Programs created with Super Expander's special statements will not work on a machine without the Super Expander plugged in.

Finally, the extra 3K memory expansion is provided automatically when you plug in the Super Expander ... you will see the screen come up with approximately 3ØØØ extra bytes of memory when you turn it on.

GETTING STARTED

1. Turn on your television set.
2. Turn your VIC 2Ø off.
3. Insert the cartridge.
4. Turn the VIC 2Ø on.
5. Your Super Expander is now activated and you may write programs in BASIC using the special commands, statements and functions listed below.

II. USING SUPER EXPANDER GRAPHICS

A few special notes are in order before you begin working through the manual. To begin with, it should be noted that for purposes of graphics plotting, the Super Expander cartridge divides the screen area into 1Ø24, by 1Ø24 coordinates. For example, if you type POINT1,512,512 you will draw a point at the very middle of the screen. These are "pseudo-coordinates" which means the VIC accepts inputs based on 1Ø24x1Ø24 dot coordinates and scales the plotting down to the VIC's actual screen size/resolution.

A full range of graphic instructions is automatically added to your VIC's standard BASIC instruction set when you plug in the Super Expander cartridge. These instructions are listed and explained below.

NOTE: WHEN WORKING WITH GRAPHICS THERE ARE TWO IMPORTANT PARAMETERS WHICH YOU MUST INCLUDE AT THE BEGINNING OF YOUR PROGRAM. These set up the graphic screen so you can plot, include characters mixed with graphics, etc. The two graphic commands you should begin your program with are: GRAPHIC and COLOR. GRAPHIC sets the VIC in the "GRAPHIC MODE" you want to use, and "COLOR" specifies the colors you will be working with.

INSTRUCTION	EFFECT
GRAPHIC	Prepares the screen for graphics
SCNCLR	Clears a graphic screen
COLOR	Select screen, border, character, and auxiliary colors
REGION	Select character color
DRAW	Plots a line between two points
POINT	Plots a single point
CIRCLE	Draw a circle, ellipse, or arc
PAINT	Fill in an enclosed area in a color
CHAR	Puts text on the graphic screen
SOUND	Set 4 tones and volume all at once

FUNCTION	RETURN VALUE
RGR(x)	Current graphic mode
RCOLR(x)	The value in a color register
RDOT(x,y)	The color of a point on the screen
RPOT(x)	The position of the game paddle
RPEN(x)	The position of the light pen
RJOY(x)	The position of the joystick
RSND(x)	The value in a sound register

GRAPHIC n

This sets up the graphic screen and activates the graphics instruction set. There are 5 different GRAPHIC modes, which are activated by typing GRAPHIC and the number, as an instruction in your program.

Example:

```
1Ø GRAPHIC 2
2Ø COLOR 1, 2, 3, 4
Type RUN and hit RETURN.
```

Value of n	Mode
Ø	Text mode (normal)
1	Multi-color mode
2	High resolution mode
3	Mixed multi and high-res
4	Return to text mode

Mode Ø is the VIC's normal text mode. Once the mode has been changed, the GRAPHIC 4 statement is used to return to this mode.

In mode 1, there are 8Ø points across a line, and 16Ø lines. Each point is drawn in one of the 4 color registers: screen color, border color, character color, or auxiliary color. A point placed in the screen color is invisible, since it is the same as the background; this can be used to erase other points. Points drawn in either the screen, border, or auxiliary colors will change instantly when these colors are changed. The character color is associated with each specific 4 dot by 16 dot space of the screen. This means that if a dot is plotted in that space in a character color, all other points in that space that were drawn in the character color will change to the new character color.

In mode 2, there are 16Ø points across a line, and 16Ø lines. The points are half as wide as in mode 1, and therefore give twice as fine resolution to the screen. However, the color resolution is more limited. Dots can be plotted only in a character color. Within each specific space on the screen, which is an 8 by 8 dot area, there can be only one character color, so you can only have dots in that color displayed on the screen.

