COPY HIRES TO PRINTER prints the present graphics (that are within the WINDOW) to the 1525 Graphics Printer (or compatible), or the Gemini series printer.

To stop printing, press the RUN STOP key.

NOTE

For Gemini printers, before you use the COPY HIRES TO PRINTER command, type the following:

SYS 32512
or
SYS &H7F00
and press RETURN.

COPY HIRES TO SPRITE

(transfers the image under the sprite)

COPY HIRES TO SPRITE <sprite number>

also see COPY SPRITE <sprite number> TO HIRES

Description

COPY HIRES TO SPRITE transfers the image under the sprite and puts it into the sprite shape data. Use this command to "pick up" objects on the screen and move them around. To transfer images from one part of the graphics screen to another, use this command in conjunction with the COPY SPRITE <sprite number> TO HIRES command.

Example

This program allows you to pick up letters (from a character set) and put them into a sprite.

```
10    CHAR LOAD "ALTERNATE"
20    CHAR RAM
30    HIRES TO 15
40    CLEAR
50    GPRINT AT 1,1 XYSIZE 3,2, "THIS IS ANOTHER CHARACTER SET"
60    SPRITE 1 ON AT 25,200
70    COPY HIRES TO SPRITE 1
80    SPRITE 1 SPEED 1,0: SPRITE MOVE
```

While the sprite is visible on the screen, type **COPY SPRITE 1 TO HIRES** and press RETURN.

COPY LOWERCASE TO RAM

(copies character-definition data from ROM to RAM)

COPY LOWERCASE TO RAM

also see CHAR SET MEMORY
CHAR RESET MEMORY
CHAR RAM
CHAR ROM
CHAR (<ascii>)
CHAR (<ascii>,n)

Description

COPY LOWERCASE TO RAM copies character-definition data from ROM into the reserved 2K RAM character set. After using this command, you can redefine characters to create custom charater sets. Whatever character set is currently in RAM is replaced.

This command reconfigures the BASIC workspace and erases all variables. (See Appendix D for the memory map.)

Example

```
COPY LOWERCASE TO RAM
CHAR RAM
CHAR(1) = 193,255,60,66,153,129,60,193
```

COPY MULTI TO PRINTER

(prints the current screen image)

COPY MULTI TO PRINTER

Description

COPY MULTI TO PRINTER prints the current image (within the WINDOW) on a 1525 Graphics Printer (or compatible), or a Gemini series printer.

To stop printing, press the RUN STOP key.

NOTE

For Gemini printers, before you use this command, type the following number: **SYS 32512** and press RETURN.

COPY MULTI TO SPRITE

(transfers the image under the sprite)

COPY MULTI TO SPRITE <sprite number>

also see COPY SPRITE <sprite number> TO MULTI

Description

COPY MULTI TO SPRITE transfers the image under the sprite and puts it into the sprite shape data. Use this command to "pick up" objects on the screen and move them around.

Example

This program creates a sprite and copies a multicolor image onto it. The COPY SPRITE 1 TO MULTI command copies the sprite back onto the screen. Press the f1 key to run the program again.

```
NEW
10      BACKGROUND 1: CLEAR
20      MULTI: MULTICOLOR PURPLE,GREEN,BLUE
30      FOR T = 1 TO 25: COLOR 3*RND(8) + 4
40      LINE TO 320*RND(8),150*RND(8): NEXT T
50      SPRITE 1 ON AT 150,100 MULTI
60      COPY MULTI TO SPRITE 1
70      FOR T = 200 TO 100 STEP-2
80      SPRITE 1 AT 150,T
90      COPY SPRITE 1 TO MULTI: NEXT T
RUN
```

COPY SPRITE TO HIRES

(displays predefined sprite shapes)

COPY SPRITE <sprite number> to HIRES

also see COPY HIRES TO SPRITE

Description

COPY SPRITE TO HIRES displays predefined sprite shapes. To duplicate a sprite in different places on the screen, move it and then use the COPY SPRITE command. You can also use this command while the sprites are automatically animating or moving.

To erase the sprites, turn them off, then use the COPY SPRITE command.

COPY SPRITE TO MULTI

(display predefined sprite shapes)

COPY SPRITE <sprite number> to MULTI

also see COPY MULTI TO SPRITE
SPRITE MULTICOLOR
MULTI COLOR

Description

COPY SPRITE TO MULTI displays predefined sprite shapes. To duplicate a sprite in different places on the screen, move it and then use the COPY SPRITE command. You can also use this command while the sprites are automatically animating or moving.

When you copy multicolor sprites to the MULTI screen, set the screen multicolors as follows: select SPRITE MULTICOLOR <sprite color1>,<sprite color2>, then set MULTICOLOR <any color>,<sprite color1>,<sprite color 2>.

To erase the sprites, turn them off, then use the COPY SPRITE command.

COPY TEXT TO HIRES

(copies the current TEXT screen to the HIRES screen)

COPY TEXT TO HIRES

Description

COPY TEXT TO HIRES copies the current characters on the TEXT screen to the HIRES screen. All images on the graphics screen are written over.

COPY TEXT TO PRINTER

(prints the current TEXT screen)

COPY TEXT TO PRINTER

Description

COPY TEXT TO PRINTER prints the current characters on the TEXT screen to a compatible printer (device 4) in 40-column format.

To use an 80-column format, use the following commands:

```
OPEN 4,4: CMD4: LIST
```

To print graphics characters, use a Commodore 1525 printer.

Example

```
10      HIRES
20      LINE 0,0 TO 120,120
COPY TEXT TO PRINTER
```

COPY UPPERCASE TO RAM

(copies character-definition data from ROM to RAM)

COPY UPPERCASE TO RAM

also see CHAR SET MEMORY
CHAR RESET MEMORY
CHAR RAM
CHAR ROM
CHAR (<ascii>)
CHAR (<ascii>,n)

Description

COPY UPPERCASE TO RAM copies character definition data from ROM into the reserved 2K RAM character set. You can then redefine the character definition data to create custom characters. This command reconfigures the BASIC workspace and erases all the variables. (See Appendix D for the memory map.)

This command replaces whatever character set currently resides in RAM.

When you are redefining characters, use this command before you use CHAR RAM.

Example

```
COPY UPPERCASE TO RAM
CHAR RAM
CHAR(2) = 255,193,255,153,60,255,66,1
```

CREATE

(creates an auto-booting disk)

CREATE

Description

Use CREATE when you want to place a program you've written onto a disk that will boot automatically.

To create an auto-booting disk:

1. Load your program into memory.
2. Insert a blank disk in the drive.
3. Type **CREATE**

When you want to run your newly created disk, turn off your computer, insert the disk into your drive and turn your computer on. The disk will automatically boot and run your Toolkit BASIC program.

DIR

(prints out a device's disk directory)

DIR [<device number>]

Description

DIR displays a given device's disk directory. Device 8 is the default. DIR does not affect the program in memory.

To stop the scrolling, press RUN STOP. Once stopped, the scrolling can't be started again.

To slow the scrolling, hold down the CTRL key.

Example

DIR9

DISK

(shows a disk drive's error status)

DISK [,<device number>]

Description

DISK displays the error status for disk drive <device number>. The default device number is 8.

Example

```
DISK
DISK,9
```

DISK Command

(sends disk drive commands to the disk drive)

DISK "<command string>"[,<device number>]

Description

DISK command sends standard disk drive commands to disk drive <device number>. The default is 8.

The disk drive commands are:

CLOSE	close
COPY	copy
I	initialize
N0	new
OPEN	open
R0	rename
S0	scratch

See your disk drive manual for details.

Examples

```
DISK "R0:BOB"
DISK "S0:HOUSTON",9
```

DO

(executes a procedure and passes variables)

DO <procedure name> [<variable1>,...,<variablen>]
also see PROCEDURE

Description

DO executes a procedure and passes given variables as parameters. Excess variables are ignored. When more parameters exist than variables, the remaining parameters are set to zero.

Example

```
DO CASTLE(6)
```

DOT

(plots a dot or a series of dots)

DOT <x>,<y>
DOT <x>,<y>;<x>,<y>;<x>,<y>...

Description

DOT plots a dot or a series of dots at location (x,y). The dot appears in the color specified in the COLOR command. The (x,y) coordinates depend on the current ORIGIN setting and SCALE. Dots outside the specified WINDOW are not plotted.

NOTE

Multicolor dots are twice as wide as high-resolution dots, but the coordinate systems are the same.

Example 1

The MULTI screen turns black and a red dot appears at 150,120.

```
10      MULTI: BACKGROUND BLACK
20      MULTICOLOR RED,WHITE,PURPLE
30      CLEAR: COLOR MULTI RED
40      DOT 150,120
RUN
```

Example 2

This program plots a blue sine wave. To stop the program, press RUN STOP, then f7.

```
10      HIRES
20      HIRES COLOR BLUE ON WHITE
30      CLEAR
40      FOR I = 1 TO 32 STEP.10: X(1) = I* 10:Y(1) = ABS(100*SIN(I))
50      COLOR BLUE: DOT X(1),Y(1)
60      NEXT I
RUN
```

ELSE

(transfers control to the ELSE statement)

ELSE <statement>
ELSE <line number>

Description

ELSE used with the IF...THEN command, when the IF...THEN is false, control transfers to ELSE. Use a colon before ELSE.

