Appendix A/ BASICG/Utilities Reference Summary

Argument ranges are indicated below by special letters and words:

ar is a single-precision floating point number > Ø.Ø (to
1* 1Ø^{38}).

b	is an integer expression of either Ø or 1.
B	specifies a box.
BF	specifies a shaded box.
c	is an integer expression of Ø or 1.
n	is an integer expression from Ø to 2.
p	is an integer expression from Ø to 3.
r	is an integer expression from Ø to 639.
x	is an integer expression from Ø to 639.
y	is an integer expression from Ø to 239.
action	is either AND, PSET, PRESET, OR, or XOR.
background	is a string of either Ø or 1.
border	is an integer expression of either Ø or 1.
end	is an expression from -6.283185 to 6.283185.
start	is an expression from -6.283185 to 6.283185.
switch	is an integer expression of Ø or 1.
tiling	is a string or an integer expression of Ø or 1.
type	is an integer expression of Ø or 1.

CIRCLE(x,y)r,c,start,end,ar Draws a circle, ellipse, semicircle, arc, or point.

```
CIRCLE(1ØØ,1ØØ),25,1              CIRCLE(15Ø,15Ø),4Ø,1,,,.6
CIRCLE(1ØØ,1ØØ),1ØØ,PI,2*PI,5     CIRCLE(-5Ø,-5Ø),2ØØ
```

CLS Clears the Text Screen and video memory.

```
CLS            SYSTEM"CLS"
```

CLR Clears the Graphics Screen.

```
CLR
```

GCLS Clears the Graphics Screen and memory.

```
GCLS      CMD "I","GCLS"     1ØØ CMD "I","GCLS"
```

GET(x1,y1)-(x2,y2),array name Reads the contents of a rectangular pixel area into an array.

```
GET(1Ø,1Ø)-(5Ø,5Ø),V
```

GLOAD filename/ext.password:d Loads graphics memory.

```
GLOAD PROG    CMD "I","GLOAD PROG"
```

GLOCATE (x,y),direction Sets the Graphics Cursor

```
GLOCATE (32Ø,12Ø),Ø
```

GPRINT Dumps graphic display on the printer.

```
 GPRINT            CMD "I","GPRINT"    1ØØ CMD "I","GPRINT"
```

GPRT2 Dumps graphic display on the printer without rotating 9Ø degrees.

```
GPRT2              CMD"I","GPRT2"      1ØØ CMD"I","GPRT2"
```

GPRT3 Dumps graphic display on the printer without rotating 9Ø degrees.

```
GPRT3              CMD"I","GPRT3"      1ØØ CMD"I","GPRT3"
```

GROFF Turns Graphic Display OFF.

```
GROFF   CMD "I","GROFF"
```

GRON Turns Graphic Display ON.

```
GRON  CMD "I","GRON"
```

GSAVE filename/ext.password:d Saves graphics memory.

```
GSAVE PROG       CMD "I","GSAVE PROG"
```

LINE(x1,y1)-(x2,y2),c,B or BF, style Draws a line/box.

```
LINE -(1ØØ,1ØØ)          LINE(1ØØ,1ØØ)-(2ØØ,2ØØ),1,B,45
LINE(Ø,Ø)-(1ØØ,1ØØ),1,BF      LINE(-2ØØ,-2ØØ)-(1ØØ,1ØØ)
```

PAINT(x,y),tiling,border,background Paints the screen.

```
PAINT(32Ø,12Ø),1,1         PAINT(32Ø,12Ø),"DDDDD",1
PAINT(32Ø,12Ø),A$,1
PAINT(32Ø,12Ø),CHR$(Ø)+CHR$(&HFF),Ø,CHR$(&HØØ)
PAINT(32Ø,12Ø),CHR$(E)+CHR$(77)+CHR$(3)
```

&POINT(x,y) A function. Tests graphics point.

```
PRINT &POINT(32Ø,12Ø)      IF &POINT(32Ø,12Ø)=1 THEN . . .
PRINT &POINT(32Ø,12Ø),-1
```

PRESET(x,y),switch Sets pixel OFF or ON.