Mode 3 is a mixture of modes 1 and 2. When the character color is less than 8, the plotting is done in high resolution. When the character color is greater than 7, multi-color mode is in effect.

Mode 4 returns the screen to text mode from a graphic mode. WARNING: if the VIC was already in text mode, this will mess up the VIC until you turn it off and then on again.

SCNCLR

This statement clears the entire graphic screen. It may be typed directly or included in a program line.

COLOR sc, bo, ch, au

This sets up the 4 different color registers. The variables here have these limits:

sc = Ø to 15
bo = Ø to 7
ch = Ø to 7 normal, 8 to 15 multi-color
au = Ø to 15

The colors are:

Ø	**black**
1	**white**
2	**red**
3	**cyan**
4	**purple**
5	**green**
6	**blue**
7	**yellow**
8	**orange**
9	**light orange**
1Ø	**pink**
11	**light cyan**
12	**light purple**
13	**light green**
14	**light blue**
15	**light yellow**

NOTE: in GRAPHIC 3 mode, character color numbers greater than 7 set multi-color mode. In this case, the actual character in that space is the color number minus 8.

EXAMPLE: COLOR 1, 2, 3, 4

REGION c

This works like the COLOR statement, except that only the character color is affected.

EXAMPLE: REGION 7

DRAW c, x1, y1 TO x2, y2 [TO x3, y3 . . . TO xn, yn]
or
DRAW c TO x1, y1 [...]

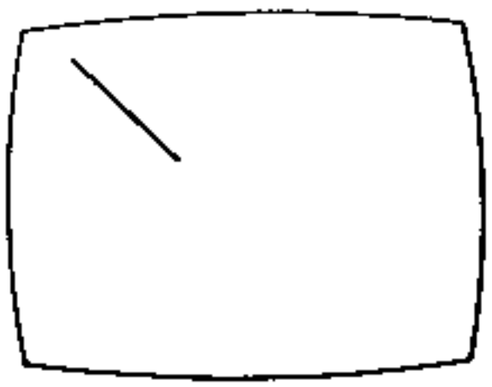
This allows you to draw lines in the screen. The variable c is the color of the line, on the following chart:

c	High-res	Multi-color
Ø	screen	screen
1	character	border
2	character	character
3	character	auxiliary

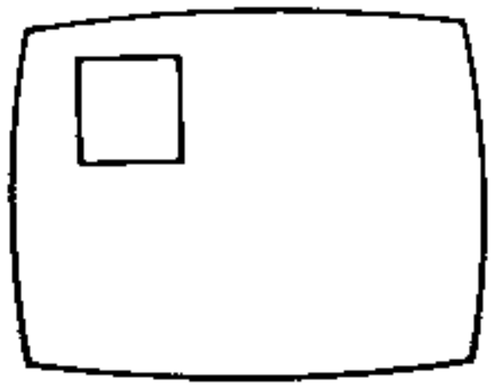
The x and y variables are the coordinates of the end-points of the line. The values of X and Y can be from Ø to 1Ø23. An X value of Ø is the left edge of the screen, and X=1Ø23 is the right edge, and the Y values determine the up-and-down position. The values from Ø to 1Ø23 are scaled down to the actual (16Ø by 16Ø or 8Ø by 16Ø) size of the screen.

EXAMPLE 1: DRAW 1, Ø, Ø TO 1ØØ, 1ØØ
EXAMPLE 2: DRAW 1, Ø, Ø TO Ø, 1ØØ TO 1ØØ, 1ØØ TO 1ØØ, Ø TO Ø, Ø

EXAMPLE 1



EXAMPLE 2



POINT c, x, y
or
POINT c, x1, y1, x2, y2, ---, xn, yn

This sets a point or points on the screen in the specified color. The points are given in the Ø to 1Ø23 form.