Example

```
10      IF B = 9, THEN PRINT "YOW!": ELSE PRINT "NOPE"
```

FILL

(fills an enclosed object with the current color)

FILL <x>,<y>

Description

FILL fills an enclosed object with the current color on the HIRES screen. Point (x,y) must reside within the object; otherwise, FILL colors the area surrounding the object.

Example

```
10      HIRES: COLOR RED
20      BOX 100,100 XYSIZE 50,50
30      FILL 130,130
RUN
```

FIND

(finds text strings)

FIND "<search string>"

Description

FIND displays all the lines in your BASIC Toolkit program that contain <search string>. You can use any character as the delimiter: FIND accepts the first character you type in the search string as the delimiter.

FIND isn't limited to whole strings: FIND can find part of a string.

Examples

```
FIND /PRINT/
FIND "GOTO"
```

If you need to find the occurrences of PRINT "FREDERICK", PRINT "FRED", PRINT "FREDDY", and so on, you can write the FIND command as

```
FIND /PRINT "F/
```

or even

```
FIND /INT "F/
```

and FIND will find all of the above.

GOTO

(branches to line numbers)

GOTO <line number>

Description

GOTO branches to a specified <line number> or to a <line number> created by a variable.

Example

```
GOTO Z
GOTO D*2
```

GPRINT

(prints letters and numbers)

GPRINT
GPRINT <output data>
GPRINT AT <x>,<y>,<output data>
GPRINT AT <x>,<y> XYSIZE <x>,<y>,<output data>

Description

GPRINT prints letters and numbers on HIRES and MULTI screens. GPRINT recognizes only alphanumeric and graphic characters, color codes, reverse on/off codes, and return.

Use (x,y) positioning to move the HIRES cursor: GPRINT does not recognize regular cursor movement.

GPRINT can expand printed characters horizontally and vertically.

NOTE

If your characters go beyond the right margin, GPRINT does not automatically wrap them. However, if your characters go beyond the bottom margin, GPRINT wraps the cursor to the first line on the next screen.

Example

```
10      HIRES TO 20:CLEAR
20      FOR I = 1 TO 13
30      GPRINT AT I,1 "TOOLKIT BASIC"
40      NEXT I
RUN
```

Error Message

ILLEGAL QUANTITY ERROR

Either the (x,y) position given for the cursor does not lie on the screen, or the (x,y) expansion factors are negative or too large.

HELP

(prints a command summary)

HELP

Description

Use HELP to see a list of Toolkit BASIC commands with descriptions. You can type HELP whenever you have the READY prompt.

HIRES

(switches to high-resolution graphics)

HIRES

Description

HIRES switches your screen into high-resolution graphics from either the TEXT or the MULTI screens. To return to TEXT or MULTI, type TEXT or MULTI.

HIRES FROM TO

(splits the screen into two sections)

HIRES [FROM <firstline>] [TO <lastline>]
HIRES TO <line>
HIRES FROM <line>

Description

HIRES FROM TO splits the screen into two sections: one for text and one for high-resolution graphics. High-resolution graphics appear from <firstline> to <lastline>. The text appears on the remaining screen. The range for <firstline> to <lastline> is 1 to 25.

Example

```
10    HIRES FROM 10 to 20
10    HIRES FROM 20
10    HIRES FROM 1 TO 2*9
10    HIRES TO 2
```

ERROR MESSAGE

ILLEGAL QUANTITY ERROR

Either number in the HIRES command is not in the range 1 to 25.

HIRES COLOR

(defines fore- and background colors)

HIRES COLOR <foregroundcolor> ON <backgroundcolor>

Description

HIRES COLOR defines the principal fore- and background colors. You may use the color numbers or names (see Appendix C).

Every character-block shares the same two colors. If you use another color, lines that cross each other may bleed. You can limit yourself to two colors, or make sure that drawings in different colors don't come into contact with each other.

Example

This example draws a red line on a black background.

```
10      HIRES COLOR RED ON BLACK
20      COLOR RED: LINE 0,0 TO 100,200
HIRES
RUN
```

ERROR MESSAGE

ILLEGAL QUANTITY ERROR

Any of the color numbers and not in the range 0 to 15.

HIRES LOAD

(retrieves a high-resolution image)

HIRES LOAD "<file name>" [,<device number>]

Description

HIRES LOAD retrieves a high-resolution image from the given device. The default is 8.

Example

```
HIRES LOAD "AUTHOR"
HIRES LOAD "FUN YET",9
HIRES LOAD A$
```

HIRES SAVE

(saves a high-resolution image)

HIRES SAVE "<file name>" [,<device number>]

Description

HIRES SAVE saves a high-resolution image to the selected device. The default is 8.

Example

```
HIRES SAVE "CHAUCER"
HIRES SAVE "ELIOT",9
```

JOY

(returns the present position of the joystick)

JOY <joystick number>

Description

JOY returns a value that corresponds to the present joystick position.

These values are:

± 1	down and left
± 2	down
± 3	down and right
± 4	left
± 5	neutral
± 6	right
± 7	up and left
± 8	up
± 9	up and right

<joystick number> must be 1 or 2.

If the value returned is negative, the joystick button is pressed.

Example

```
10      N=JOY(1)
20      IF N<O THEN PRINT"ZAP!":N=-N
30      PRINT"X,Y DIRECTION",
40      Y=INT((N-4)/3):X=N-Y*3=-5
50      PRINT X;Y
60      GOTO 10
RUN
```

Error Message

ILLEGAL QUANTITY ERROR

The joystick number is not 1 or 2.

KEY

(assigns a string to a function key)

KEY (<key number>) = "<string>"

also see KEY ON
KEY OFF
KEY LOAD
KEY SAVE

Description

KEY assigns a specified string to a specified function key. Use this command to refine the function keys. However, you cannot assign more than 32 characters to one function key.

If you add +CHR$(13) to an operation, you don't need to press carriage return after pressing the function key.

Example

```
KEY(A)=STR$(A)
KEY(J+1)="AM I PROGRAMMING YET?"
```

KEY LIST

(lists the operations in the current-function keys)

KEY LIST

also see KEY
KEY SAVE
KEY LOAD

Description

You can change the operation that a function key performs. KEY LIST lists what operations are currently assigned to what function keys. To see the list, type KEY LIST and press RETURN.

KEY LOAD

(loads the current-function key assignments)

KEY LOAD "<file name>" [,<device>]

also see KEY
KEY ON
KEY OFF

Description

KEY LOAD loads the present function-key assignments from the <device>. The default is 8.

Example

```
KEY LOAD "NEW KEYS"
KEY LOAD "GOLD KEYS",9
```

KEY OFF
(turns off Toolkit BASIC function keys)

KEY OFF

Description

KEY OFF turns off Toolkit BASIC function keys: the function keys operate as they do when Toolkit BASIC is not running.

KEY ON
(turns on Toolkit BASIC function keys)

KEY ON

Description

KEY ON turns on the Toolkit BASIC function key. The current assignments to the function keys are used when the function keys are pressed.

KEY SAVE
(saves the current-function key assignments)

KEY SAVE "<file name>" [,<device>]

Description

KEY SAVE saves the current function-key assignments to <device>. The default is 8.

Examples

```
KEY SAVE "NEW KEYS"
KEY SAVE "GOLD KEYS",8
```

LINE
(draws lines)

LINE [<x>,<y>] [TO <x>,<y>] [TO <x>,<y>]...
LINE <x>,<y>
LINE <x>,<y> TO <x>,<y>
LINE TO <x>,<y> TO <x>,<y>...

Description

LINE draws a line from one point to another in the color specified through the COLOR command. You can use the DOT command as the starting point for the LINE command.

Examples

To draw a line from (x1,y1) to (x2,y2)

```
(LINE x1,y1 TO x2,y2)
```

To draw a line from the last plotted point to (x,y)

```
(LINE TO x,y)
```

To draw a line to connect several points (x1,y1) to (x2,y2) to (x3,y3)

```
(LINE x1,y1 TO x2,y2 TO x3,y3)
```

To set the starting point for the next LINE TO command to (x,y)

```
(LINE x,y)
```

no line appears

```
10      HIRES
20      HIRES COLOR RED ON BLACK
30      LINE 50,40 TO 100,150
RUN
```

LIST

(lists the program to the screen)

LIST

Description

LIST lists the program in Toolkit BASIC, just as it does in BASIC version 2.

You can use LIST in program or direct mode. In direct mode, to slow the scrolling, hold down the CTRL key.

Example

```
10      PRINT: PRINT "I WROTE THIS"
20      LIST
30      PRINT: PRINT "AND I AM FINISHED NOW"
40      END
RUN
```

LLIST

(lists the program or directory to the printer)

LLIST <line number>-<line number>
LLIST DIR
LLIST <"filename">,<device number>

Description

LLIST lists the program in the same way as LIST, except that the output goes to the printer instead of to the screen.