PRESET(1ØØ,1ØØ),Ø

PRINT #-3, item list Write text characters to the Graphics Screen.
PRINT #-3,"MONTHLY"

PSET(x,y),switch Sets pixel ON or OFF.
PSET(1ØØ,1ØØ),1

PUT(xl,yl),array name,action Puts graphics from an array onto the screen.
PUT(1ØØ,1ØØ),A,PSET PUT(1ØØ,1ØØ),A,AND
PUT(A,B),B

SCREEN type Selects the screen.
SCREEN Ø

VIEW(xl,yl)-(x2,y2),c,b Redefines the screen and creates a viewport.
VIEW(1ØØ,1ØØ)-(15Ø,15Ø) VIEW(1ØØ,1ØØ)-(15Ø,15Ø),Ø,1

&VIEW(p) A function. Returns viewport's coordinates.
PRINT &VIEW(1)

Appendix B/ BASICG Error Messages

Code	Abbreviation	Explanation
1	NF	NEXT without FOR. NEXT is used without a matching FOR statement. This error may also occur if NEXT variables are reversed in a nested loop.
2	SN	Syntax. This is usually the result of incorrect punctuation, an illegal character or a misspelled command.
3	RG	RETURN without GOSUB. A RETURN statement was executed with insufficient data available. The DATA statement may have been left out or all data may have been read.
4	OD	Out of data. A READ statement was executed with insufficient data available. The DATA statement may have been left out or all data may have been read.
5	FC	Illegal function call. An attempt was made to execute an operation using an illegal parameter. Graphic examples: PUTting a display that is partially off the Screen, GETting an array that is not properly dimensioned, or using more than two OFF tiles or two ON tiles in a string when tiling (with PAINT).
6	OV	Overflow. The magnitude of the number derived or input is too large for the data storage type assigned to it. The integer range is (-32768 to 32767) for BASICG.
7	OM	Out of memory. All available memory has been used or reserved. This may occur with large array dimensions and nested branches such as GOSUB and

		FOR/NEXT loops.
8	UL	Undefined line. An attempt was made to reference a non-existent line.
9	BS	Bad subscript. An attempt was made to assign an array element with a subscript beyond the dimensioned range.
1Ø	DD	Double-dimensioned array. An attempt was made to dimension an array which had previously been created with DIM or by default statements. ERASE must be used first.
11	/Ø	Division by zero. An attempt was made to use a value of zero in the denominator. Note: If you can't find an obvious division by zero, check for division by numbers smaller than allowable ranges (see OV).
12	ID	Illegal direct. An attempt was made to use a program only statement like INPUT in an immediate (non-program) line.
13	TM	Type mismatch. An attempt was made to assign a number to a string variable or a string to a numeric variable.
14	OS	Out of string space. The amount of string space allocated was exceeded. Use CLEAR to allocate more string space. 1ØØ bytes is the default string space allocation.
15	LS	Long string. A string variable was assigned a string which exceeded 255 characters in length.
16	ST	String too complex. A string operation was too complex to handle. The operation must be broken into shorter steps.
17	CN	Can't continue. A CONT command was given at a point where the command can't be carried out, e.g., directly after the program has been edited.

18	UF	Undefined user function. An attempt has been made to call a USR function without first defining its entry point via a DEFUSR statement.
19	NR	No RESUME. During an error-trapping routine, BASICG has reached the end of the program without encountering a RESUME.
2Ø	RW	RESUME without error. A RESUME was encountered when no error was present. You need to insert END or GOTO in front of the error-handling routine.
21	UE	Undefined error. Reserved for future use.
22	MO	Missing operand. An operation was attempted without providing one of the required operands.
23	BO	Buffer overflow. An attempt was made to input a data line which has too many characters to be held in the line buffer.
24	NB	Files not compatible. An attempt was made to load a BASIC file (in compressed format) into BASICG.
25-49	UE	Undefined error. Reserved for future use.
5Ø	FO	Field overflow. An attempt was made to have more characters than the direct-access file record length allows. The record length is assigned when the file is first opened. The default length is 256.
51	IE	Internal error. Also indicates an attempt to use EOF on a file which is not open.