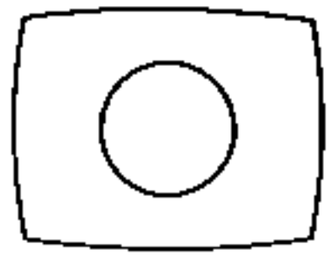
EXAMPLE 1: POINT 1, 5ØØ, 5ØØ
EXAMPLE 2: POINT 2, 5ØØ, 5ØØ, 6ØØ, 6ØØ, 7ØØ, 7ØØ

CIRCLE c, x, y, rx, ry [,as,ae]

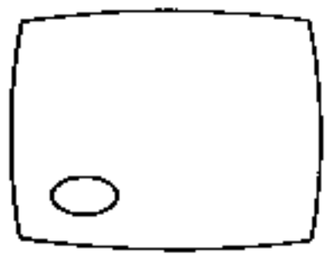
Where: c is the color register (see chart in DRAW statement) in which the circle is drawn
x and y are the coordinates of the center of the circle
rx and ry are width and height of the circle
as and ae are the optional starting and ending angles of the arc, in gradians (Ø to 1ØØ degrees in a full circle)

EXAMPLE 1: CIRCLE 1, 511, 511, 3ØØ, 4ØØ
EXAMPLE 2: CIRCLE 1, 2ØØ, 9ØØ, 1ØØ, 1ØØ
EXAMPLE 3: CIRCLE 2, 5ØØ, 6ØØ, 2ØØ, 3ØØ, 1Ø, 7Ø

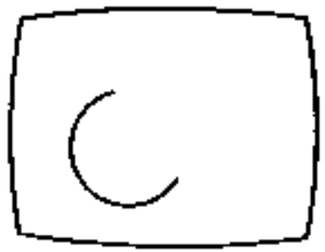
EXAMPLE 1



EXAMPLE 2



EXAMPLE 3



PAINT c, x, y

This paints an enclosed area in a color. c is the color register, and x and y are any point within the area to be painted. Each area can only be painted once. Because of the color limitations of some GRAPHIC modes, some nearby shapes may be distorted by this operation. If an area is bounded by multi-color mode areas, it must be PAINTed in the multi-color mode.

EXAMPLE: 1Ø GRAPHIC 1
2Ø COLOR 1, 7, Ø, 1Ø
3Ø CIRCLE 2, 511, 511, 2ØØ, 3ØØ
4Ø PAINT 3, 511, 511
5Ø PAINT 1, 511, Ø

CHAR ro, co, "text"

This will put normal text on a graphic screen. The ro and co variables are the row and column on which the text is to appear. The text can be any string variable or text inside quote marks. Text should only be displayed when in the high-resolution mode, or the mixed mode, but never in multi-color mode.

EXAMPLE: CHAR 1, 1, "HELLO THERE"

SOUND s1, s2, s3, s4, v

This sets all 4 sound "speakers" and the volume control all at once. The S1 value is put into register 1. If its value is less than 128, no sound will play from that speaker. The volume can range from 0 (off) to 15. This lets you create multi-speaker sound effects.

EXAMPLE: SOUND 225, 225, 225, Ø, 15

RGR(x)

This function reads the graphic mode that was set with the GRAPHIC command. The value of x can be from Ø to 255, and doesn't effect the result.

EXAMPLE: X = RGR(Ø)

RCOLR(x)

This reads the color from any of the 4 registers, depending on the value of x on the following table:

X	COLOR REGISTER
Ø	screen color
1	border color
2	character color
3	auxiliary color

EXAMPLE: X = RCOLR(2)

RDOT(x, y)

This reads the color value of a dot on the screen. The values of x and y are the coordinates of the point, and the color value returned is the register of the color on that space.

EXAMPLE: X = RDOT(511, 511)

RPOT(x)

This reads the value of the paddle. If x = Ø, paddle X is read, and if x = 1, paddle Y is read.