LLIST <n–n> lists the program from <line number> to <line number> to the printer.

LLIST DIR lists the disk directory to the printer.

LLIST <"filename">,<device number> lists sequential files from the selected device number to the printer.

Example

```
10    PRINT: PRINT "DID I WRITE THIS"
20    LLIST
30    PRINT: PRINT "OH, YEAH. NOW I REMEMBER."
40    END
RUN
```

LPRINT

(print to printer)

LPRINT

Description

Use LPRINT when you want to print something at the printer. LPRINT works in the same manner as PRINT, except that the output goes to the printer instead of to the screen.

MULTI

(switches to multicolor mode)

MULTI

also see HIRES
TEXT

Description

MULTI switches the screen to multicolor mode. To switch HIRES mode, type HIRES. To switch to text mode, type TEXT.

MULTICOLOR

(selects three principal colors)

MULTICOLOR <color 1>,<color 2>,<color 3>

Description

MULTI COLOR selects the three principal colors for the MULTI screen. You may use the color names or numbers (see Appendix C).

The background color for the MULTI screen and the TEXT screen is always the same. Select the background color through BACKGROUND.

Usually, MULTI mode has four colors: three colors selected through the MULTI COLOR command and the backgound color. If you use more colors, lines that cross may bleed.

NOTE

In a program, if you select a color other than the three already selected plus the background, the new color replaces <color 1>.

Example

```
10      MULTICOLOR RED,GREEN, WHITE
20      BACKGROUND PURPLE: COLOR BLACK
30      LINE 100,100 TO 200,200
MULTI
RUN
```

Error Message

ILLEGAL QUANTITY ERROR

The selected color numbers are not in the range 0 to 15.

MULTI FROM TO

(splits the screen into two sections)

MULTI [FROM <firstline>] [TO <lastline>]
MULTI TO <line>
MULTI FROM <line>

Description

MULTI FROM TO splits the screen from <firstline> to <lastline>. A TEXT screen appears on the remaining screen. <firstline> and <lastline> must be in the range 1 to 25.

Example

```
MULTI FROM 13 TO 22
```

Error Message

ILLEGAL QUANTITY ERROR

One of the line numbers is not in the range 1 to 25.

MULTI LOAD

(retrieves the MULTI Screen image)

MULTI LOAD "<file name>"[,<device>]

Description

MULTI LOAD loads the MULTI screen image from <device>. The default device is 8.

Example

```
MULTI LOAD "HOUSE"
MULTI LOAD "GARDEN",9
```

MULTI SAVE

(saves a MULTI screen image)

MULTI SAVE "<file name>"[,<device>]

Description

MULTI SAVE saves a multicolor image to <device>. The default device is 8.

Example

```
MULTI SAVE A$
MULTI SAVE "HOUSE"
MULTI SAVE "GARDEN",9
```

ON ERROR GOTO

(when an error occurs, jumps to the selected line number)

ON ERROR GOTO <line number>

also see ON ERROR OFF
ON ERROR ON

Description

When an execution occurs, the program jumps to <line number>. The variable ER contains the error number, and the variable LI contains the line number where the error occurred. (See Appendix G for error numbers.)

If an error occurs before the program encounters ON ERROR GOTO, but ON ERROR ON was executed, an undefined statement error occurs.

When you use the ON ERROR GOTO command, control jumps to <line number>, and error trapping automatically turns off. Then, to trap more errors, you must use the ON ERROR ON command again. In this way, an error-handling routine can't call on itself.

An error in a FOR...NEXT loop terminates the loop; the loop can't be continued.

Example

```
10    ON ERROR GOTO 80
20    J=1
30    FOR I=1 TO 999999
40    J=J*I*I*I
50    PRINT J
60    NEXT I
70    END
80    PRINT "NUMBERS TOO LARGE"
RUN
```

ON ERROR OFF

(turns off the error trapping)

ON ERROR OFF

also see ON ERROR GOTO
ON ERROR ON

Description

ON ERROR OFF disables the error trapping performed by the ON ERROR GOTO command. Error handling reverts to the normal Toolkit BASIC method.

ON ERROR ON

(turns on error trapping)

ON ERROR ON

also see ON ERROR GOTO
ON ERROR OFF

Description

ON ERROR ON turns on error trapping by the ON ERROR GOTO command and causes the program to branch to the line number specified in ON ERROR GOTO <line number>. No error message appears.

Example

```
10    DIM A(10)
20    ON ERROR GOTO 100
30    ON ERROR ON
40    FOR I = 1 TO 20
50    A(I) = I * 100
60    PRINT A(I)
70    NEXT I
80    END
100   PRINT "ERROR IN LINE";LI
110   PRINT "VARIABLE NOT DIMENSIONED HIGH ENOUGH"
120   END
RUN
```

PAD

(reads bitpad inputs)

PADX(<port number>)
PADY(<port number>)
PADB(<port number>)

Description

PADX, PADY, and PADB read inputs from the bitpad. PADX reads the x-coordinates and PADY reads the y-coordinates. PADB reads the buttons:

0 = neither button
1 = left button
2 = right button
3 = both buttons

<port number> is the port the bitpad is plugged into — usually port 2.

PRINT AT

(prints specified data at the selected cursor position)

PRINT AT <x>,<y>[,][<data>]

also see GPRINT

Description

On the text screen, PRINT AT prints the specified data at the selected cursor position. Note that <x> and <y> are referenced from the screen's upper left corner (1,1).

Example

```
PRINT AT A,B,C
PRINT AT 10,5 A$
PRINT AT X*3,Y+5"ARE WE HAVING FUN YET?"
```

PROCEDURE

(defines the procedure for a DO command)

PROCEDURE <procedure name> [<variable 1>,...,<variable n>]
.
.
.
RETURN

also see DO

Description

The word PROCEDURE begins the definition of a procedure that the DO command will execute.

A procedure is a subroutine: it has a name and can pass up to 10 variables. Use of procedures can reduce use of excess GOTOs or GOSUBs. The DO command executes the procedure. RETURN terminates a procedure definition.

Don't use spaces in the procedure name.

Don't use integer or string variables: all variables must be standard floating point variables.

Passing an array can create recursive procedures.

Example

```
10    PROCEDURE POLYGON (N)
20    CLEAR
30    CIRCLE 100,100 XYSIZE 28,20 STEP 360/N
40    RETURN
HIRES TO 15
DO POLYGON(5)
DO POLYGON(8)
```

REN

(resequences the program's line numbers)

REN [<increment>],[<starting line number>]

Description

REN resequences a program's line numbers and modifies all line references. <increment> and <starting line number> default to 10.

When REN encounters any computed line number, however, you receive the warning COMPUTED ADDRESS <line number>. Check that the line numbers still address the correct lines after you use REN.

Example

```
REN    5,100
REN    1
REN
```

RESET

(returns sound, graphics, and sprites to normal)

RESET

Description

RESET returns sound, graphics, and sprites to normal. Use it at the end or beginning of programs.

RESET sets the following conditions:

- The screen goes to TEXT mode.
- All eight sprites turn off.
- Sprite animation and movement freeze.
- Animation sequences clear.
- The ORIGIN returns to the screen's lower left corner.
- The WINDOW returns to the screen size.
- All sprite XYSIZEs set to 1,1.
- All sprites go to HIRES mode.
- Automatic sound freezes.
- All three voices turn off.
- Automatic sound sequences clear.
- HIRES and MULTI screens clear.
- BACKGROUND and BORDER colors reset.
- The TEXT screen clears.
- ON ERROR OFF becomes active.
- Scrolling WINDOW resets.
- Memory resets.
- Variables clear.
- ROM character set returns.

Example

```
RESET
10      RESET
```

RESTORE

(resets the pointer to the next DATA element)

RESTORE [<line number>]

Description

RESTORE sets the pointer to the next DATA element to the beginning of the specified <line number>.

Example

```
A=6
RESTORE A
RESTORE 100
```

ROLL

(rolls the screen in the selected direction)

ROLL <direction><number of characters>[WINDOW <min x>,<min y>,<max x>,<max y>]

also see SCROLL

Description

ROLL rolls the selected part of the screen in the specified direction. Directions are:

Right
Left
Up
Down

(You need only enter the first direction character, for example L for LEFT.)

If you omit WINDOW, the program uses the previous WINDOW setting. If no window is set, the WINDOW defaults to the whole screen.

Example

```
ROLL DOWN 5
ROLL DOWN 5 WINDOW 1,1,15,15
ROLL UP A*2
```

SCALE

(changes the scale)

SCALE [<x range>,<y range>]

also see SETORIGIN

Description

SCALE changes the scale for both the HIRES and MULTI screens. The limits are 1 to 10,000 vertically or horizontally.

The coordinates that define the origin are scaled according to what scale you use in this command. Use the SETORIGIN command to move the origin.