52	BN	Bad file number. An attempt was made to use a file number which specifies a file that is not open or that is greater than the number of files specified when BASICG was started up.
53	FF	File not found. Reference was made in a LOAD, KILL or OPEN statement to a file which did not exist on the diskette specified.
54	BM	Bad file mode. Program attempted to perform direct access on a file opened for sequential access or vice-versa.
55	AO	File already open. An attempt was made to open a file that was already open. This error is also output if KILL, LOAD, SAVE, etc., is given for an open file.
56	IO	Disk I/O error. An error has been detected during a disk access.
57	UE	Undefined in Model III BASIC.
58	UE	Undefined error. Reserved for future use.
59	DF	Diskette full. All storage space on the diskette has been used. KILL unneeded files or use a formatted diskette which has available space.
6Ø	EF	End of file. An attempt was made to read past the end of file.
61	RN	Bad record number. In a PUT or GET statement, the record number is either greater than the allowable maximum, equal to zero, or negative.
62	NM	Bad file name.
63	MM	Mode mismatch. A sequential OPEN was executed for a file that already existed on the diskette as a direct access file, or vice versa.

64	UE	Undefined error. Reserved for future use.
65	DS	Direct statement. A direct statement was encountered during a load of a program in ASCII format. The load is terminated.
66	FL	Too many files.

Appendix C/ Subroutine Language Reference Summary

CIRCLE (radius,color,start,end,ar) Draws a circle, ellipse, semicircle, arc, or point. (x,y) coordinates set by SETXY.

```
CALL CIRCLE(1ØØ,1,Ø,Ø,Ø)
```

CLS Clears the Graphics Screen.

```
CALL CLS
```

FVIEW (n) Returns viewport parameter.

```
I=FVIEW(Ø)
```

GET (array,size) Reads the contents of a rectangular pixel area into an array for future use by PUT.

```
CALL GET(A,4ØØØ)
```

GPRINT (size,array) Displays textual data.

```
CALL GPRINT (28,ARRAY1)
```

GRPINI(option) Graphics initialization routine.

```
CALL GRPINI(Ø)
```

LINE (color,style) Draws a line. Coordinates set by SETXY or SETXYR.

```
CALL LINE (1,-1)
```

LINEB (color,style) Draws a box. Coordinates set by SETXY or SETXYR.

```
CALL LINEB (1,-1)
```

LINEBF (color) Draws a filled box. Coordinates set by SETXY or SETXYR.

```
CALL LINEBF (1)
```

LOCATE (n) Sets the direction for displaying textual data.

```
CALL LOCATE (Ø)
```

PAINT (color,border) Paints the screen.

```
CALL PAINT(1,1)
```

PAINTT (arrayT,border,arrayS) Paints the screen with defined paint style.

```
    CALL PAINTT (A,1,V)
```

POINT Returns the pixel value at current coordinates.

```
    K=POINT(M)
```

PRESET (color) Sets the pixel ON or OFF.

```
    CALL PRESET(Ø)
```

PSET (color) Sets the pixel ON or OFF.

```
    CALL PSET(Ø)
```

SCREEN (type) Sets the screen.

```
    CALL SCREEN(1)
```

SETXY(X,Y) Sets the coordinates (absolute).

```
    CALL SETXY(1ØØ,1ØØ)
```

SETXYR(X,Y) Sets the coordinates (relative).

```
    CALL SETXYR(5Ø,5Ø)
```

VIEW(leftX,leftY,rightX,rightY,color,border)
Sets the viewport.

```
    CALL VIEW(1ØØ,1ØØ,2ØØ,2ØØ,Ø,1)
```