EXAMPLE:
```
1Ø GRAPHIC2:COLOR11,6,6,6,
2Ø X=RPOT(Ø):Y=RPOT(1):J=RJOY(Ø)
3Ø IFJ=12THEN:SCNCLR
4Ø IFJ=4THEN:REGION2
5Ø IFJ=8THEN:REGION6
6Ø POINT2,X*4,Y*4:GOTO2Ø
```

RPEN(x)

This reads the screen coordinate of the light pen. If x=Ø, the X coordinate is returned, and if x=1, the Y coordinate.

EXAMPLE: X = RPEN(Ø)

RJOY(x)

This reads the value of the joystick. The x variable must be between 0 and 255, and has no effect on the outcome.

```
EXAMPLE: 1Ø GRAPHIC2:COLOR11,6,6,6,:X=17Ø:Y=17Ø
         2Ø J=RJOY(Ø)
         3Ø X=X+((JAND4)=4)-((JAND8)=8)
         4Ø Y=Y+((JAND1)=)-((JAND2)=2)
         5Ø POINT3,X*3,Y*3:IFJ=128THEN:SCNCLR
         6Ø GOTO2Ø
```

RSND(x)

This reads the value of any of the sound registers, depending on the value of x from this chart:

x	sound register
1	sound 1
2	sound 2
3	sound 3
4	sound 4
5	volume

III. WRITING SUPER EXPANDER MUSIC

The Super Expander's music writing mode lets you play notes directly by typing on the keyboard, or, more important, by using the PRINT statement in your BASIC PROGRAMS. It is "interrupt driven", which lets music play while the program proceeds with other tasks. It is designed primarily for playing of music, as opposed to sound effects, and makes writing programs with chords much easier.

Music mode is entered by holding down the CTRL key and hitting the **left arrow** (←). When you type this in quotes in a program a "reverse f" should appear. This is a special symbol that shows you where music mode is included in your program. Now any key you type will be interpreted by the music package, until the RETURN key is hit. Here are the characters recognized by music mode:

Character	Effect
P	Screen echo on
Q	Screen echo off
V	Volume selection
S	Sound register selection
O (the letter O, not the number)	Octave selection
T	Tempo selection
R	Rest for 1 beat
C, D, E, F, G, A, B	Play note
#	Play the next note sharp
$	Play the next note flat
RETURN	Ends music mode

When PRINTING strings of these characters, the semicolon (;) can be used to prevent the RETURN character from ending music mode.

The screen echo is normally off. Any characters recognized by the music package are not displayed on the screen. However, when the echo is turned on, they appear normally.

Some of the letters recognized by music mode must be followed by a single digit. Here is a table of these letters and their associated numbers:

Letter	Numbers	Value of numbers
V	Ø to 9	Ø is quietest, 9 loudest
S	1 to 4	Speaker no.
O	1 to 3	1 is lowet, 3 is highest
T	Ø to 9	Tempo

The tempo is selected according to this chart:

Tempo number	Beats per minute	Duration of note times 1/60 second
Ø	9ØØ	4
1	6ØØ	6
2	45Ø	8
3	3ØØ	12
4	225	16
5	15Ø	24
6	112.5	32
7	56.25	64
8	28.	128
9	14.	255

Each sound register has 3 complete octaves in its range. However, since register 2 is one octave higher than register 1, the three octaves are different. When playing notes in different sound registers, be sure that the octaves used are chosen correctly.

Here are some examples of strings that play music (the reversed f symbol means hold down the CTRL Key and type the left arrow key at the top left corner of the keyboard).

```
1Ø PRINT"ALL THE SAME NOTE[CTRL][←] T7 V9 S1 Ø3 C S2 C
S3 Ø1 C"
20 PRINT "SCALE [CTRL] [←] T3 V9 S2 Ø1 CDEFGAB Ø2
BDEFGAB Ø3 CDEFGAB"
3Ø PRINT "CHORDS[CTRL][←] T5 V9 S1 Ø3 C S2 E S3 Ø1 G";
4Ø PRINT "S1 Ø3 D S2 Ø2 #F S3 Ø1 B";
5Ø PRINT "S1 Ø3 E S2 Ø2 A S3 Ø2 C"
```