Example

The commands **SCALE 20,4** and **SETORIGIN 10,2** create the following (Figure 6-4):

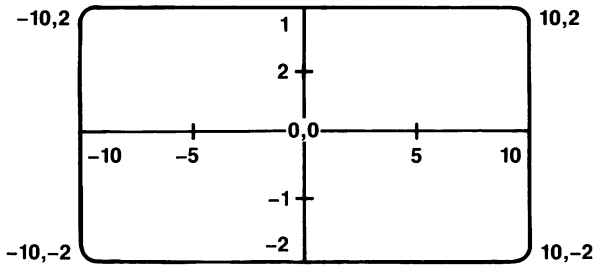


Figure 6-4. SCALE and SETORIGIN

SCROLL

(scrolls the screen in the selected direction)

SCROLL <direction><number of characters>[WINDOW <min x>,<min y>,<max x>,<max y>]

also see ROLL

Description

SCROLL scrolls the selected part of the screen in the specified direction. Directions are:

Left
Right
Up
Down

(You need only type in the direction's first letter, for example U for Up.)

If you omit WINDOW, the program uses the previous WINDOW setting.

If no window is set, WINDOW defaults to the whole screen.

Example

```
SCROLL UP 6
SCROLL DOWN 5 WINDOW 1,1,15,15
SCROLL DOWN A*5
```

SETORIGIN

(sets the origin)

SETORIGIN [<origin x>,<origin y>]
SETORIGIN

also see SCALE

Description

SETORIGIN sets the origin anywhere, even off the screen. The origin is the 0,0 reference point for an x,y coordinate system. All lines, dots, window settings, and sprites are set with respect to the origin. The default for the origin is the screen's lower left corner.

Note that SCALE also affects the coordinates of the origin.

Example

```
10      HIRES
20      HIRES COLOR RED ON BLACK
30      SETORIGIN 20,35
40      SCALE 20,4
50      LINE -10,-5 TO 40,55
RUN
```

SORT

(sorts an array)

SORT A$[,<direction>]
SORT A%[,<direction>]
SORT A[,<direction>]

Description

Use SORT to sort an array in ascending or descending order. <direction> is D for descending and A for ascending. The default is ascending.

Example

```
SORTA$,D
```

SOUND CLEAR

(resets all three voices)

SOUND CLEAR

Description

For all three voices, SOUND CLEAR resets VOLUME, ADSR, TONE, and WAVE values. To be sure that the sound generator is reset, use SOUND CLEAR at the beginning of a program.

Example

```
20      SOUND CLEAR
```

SOUND FREEZE

(stops automatic sound sequences)

SOUND FREEZE

also see SOUND GO

Description

SOUND FREEZE stops all automatic sound sequences until the program encounters SOUND GO. Use this command to help sychronize voices.

SOUND GO

(starts automatic sound sequences)

SOUND GO

also see SOUND FREEZE

Description

SOUND GO starts automatic sound sequences: the VOICE <voice number> PLAY command takes effect no matter what other voices are doing.

Use this command to synchronize several voices.

SOUND OFF

(turns the volume to zero)

SOUND OFF

also see SOUND ON

Description

SOUND OFF turns the master volume to zero.

Example

```
10    SOUND OFF
10    VOLUME 5
20    SOUND OFF
```

SOUND ON

(restores the volume)

SOUND ON

also see SOUND OFF

Description

After the SOUND OFF command, SOUND ON restores the volume level. Before the SOUND OFF command, SOUND ON sets the master volume to its default (zero).

Example

```
20      SOUND ON
```

SPRITE

(checks sprite-to-sprite collision)

SPRITE <sprite number>

Description

SPRITE checks for sprite-to-sprite collision, and evaluates to true if the specified sprite previously came in contact with another sprite. However, you cannot know which sprite contacted which sprite. Once SPRITE is called, the sprite-to-sprite collision flag clears, but sets again when the sprite collides with another sprite.

For moving sprites, their collision flags remain set even when the sprites move away from each other. To clear these flags, execute the SPRITE CLEAR HIT command. Use the SPRITE command to check whether a sprite has hit or is hitting another sprite. To know if a sprite is in collision at any instant, use the SPRITE CLEAR HIT command before you use SPRITE.

Sprite range is from 1 to 8.

Example

```
10    SPRITE 1 ON AT 150,100 SPEED 1,0 COLOR RED SHAPE 1
20    SPRITE 2 ON AT 300,100 SPEED -4,0 COLOR GREEN SHAPE 1
30    SPRITE MOVE
40    FOR T = 1 TO 100: NEXT T: SPRITE CLEAR HIT
50    IF SPRITE(1) AND SPRITE(2) THEN SPRITE 1 SPEED -1,0: SPRITE 2 SPEED 4,0
60    FOR T = 1 TO 100: NEXT T: SPRITE CLEAR HIT
70    IF SPRITE(1) AND SPRITE(2) THEN SPRITE 1 SPEED 1,0: SPRITE 2 SPEED -4,0
80    GOTO 40
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE ANIMATE OFF

(turns off sprite animation)

SPRITE <sprite number> ANIMATE OFF

also see SPRITE ANIMATE ON
SPRITE ANIMATE SPEED
SPRITE MOVE

Description

SPRITE ANIMATE OFF turns off animation for an individual sprite.

Example

```
30  SPRITE 1 ANIMATE OFF
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE ANIMATE ON

(turns on sprite animation)

SPRITE <sprite number> ANIMATE ON

also see SPRITE ANIMATE OFF
SPRITE ANIMATE SPEED
SPRITE MOVE

Description

SPRITE ANIMATE ON turns on animation for individual sprites.

Example

```
20    SPRITE 1 ANIMATE ON
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE ANIMATE SPEED

(selects the shapes to flip between)

SPRITE <sprite number> ANIMATE <shape1>,[<shape2>]...SPEED<speed>

also see SPRITE ANIMATE ON
SPRITE ANIMATE OFF
SPRITE MOVE

Description

After both SPRITE ANIMATE ON and SPRITE MOVE execute, SPRITE ANIMATE SPEED selects the shapes to flip between. The shape changes automatically through the sequence of <shape1>,<shape2>...<shape16>, then repeats the sequence. (Sixteen is the maximum allowed shapes.)

SPEED controls how quickly the program goes through the shape sequence. The SPEED range is 0 to 127. A SPEED of 60 changes the shape approximately once every second.

Example

```
40    SPRITE 1 ANIMATE 10,11,12,13,14,13,12,11 SPEED 45
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8; <shape> is not in the range 1 to 16; or <speed> is not in the range 0 to 127.

SPRITE AT

(positions sprites)

SPRITE <sprite number> AT <sprite x>,<sprite y>

Description

SPRITE AT positions sprites at specific screen locations.

The <sprite number> range is 1 to 8.

Sprites are set relative to the origin.

When <sprite x>,<sprite y> exceeds the screen boundary, the sprite wraps to the other side of the screen. The WINDOW command doesn't effect sprites.

Example

```
10  SPRITE 1 ON COLOR GREEN SHAPE 10
20  SPRITE 1 AT 100,146
```

SPRITE CLEAR HIT

(clears sprite collision flags)

SPRITE [<sprite number>] CLEAR HIT

Description

SPRITE CLEAR HIT clears a single sprite's, or all sprites', sprite-collision flags. Use this command before the SPRITE and BACKGROUND commands to check whether a sprite is currenly in collision.

To see if sprites were in contact in the past, use the SPRITE and BACKGROUND commands before the SPRITE CLEAR HIT command (or alone) to indicate collision status since the flags were last cleared.

Example

```
20  SPRITE 1 CLEAR HIT
```

SPRITE COLOR

(sets a sprite's color)

SPRITE <sprite number> COLOR <sprite color>

also see SPRITE MULTICOLOR

Description

SPRITE COLOR sets the color for a sprite. Use either the color number or the color name (see Appendix C).

In HIRES mode, the whole sprite appears in the given color. For multicolor sprites, this command sets only one sprite color. To set the other two colors, use the SPRITE MULTICOLOR command.

Example

```
10  SPRITE 3 COLOR PURPLE
20  X=7
30  SPRITE X COLOR 5
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8, or <sprite color> is not in the range 0 to 15.

SPRITE FREEZE

(stops sprite animation or movement)

SPRITE FREEZE

also see SPRITE MOVE

Description

SPRITE FREEZE stops sprite animation and movement until you use the SPRITE MOVE command.

Use this command to synchronize sprite movement and animation.

Example

```
10  SPRITE 1 SHAPE 1 ON AT 10,30
20  SPRITE 1 SPEED 1,1
30  SPRITE FREEZE
```

SPRITE HIRES

(puts sprite in HIRES mode)

SPRITE <sprite number> HIRES

Description

SPRITE HIRES puts an individual sprite into HIRES mode.

Example

```
20    SPRITE 3 HIRES
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE LOAD

(loads sprite shapes)

SPRITE LOAD "<file name>"[,<device number>]

also see SPRITE SAVE

Description

SPRITE LOAD loads the sprites shapes that you saved through the SPRITE SAVE command or were created by the SPRITE Editor. The <device number> default is 8. The sprites return to the locations from which they were saved.

You can use SPRITE LOAD in a program. The sprites are then loaded to the sprite editor and the program continues at the next BASIC statement.

If a disk error occurs, the load aborts with no error message; only the red light on your disk drive flashes.

You can use SPRITE LOAD to vector-load machine language programs: this action does not affect BASIC memory pointers.

Example

```
20    SPRITE LOAD "CACTUS"
```

SPRITE MOVE

(turns on sprite movement or animation)

SPRITE MOVE

also see SPRITE FREEZE

Description

SPRITE MOVE turns on sprite movement and animation after a SPRITE FREEZE command. Then, all sprites that have a predefined speed move, and all sprites that have predefined animation sequences begin to animate (as if you'd used the SPRITE ANIMATE ON command).

Use this command to synchronize sprite movement or animation.

Example

```
20  SPRITE 1 SHAPE 1 ON AT 10,30
30  SPRITE 1 SPEED 1,1
40  SPRITE MOVE
```

SPRITE MULTI

(puts a sprite into MULTI mode)

SPRITE <sprite number> MULTI

also see SPRITE COLOR
SPRITE MULTICOLOR

Description

SPRITE MULTI puts an individual sprite into MULTI mode. The sprite colors correspond to those you set in the SPRITE COLOR and SPRITE MULTICOLOR commands.

Example

```
10  SPRITE 1 MULTI
10  SPRITE 3*3 MULTI
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE MULTICOLOR

(defines two sprite colors)

SPRITE MULTICOLOR <sprite multicolor1>,<sprite multicolor2>

also see SPRITE COLOR
SPRITE MULTI

Description

SPRITE MULTICOLOR defines two colors for multicolor sprites.

Example

```
10  SPRITE MULTICOLOR BLUE,GREEN
20  SPRITE 1 ON AT 100,100 SHAPE 10 SPEED 1,0
30  SPRITE 1 MULTI
40  SPRITE 1 COLOR RED
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8; <sprite multicolor1> or <sprite multicolor2>, or both, are not in the range 0 to 15.

SPRITE OFF

(turns a sprite off)

SPRITE <sprite number> OFF

also see SPRITE ON

Description

SPRITE OFF turns an individual sprite off. When a sprite is off, it is no longer displayed, but it retains its shape, position, and any other parameters.

Example

```
20  SPRITE 2 OFF
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE ON

(turns a sprite on)

SPRITE <sprite number> ON

Description

SPRITE ON turns an individual sprite on.

Example

```
30  SPRITE 7 ON
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE ON BACKGROUND

(places a sprite on top of the background)

SPRITE <sprite number> ON BACKGROUND

also see SPRITE UNDER BACKGROUND

Description

SPRITE ON BACKGROUND places a sprite on top of the background in front of any characters or graphics that also appear at that location. This command does not affect sprite-to-sprite priority.

Example

```
10  SPRITE 4 ON BACKGROUND
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE UNDER BACKGROUND

(places a sprite behind the background)

SPRITE <sprite number> UNDER BACKGROUND

also see SPRITE ON BACKGROUND

Description

SPRITE UNDER BACKGROUND places a sprite behind any character or graphic that appears in the same location. This command does not affect sprite-to-sprite priority.

Example

```
20    SPRITE 2 UNDER BACKGROUND
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

SPRITE SAVE

(saves the sprite shape)

SPRITE SAVE <first shape>,<last shape>,"<file name>"[,<device number>]

also see SPRITE LOAD

Description

SPRITE SAVE saves sprite shapes to disk or cassette. The default device is 8.

If a disk error occurs, the SPRITE SAVE aborts, and the program returns to Toolkit BASIC, but no error message appears; the red light on the disk drive flashes.

Examples

```
20  SPRITE SAVE 14,14 "HOPIBOB"
20  SPRITE SAVE 10,17 "POTTERY",1
```

SPRITE SHAPE

(assigns a shape to a sprite)

SPRITE <sprite number> SHAPE <shape number>

Description

SPRITE SHAPE assigns a shape to the selected sprite. The sprite editor allows 32 shapes. Each sprite can have a different shape, or sprites can share shapes.

To find a sprite shape in memory, calculate:

First shape location = (shape number + 31) * 64

Shape 1 starts at memory location 2048, which is the space immediately in front of your Toolkit BASIC program, so take care when you POKE data directly into this area.

Example

```
10  A=2
20  SPRITE 1 SHAPE 4
30  SPRITE A SHAPE 4
40  SPRITE 8 SHAPE 16+3
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8, or <shape number> does not reference a valid memory location.

SPRITE SPEED

(defines the speed of the sprite)

SPRITE <sprite number> SPEED <x speed>,<y speed>

Description

SPRITE SPEED defines the selected sprite's x and y speeds. 0,0 keeps the sprite still; 5,5 moves the sprite rapidly to the right. The range is 127 to -128.

Remember that motion doesn't start until you issue the SPRITE MOVE command.

Examples

```
10  SPRITE 1 SPEED -1,0
10  SPRITE 3 SPEED -5,2
20  A=8
30  SPRITE 4 SPEED A,A*6
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8, or <x speed>,<y speed> is not in the range 127 to -128.

SPRITE XYSIZE

(sets a sprite's horizontal and vertical size)

SPRITE <sprite number> XYSIZE <x factor>,<y factor>

Description

SPRITE XYSIZE sets a selected sprite's horizontal and vertical size. 1,1 sets the sprite to its normal size; 2,2 doubles the horizontal and vertical sizes. The range is 1 to 2.

Examples

```
20  SPRITE 2 XYSIZE 2,2
20  SPRITE 4 XYSIZE 2,1
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8, or <x factor>,<y factor> is not 1 or 2.

TEXT

(switches to TEXT mode)

TEXT

also see HIRES
MULTI
TEXT FROM TO

Description

TEXT places your computer into TEXT mode, which is the mode that your computer is in when you turn it on.

TEXT FROM TO

(splits the screen into two sections)

TEXT [FROM <firstline>] [TO <lastline>]
TEXT
TEXT TO <line>
TEXT FROM <line>

Description

TEXT FROM TO splits the screen into two sections: one for text and one for the previous graphics mode.

If you omit FROM <firstline>, text appears from the top of the screen TO <lastline>.

If you omit TO, text appears FROM <firstline> to the bottom of the screen.

The range for <firstline> and <lastline> is 1 to 25.

Examples

```
20   TEXT FROM 10 TO 10
20   TEXT FROM 25
20   A = 4: B = 7
30   TEXT FROM A TO B
20   TEXT TO 17
```

Error Message

ILLEGAL QUANTITY ERROR

<firstline> or <lastline>, or both, are not in the range 1 to 25.

TEXT LOAD

(retrieves text)

TEXT LOAD "<file name>" [,<device>]

also see TEXT SAVE

Description

TEXT LOAD retrieves the text screen from disk created by TEXT SAVE or the BACKGROUND/Font Editor. The default device is 8.

Examples

```
TEXT LOAD "GARDEN",9

TEXT LOAD A$,D
```

TEXT SAVE

(saves text)

TEXT SAVE "<file name>" [,<device>]

also see TEXT LOAD

Description

TEXT SAVE saves the text screen to disk or cassette. The default device is 8.

Examples

```
TEXT SAVE "CACTUS"
TEXT SAVE "BROMELIAD",9
```

UNNEW

(reverses the action of the NEW command)

UNNEW

Description

Use UNNEW when you have used the NEW command by accident, or when you change your mind about using the NEW command.

Remember that if you type in a new BASIC line or load a program after the READY prompt, you lose your chance to use UNNEW.

Use UNNEW in direct mode.

Example

```
UNNEW
```

VOICE ADSR

(determines the volume envelope)

VOICE <voice number> ADSR <attack>,<decay>,<sustain>,<release>

also see VOICE FREEZE
VOICE GO
SOUND OFF
SOUND CLEAR

Description

VOICE ADSR determines the form of the volume envelope for a selected voice.

<attack>, <decay>, and <release> determine the rate at which the volume level changes. The range is from 0 (instantaneous change) to 15 (change over several seconds).

<sustain> is a volume setting.

When the VOICE GO command executes, <attack> causes the volume to rise at the <attack> rate. After peaking, the volume falls at the <decay> rate until it reaches the <sustain> level. The <sustain> level remains in effect until a VOICE FREEZE command executes; then the volume falls at the <decay> rate until the volume is zero. See Figure 6-5.

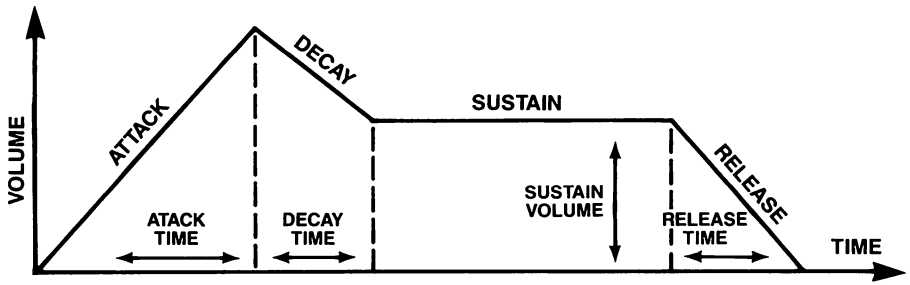


Figure 6-5. VOICE ADSR Sequence

You can issue SOUND OFF to stop the sound.
SOUND CLEAR resets ADSR values to zero.

Example