IV. PROGRAMMABLE FUNCTION KEYS

There are 8 functions that can be accessed with the function keys along with the SHIFT key. These are pre-assigned to the following:

Key	Text
f1	GRAPHIC
f2	COLOR
f3	DRAW
f4	SOUND
f5	CIRCLE
f6	RUN + RETURN
f7	POINT
f8	LIST + RETURN

Any time you hit these keys, the characters with which they are programmed enter the input buffer exactly as if you had typed them in yourself. Each key can be programmed with up to 128 characters, using the KEY command.

KEY
or
KEY n, "string"

The word KEY all by itself,makes the VIC display a list of all 8 function keys and their current value. This is formatted on the screen so the contents of the strings can be edited by moving the cursor up to the string, making the desired change, and hitting RETURN, just like editing a program. You can change any key in the KEY MENU by changing the information inside the quotation marks and hitting return or by typing the KEY command using the format shown in the example below.

When followed by a number, this command changes the string value of the key to the new value. This can be up to 128 characters, including cursor and color controls, the RETURN (CHR$(13)), or any other character. The keys can be changed under program control as well.

EXAMPLE: KEY 1, "ISN'T IT AMAZING?"

V. 3K MEMORY EXPANSION

When the 3K cartridge is used to expand the VIC-2Ø, the BASIC program area is changed so it starts at location 1Ø24 ($ØØ4Ø).

Note that with the 3K cartridge plugged in, only on-board memory (4Ø96-8191 or $1ØØØ-1FFF) can be used as VIC screen and hi-resolution graphics areas.

SUPEREXPANDER PROGRAM EXAMPLES

PROGRAM 1

```
5 REM TRIG PLOT DEMO
1Ø GRAPHIC2
11 SCNCLR
15 SC=INT(RND(1)*16):CH=INT(RND(1)*8):IFSC=CHTHEN15
2Ø COLORSC,SC,CH,1
3Ø AM=RND(1)*1ØØ+1ØØ
4Ø FR=RND(1)*1ØØ+5Ø
5Ø AD=RND(1)**2
6Ø TA=INT(RND(1)+.3)
7Ø IFTA=1THENAM=RND(1)*1ØØ+5Ø:FR=RND(1)*2ØØ+5Ø:CT=INT(RND(1)+.5)
8Ø X=Ø:GOSUB2ØØ
9Ø DRAW1,X,Y:YØ=Y
1ØØ FORX=1ØTO1Ø24STEP2Ø
11Ø GOSUB2ØØ
12Ø IFABS(Y-YØ)>6ØØTHEN:DRAW1,X,Y
13Ø DRAW1TOX,Y:YØ=Y
14Ø NEXT
15Ø FORX=1TO2ØØØ:NEXT
16Ø RUN
2ØØ Y=AM*SIN(X/FR+AD)/COS(TA*(X/FR+AD)):IFCT=1THENY=-Y
21Ø Y=512+Y:IFY<ØTHENY=Ø
22Ø IFY>1Ø23THENY=1Ø23
23Ø RETURN
```

PROGRAM 2

```
5 REM 3-D PYRAMID
10 GRAPHIC2:COLOR0,7,1,12
20 FOR I=1TO5:READX(I),Y(I),Z(I):NEXT
100 FOR I=1TO5
105 YY(I)=Y(I)
110 XX(I)=X(I)
111 ZZ(I)=Z(I)
120 POINT2,500+(200*XX(I))/(YY(I)+200),500
130 NEXT:XC=0:YC=220
160 A=A+20:R=A/57.29
165 FOR I=1TO5
170 XX(I)=(X(I)-XC)*COS(R)+(Y(I)-XC)*SIN(R)+XC
180 YY(I)=(Y(I)-YC)*COS(R)+(X(I)-XC)*SIN(R)+YC+100
185 AX(I)=(300*XX(I))/(YY(I)+300)+500:AY(I)=(300*ZZ(I))/(YY(I)+300)+500
190 NEXT
191 SCNCLR
195 FOR I=1TO4
200 DRAW2,AX(5),AY(5)TOAX(I),AY(I)
210 NEXT
220 DRAW2,AX(1),AY(1)TOAX(2),AY(2)
221 DRAW2,AX(3),AY(3)TOAX(4),AY(4)
222 DRAW2,AX(3),AY(3)TOAX(1),AY(1)
223 DRAW2,AX(2),AY(2)TOAX(4),AY(4)
999 GOTO160
1000 DATA-200,20,100,-200,420,100,200,20,100,200,420,100,0,220,-800
```