```
10      VOLUME 15
20      VOICE 2 WAVE 2 ADSR 6,0,15,0 TONE 10000
30      VOICE 2 ON
RUN
```

Error Message

ILLEGAL QUANTITY ERROR

<voice number> is not in the range 1 to 3, or an ADSR is not in the range 0 to 15.

VOICE FREEZE

(stops the automatic voice sequence)

VOICE <voice number> FREEZE

also see VOICE GO

Description

VOICE FREEZE stops the automatic voice sequence. VOICE FREEZE stops the voice; VOICE GO turns it on again.

Example

```
20    VOICE 1 WAVE 1 ADSR 0,0,15,0 PLAY 1000,4000 SPEED 10
30    VOICE 1 FREEZE
```

Error Message

ILLEGAL QUANTITY ERROR

<voice number> is not in the range 1 to 3.

VOICE GO

(starts the automatic voice sequence)

VOICE <voice number> GO

also see VOICE FREEZE

Description

VOICE GO starts the automatic voice sequence. SOUND GO must precede VOICE GO for the voice to be played.

SOUND FREEZE stops the voice; SOUND GO turns it on again.

Example

```
20    VOICE 1 WAVE 1 ADSR 0,0,15,0 PLAY 1000,4000, SPEED 10
30    VOICE 1 GO
```

Error Message

ILLEGAL QUANTITY ERROR

<voice number> is not in the range 1 to 3.

VOICE OFF
(turns off a voice)

VOICE <voice number> OFF

Description

VOICE OFF turns off a single voice. <voice number> is in the range from 1 to 3.

Error Message

ILLEGAL QUANTITY ERROR

<voice number> is not in the range 1 to 3.

VOICE ON
(turns on a voice)

VOICE <voice number> ON

Description

VOICE ON turns on a single voice. <voice number> is in the range from 1 to 3.

Error Message

ILLEGAL QUANTITY ERROR

<voice number> is not in the range 1 to 3.

VOICE PLAY
(defines a voice's tone sequence)

VOICE <voice number> PLAY [CONT] <tone> [;<note duration>] [;<release time>],...[←] [SPEED <speed>]

Description

VOICE PLAY defines a voice's tone sequence for automated sound. The number of notes that you can assign to any one voice is 63.

<tone> values play in sequence at the selected <speed>. ← causes the sequence to repeat endlessly.

<note duration> defines the time between the attack start and the release start.

<release time> defines the time between the release start and the next note start.

To calculate the seconds between the start of each note:
time = <note duration> + <release time> = 1 * <speed> + 1 /60

When you use CONT after PLAY, you can define the notes in the sequence through separate PLAY commands, or use one command in a loop. For example,

FOR I = 1 TO 3: VOICE 1 PLAY CONT 1000*I SPEED 10: NEXT

Note that the automated sound does not always wait for a decay or attack before starting the next note or releasing the present note when you use short notes with long attack, release, and decay times.

Error Message

ILLEGAL QUANTITY ERROR

The number of notes to be played in any one voice exceeds 63.

VOICE TONE

(sets the voice tone)

VOICE <voice number> TONE <tone number>

Description

VOICE TONE sets a voice's tone. <tone number> causes the voice to play different notes. (See Appendix F for a list of the tones.) The range for <tone number> is 0 to 65535.

Example

```
10  VOLUME 15
20  VOICE 1 TONE 6000 ADSR 0,0,15,0
30  VOICE 1 ON
```

Error Message

ILLEGAL QUANTITY ERROR

<voice number> is not in the range 1 to 3, or <tone number> is not in the range 0 to 65535.

VOICE WAVE

(selects a voice's waveform)

VOICE <voice number> WAVE <wave number> [,<pulse width>]

Description

VOICE WAVE selects a voice's waveform. The waveforms available are:

WAVE 1 (WAVE TRIANGLE)	Triangle
WAVE 2 (WAVE SAW)	Saw
WAVE 3,<pulse width> (WAVE PULSE,<pulse width>	Pulse (range 0 to 1024)
WAVE 4 (WAVE NOISE)	Noise

Figures 6-6 through 6-9 illustrate these waveforms.

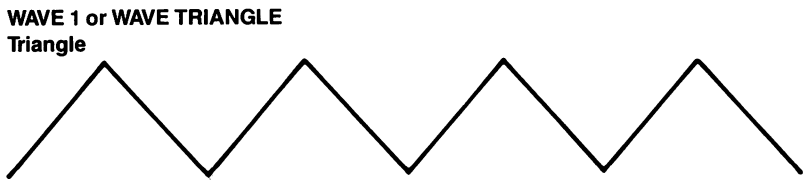


Figure 6-6. Triangle

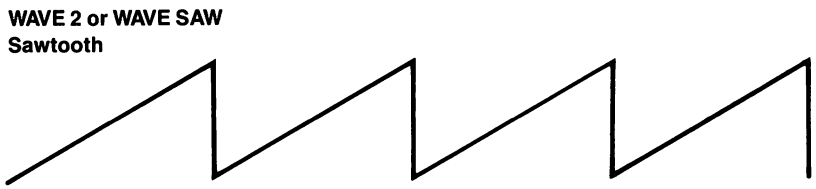


Figure 6-7. Sawtooth

WAVE 3, <pulse width> or WAVE PULSE, <pulse width>
Pulse

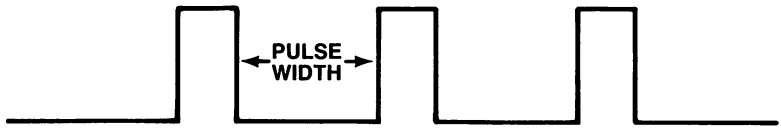


Figure 6-8. Pulse

WAVE 4 or WAVE NOISE
Noise

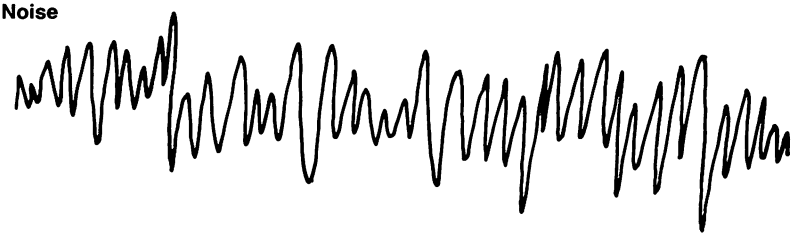


Figure 6-9. Noise

Examples

```
20  VOICE 2 WAVE 1
20  W = 4
30  VOICE 2 WAVE W
20  VOICE 3 WAVE 2,545
```

Error MessageS

ILLEGAL QUANTITY ERROR

<voice number> is not in the range 1 to 3; <wave number> is not in the range 1 to 4; or <pulse width> is not in the range 1 to 1024.

SYNTAX ERROR

A <pulse width> cannot follow wave 3.

VOLUME

(master volume control)

VOLUME <level>

Description

VOLUME is the master volume control for all three voices. The range is from 0 (no sound) to 15.

Examples

```
20  VOLUME 15
20  VOLUME 5
20  FOR I = 0 TO 15
30  VOLUME I
40  NEXT I
```

Error Message

ILLEGAL QUANTITY ERROR

<level> is not in the range 0 to 15.

WINDOW

(sets up a window)

WINDOW [<lowerleft x>,<lowerleft y>,<upperright x>,<upperright y>]
WINDOW

also see ORIGIN
SCALE

Description

WINDOW sets up a window on the screen. The window clips lines or dots inside the designated area. The default is WINDOW = screen size. The window coordinates are relative to the origin and affected by the scale.

WINDOW allows you to draw figures larger than the screen; you see parts of the figure as if you were looking through a window.

Example

```
30    WINDOW 180,50,310,160
```

XPOS

(returns the selected sprite's x-coordinate)

XPOS <sprite number>

Description

XPOS returns the selected sprite's x-coordinate. ORIGIN and SCALE affect this value.

Example

```
20    X=XPOS(2)
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

YPOS

(returns the selected sprite's y-coordinate)

YPOS <sprite number>

Description

YPOS returns the selected sprite's y-coordinate. ORIGIN and SCALE affect this value.

Example

```
20    Y = YPOS(5)
```

Error Message

ILLEGAL QUANTITY ERROR

<sprite number> is not in the range 1 to 8.

Appendix A CAUTIONS

Toolkit BASIC extends the BASIC that already exists in your Commodore computer. Your standard Commodore BASIC programs won't have any trouble running under Toolkit BASIC. However, programs that don't check memory before POKEing in data, programs that patch onto standard BASIC, and programs that use memory locations that Toolkit BASIC also uses may not run.

Be careful, especially, in the following areas: with vectors, interrupts, and disk files.

VECTORS

Toolkit BASIC uses the following vectors, so don't change them:

$3014-$3015 IRQ interrupt routine
$0318-$0319 NMI interrupt routine
$0330-$0331 LOAD routine
$0332-$0333 SAVE routine
$0326-$0327 Character-output routine

INTERRUPTS

Don't disable or change interrupts from BASIC or machine language when you're using Toolkit BASIC. If any program modifies the IRQ interrupt routine, all Toolkit BASIC interrupt-driven features become inoperable (TEXT, HIRES, MULTI, SPRITE SPEED, SPRITE ANIMATE, AND VOICE PLAY).

DISK FILES

When you're using a Toolkit BASIC interrupt-driven feature, don't also use an OPEN command on a disk file.

Before you use a disk file, execute the RESET command to disable interrupt-driven functions. If you forget to use RESET, and the computer halts, press RUN STOP and RESTORE at the same time: this action restarts Toolkit BASIC, but your program and variables are retained.

Appendix B DISK INFORMATION

FORMATTING A DISK

You need to format a disk before you can use it to store programs on. To format a disk,

1. Insert a blank disk into the disk drive.
2. Type DISK "NO:<name>,ID"
   where <name> is the disk name and ID is the disk number.
3. Press RETURN.
4. Watch for the red light on the disk drive: it stays on for about a minute while the disk is being formatted. When the red light goes out, you can remove the formatted disk, or leave it in the disk drive to store programs on.

NOTE

When you format a disk, you erase any files that already may exist on that disk.

Caring For Disks

Your disks need to last a long time and be reliable. To help them along:

- Keep disks away from magnetic fields and heat. Don't leave them on your TV or monitor, on top of the disk drive, the power supply, the printer, or a telephone. Be sure no magnets are in the area. Don't leave your disks where the sun can get to them or near a heater.
- Don't touch the exposed parts of disk. These are the shiny parts that you can see through the holes in the disk envelope.
- Store your disks in their envelopes, and leave them to rest in the vertical position.

Appendix C COLORS

Toolkit BASIC gives you 16 colors to play with. You can use either the color names or the color numbers in any command.

0	black	8	peach
1	white	9	brown
2	red	10	pink
3	cyan	11	gray1
4	purple	13	gray2
5	green	13	lgreen
6	blue	14	sky
7	yellow	15	gray3

Appendix D MEMORY MAP

Toolkit BASIC reconfigures part of the Commodore 64's memory. Zero page, however, remains the same except for four memory locations that Toolkit BASIC uses: $00FB-$00FE.

Figures D-1 through D-5 illustrate the memory configuration when Toolkit BASIC is installed.

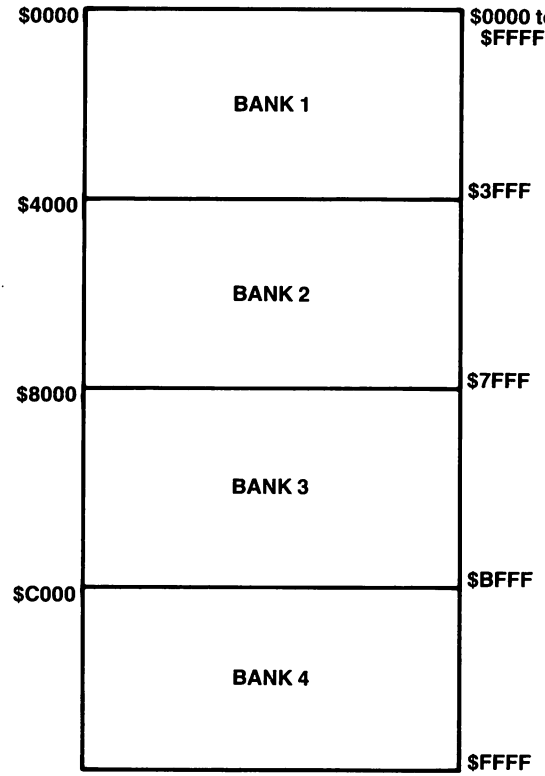


Figure D-1. Memory Map Overview

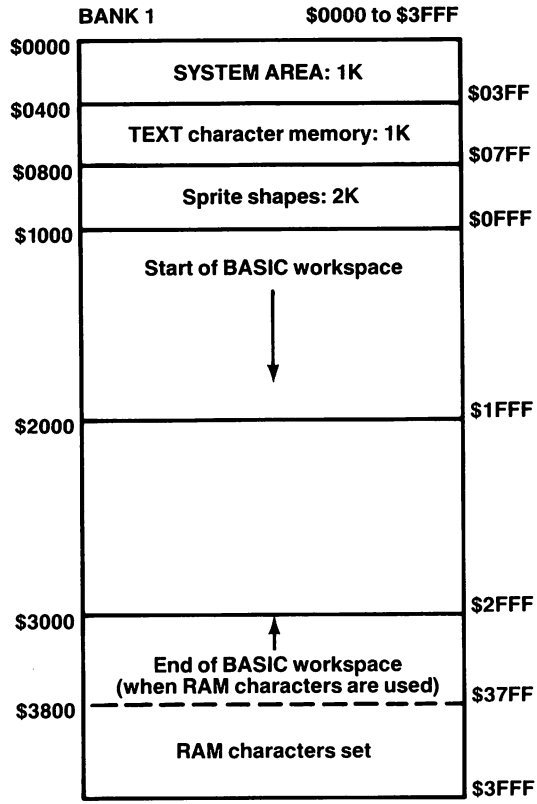


Figure D-2. Bank 1

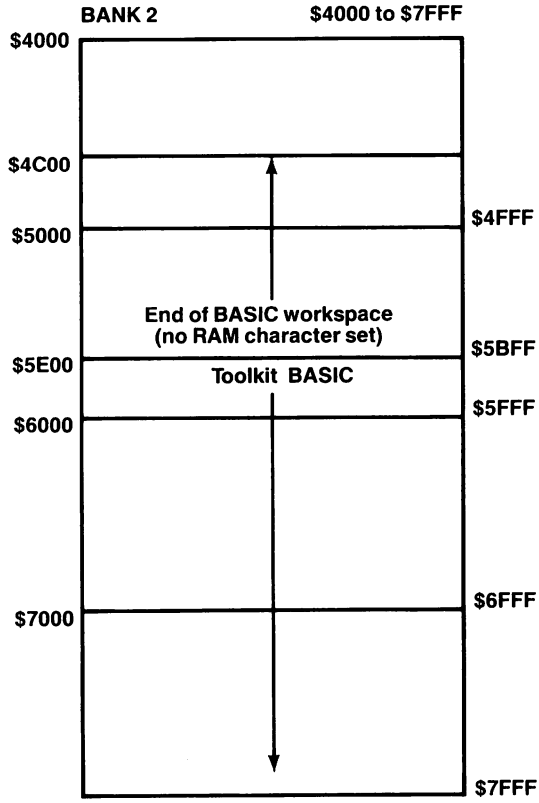


Figure D-3. Bank 2

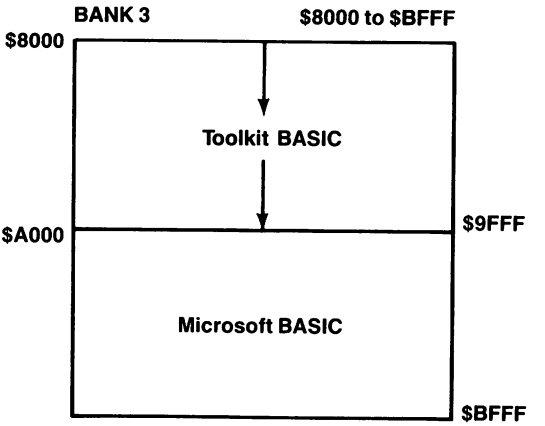


Figure D-4. Bank 3

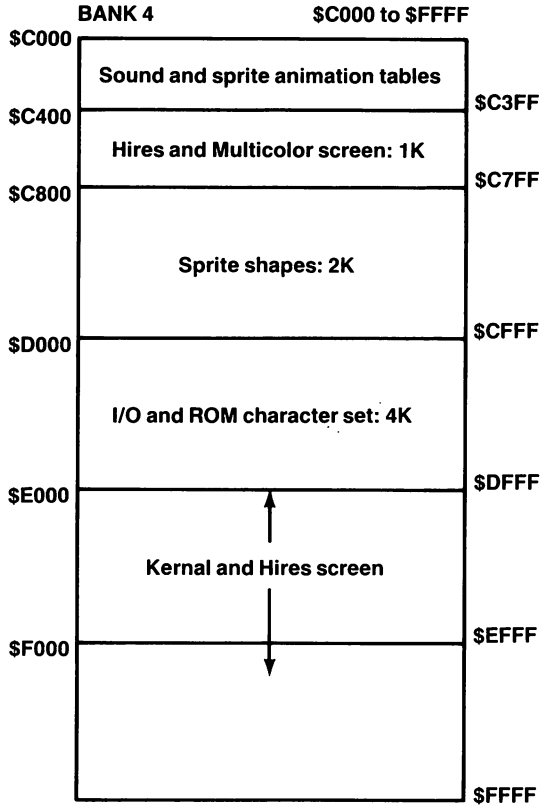


Figure D-5. Bank 4

Appendix E MUSIC NOTES

Table E-1 lists the notes available in Toolkit BASIC for the 4th octave (you have use of 7) along with the octave, tone number, and note number.