PROGRAM 3

```
5 REM GRAPHIC DEMO
6 REM KINETIC STRING ART ADAPTED FROM BYTE ARTICLE
10 GRAPHIC2
20 COLOR0,0,5,5
30 FORL=0TO500STEP40
40 CIRCLE2,511,511,L,500-L
50 NEXT
55 GOSUB900
60 Q=RND(1)/2+.25
70 FORL=0TO500STEP40
80 CIRCLE2,511,511,L,500-L*Q
90 NEXT
100 GOSUB900
110 Q=RND(1)/2+.25
120 FORL=0TO500STEP40
130 CIRCLE2,511,511,L*Q,500-L
140 NEXT
150 GOSUB900
220 FORL=0TO2*πSTEPπ/25
230 CIRCLE2,511+400*SIN(L),511+400*COS(L),75,100
240 NEXT
250 GOSUB900
260 FORL=0TO2*πSTEPπ/50
270 CIRCLE2,511+400*SIN(L),511+400*COS(L)/(L+1),75-L*10,100-L*13
280 NEXT
290 GOSUB900
300 GRAPHIC1:DIMA%(200,3):FORL=1TO5:SC=INT(RND(1)*8)
301 BO=INT(RND(1)*16):IFBO=SCTHEN301
302 CH=INT(RND(1)*8):IF(CH=SC)OR(CH=BO)THEN302
303 AU=INT(RND(1)*16):IF(AU=SC)OR(AU=BO)OR(AU=CH)THEN303
304 COLORSC,BO,CH,AU
310 X=INT(RND(1)*1024):Y=INT(RND(1)*1024):X1=INT(RND(1)*1024)
315 Y1=INT(RND(1)*1024)
320 C1=0:C2=0
330 FORM=0TO200
340 IFC1<1THENC1=5+INT(RND(1)*10):RE=INT(RND(1)*3)+1
350 IFC2<1THENDX=INT(RND(1)*81)-40:DY=INT(RND(1)*81)-40:DA=INT(RND(1)*81)-40
355 IFC2<1THENDB=INT(RND(1)*81)-40:C2=15+INT(RND(1)*10)
360 X=X+DX:Y=Y+DY:X1=X1+DA:Y1=Y1+DB
361 IF(X<0)OR(X>1023)THENDX=-DX:X=X+DX*2
362 IF(Y<0)OR(Y>1023)THENDY=-DY:Y=Y+DY*2
363 IF(X1<0)OR(X1>1023)THENDA=-DA:X1=X1+DA*2
364 IF(Y1<0)OR(Y1>1023)THENDB=-DB:Y1=Y1+DB*2
365 C1=C1-1:C2=C2-1
370 DRAW0,A%(M,0),A%(M,1)TOA%(M,2),A%(M,3)
380 DRAWRE,X,YTOX1,Y1:A%(M,0)=X:A%(M,1)=Y:A%(M,2)=X1:A%(M,3)=Y1
390 NEXT:NEXT
400 GOSUB900
899 GRAPHIC4:COLOR1,2,6,6:PRINT"HOPE YOU LIKED THE SHOW!":GOSUB900:RUN
900 FORD=1TO2000:NEXT:SCNCLR:RETURN
```