When you need a note you need doesn't appear in the table, you can calculate it:

<Tone number> = (1.059463157)↑ <note number> * 59059 * 2↑ <octave> −7)

where <note number> is the number of the note you need (get its value from the table), and <octave> is in the range 0 to 6, which cooresponds to the 7 octaves available.

Table E-1. Notes

Note	*Octave*	*Tone Number*	*Note Number*	
A	4	7382	0	440 Hz
Bb	4	7821	1	
B	4	8286	2	
C	4	8779	3	
C#	4	9301	4	
D	4	9854	5	
D#	4	10440	6	
E	4	11061	7	
F	4	11718	8	
F#	4	12415	9	
G	4	13253	10	
G#	4	13935	11	

Appendix F ERROR MESSAGES

When Toolkit BASIC encounters an error in your program, it displays the line that contains the error, with reverse-video angle brackets < > at the error location.

You can move the cursor into the displayed line and make the correction (be sure to remove the angle brackets).

Or, you can retype the line below the READY prompt.

Table F-1 lists the error messages. See your Commodore Programmer's Reference Guide for detailed explanations.

Table F-1. Error Messages

Error Number	*Message*	*Error Number*	*Message*
1	Too many files	16	Out of memory
2	File open	17	Undefined statement
3	File not open	18	Bad subscript
4	File not found	19	Redimensioned array
5	Device not present	20	Division by zero
6	Not input file	21	Illegal direct
7	Not output file	22	Type mismatch
8	Missing file name	23	String too long
9	Illegal device number	24	File data
10	Next without For	25	Formula too complex
11	Syntax	26	Can't continue
12	Return without Gosub	27	Undefined function
13	Out of data	28	Verify
14	Illegal quantity	29	Load
15	Overflow	30	Break

Appendix G FILE DESCRIPTIONS

Following are the file descriptions for the Programmer's BASIC Toolkit. All files are sequential.

CHARACTER SET FILE DESCRIPTION

Byte	*Description*
0-1	"CH" <ascii>
2	HIRES/MULTI (value to be restored to location $D016)
3	Multicolor 1
4	Multicolor 2
Next 2048	Character set definition
Next 256	Character colors for editor's use

TEXT/BACKGROUND FILE DESCRIPTION

Byte	*Description*
0-1	"TX" <ascii>
2	40 (decimal) number of columns
3	25 (decimal) number of rows
4	Background color
5	Multicolor 1
6	Multicolor 2
7	HIRES/MULTI (value to be restored to location $D016)
8	Reserved
Next 1000	Screen names
Next 1000	Color RAM data

SPRITE FILE DESCRIPTION

Byte	*Description*
0-1	"SP" <ascii>
2	32 (number of sprite shapes)
3	X expand (for editor's use)
4	Y expand (for editor's use)
5	Sprite multicolor 1
6	Sprite multicolor 2
7	Priority (1=under background)
8-15	Sprite colors

16	0 (version number)
17-18	Pointer into the next empty location for the sprite multiple definition
19-23	Reserved
Next 32×64	32 sprite definitions consisting of 63 bytes of shape and 1 byte of attribute

Bit 7	Hires/multi (1=Multi)
Bit 6	X expand
Bit 5	Y expand
Bit 4	Priority
Bit 3-0	Color

Next 16×26	16 multiple sprite definitions consisting of 26 bytes each.
	6 bytes of multiple definition status (1=yes)
	6 bytes display box x offsets
	6 bytes display box y offsets
	6 bytes shape names (6-31)
	1 byte of x expand (top 6 bits used)
	1 byte of y expand (top 6 bits used)

HIRES FILE DESCRIPTION

Byte	*Description*
Byte 0	"H" (ascii)
Next 1800	Bitmap data
Next 1000	Color RAM data

MULTI FILE DESCRIPTION

Byte	*Description*
Byte 0	"M" (ascii)
Next 8000	Bitmap data
Next 1000	Multicolor color screen data
Next 100	Color RAM data

Appendix H DEMONSTRATION PROGRAMS

To use the demonstration programs.

1. At the READY prompt, type:
   LOAD "<filename>",8
2. When the READY prompt reappears, type
   RUN

Table H-1 lists the demonstration programs.

Table H-1. Demonstration Programs

AIRPLANE DEMO
DRIVING DEMO
ELLIPSE DEMO
HATMAN RACE DEMO
JOYSTICK DEMO
BITPADMULT (Requires a graphics tablet in port 1)
MULTIBAR DEMO
SPLIT DEMO
WINDOW DEMO
8BALLS DEMO

NOTE

The file CONVERTER converts Koala files to sequential files that are compatible with the Toolkit's background editor.

1043 Kiel Court
Sunnyvale, CA 94089

Programmers' BASIC Toolkit was developed by Ron Gilbert, Tom McFarlane, and Softalent. Additional programming and graphics by Tony Garcia. Commodore 128 is a trademark, and Commodore 64 is a registered trademark of Commodore Electronics Limited. Programmers' BASIC Toolkit, Vorpal and Fast Load Cartridge are trademarks of Epyx, Inc.

LIMITED WARRANTY

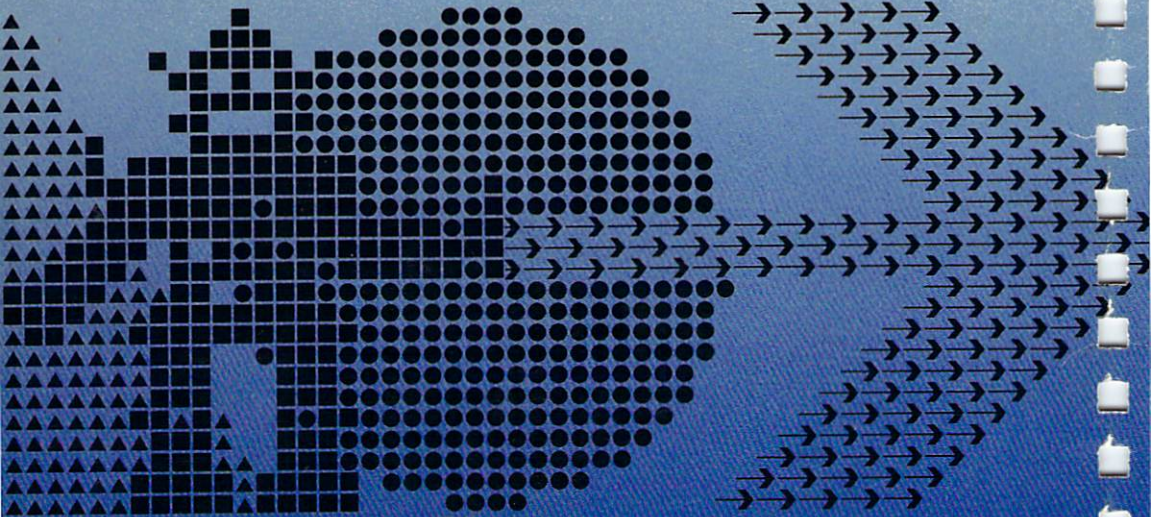
Epyx, Inc., warrants to the original purchaser of this Epyx software product that the medium on which this computer program is recorded is free from defects in materials and workmanship for a period of ninety (90) days from the date of purchase. This Epyx software program is sold "as is," that is without express or implied warranty of any kind, and Epyx is not liable for any losses or damages of any kind resulting from use of this program. Epyx agrees for a period of ninety (90) days to either repair or replace, at its option, free of charge, any Epyx software product, postage paid, with proof of date of purchase, at its Factory Service Center.

This warranty is not applicable to normal wear and tear. This warranty shall not be applicable and shall be void if the defect in the Epyx software product has arisen through abuse, unreasonable use, mistreatment or neglect. THIS WARRANTY IS IN LIEU OF ALL OTHER EXPRESS WARRANTIES AND NO OTHER REPRESENTATION OR CLAIMS OF ANY NATURE SHALL BE BINDING ON OR OBLIGATE EPYX. ANY IMPLIED WARRANTIES APPLICABLE TO THIS SOFTWARE PRODUCT, INCLUDING WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED TO THE NINETY (90) DAY PERIOD DESCRIBED ABOVE. IN NO EVENT WILL EPYX BE LIABLE FOR ANY SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGE RESULTING FROM POSSESSION, USE OR MALFUNCTION OF THIS EPYX SOFTWARE PRODUCT.

Some states do not allow limitations as to how long an implied warranty lasts and/or the exclusion or limitation of incidental or consequential damages so the above limitations and/or exclusions or limitation of liability may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Proof-of-Purchase
Product No. 9127D
EPYX

Part No. 91207D-60



EPYX®