Appendix D/ Sample Programs

BASICG

```
1Ø '
2Ø ' Pie Graph Program ("PECANPIE/GRA")
3Ø '
4Ø ' Object
5Ø '   The object of this program is to draw a pie graph of the
6Ø '   expenses for a given month of eight departments of a company,
7Ø '   along with the numerical value of each pie section
8Ø '   representation.
9Ø '
1ØØ '
11Ø ' Running the program
12Ø '  The month and the amounts spent by each department are input,
13Ø '  and the program takes over from there.
14Ø '
15Ø ' Special features
16Ø '    The amounts spent by each account as well as the total
17Ø '    amount spent are stored in strings.  The program will
18Ø '    standardize each string so that it is 9 characters long
19Ø '    and includes two characters to the right of the decimal
2ØØ '    point.  This allows for input of variable length and an
21Ø '    optional decimal point.
22Ø '
23Ø '    The various coordinates used in the program are found
24Ø '    based on the following equations:
25Ø '
26Ø '     x = r * cos(theta)
27Ø '     y = r * sin(theta)
28Ø '
29Ø '    where x and y are the coordinates, r is the radius, and theta
3ØØ '    is the angle.  (Note:  The y-coordinates are always multiplied
31Ø '    by Ø.5.  This is because the y pixels are twice the size of the
32Ø '    x pixels.)
33Ø '
34Ø '    If an angle theta is generated by a percent less than 1%, the
35Ø '    section is not graphed, and the next theta is calculated.
36Ø '    However, the number will still be listed under the key.
37Ø '
38Ø ' Variables
39Ø '    ACCT$(i)   Description of the account
4ØØ '    BUD$(i)    Amount spent by the account
```

```
41Ø '     DS$            Dollar sign (used in output)
42Ø '     HXCOL          Column number for the pie section number
43Ø '     HYRW           Row number for the pie section number
44Ø '     I              Counter
45Ø '     MN$            Month
46Ø '     PER(i)         Percent value of BUD$(i)
47Ø '     R              Radius of circle
48Ø '     TØ             Angle value line to be drawn
49Ø '     Tl             Angle value of the next line
5ØØ '     TBUD$          Total of all the BUD$(i)'s
51Ø '     THALF          Angle halfway between Tl and TØ (used for
52Ø '                    location position for section number)
53Ø '     TILE$(i)       Paint style for each section
54Ø '     TWOPI          Two times the value of pi
55Ø '     XØ             X-coordinate for drawing the line represented
56Ø '                    by TØ
57Ø '     XP             X-coordinate for painting a section
58Ø '     YØ             Y-coordinate for drawing the line represented
59Ø '                    by TØ
6ØØ '     YP             Y-coordinate for painting a section
61Ø '
62Ø ' Set initial values
63Ø '
64Ø CLEAR 1ØØØ
65Ø DIM THALF(15),BUD$(15),ACCT$(15),PER(16)
66Ø TWOPI=2*3.14159
67Ø R=18Ø
68Ø DS$="$"
69Ø ACCT$(1) = "Sales"
7ØØ ACCT$(2) = "Purchasing"
71Ø ACCT$(3) = "R&D        "
72Ø ACCT$(4) = "Accounting"
74Ø ACCT$(5) = "Advertising "
75Ø ACCT$(6) = "Utilities  "
76Ø ACCT$(7) = "Security "
77Ø ACCT$(8) = "Expansion"
78Ø TILE$(Ø)=CHR$(&H22)+CHR$(&HØØ)
79Ø TILE$(1)=CHR$(&HFF)+CHR$(&HØØ)
8ØØ TILE$(2)=CHR$(&H99)+CHR$(&H66)
81Ø TILE$(3)=CHR$(&H99)
82Ø TILE$(4)=CHR$(&HFF)
83Ø TILE$(5)=CHR$(&HFØ)+CHR$(&HFØ)+CHR$(&HØF)+CHR$(&HØF)
84Ø TILE$(6)=CHR$(&H3C)+CHR$(&H3C)+CHR$(&HFF)
85Ø TILE$(7)=CHR$(&HØ3)+CHR$(&HØC)+CHR$(&H3Ø)+CHR$(&HCØ)
86Ø '
87Ø ' Enter values to be graphed, standardize them, and calculate
88Ø ' the percent they represent
89Ø '
```

```
9ØØ CLR
91Ø CLS
92Ø SCREEN1
93Ø PRINT @64,"Enter month "
94Ø PRINT @192,"Enter amount spent by"
95Ø PRINT @256,"$"
96Ø PRINT @Ø,""
97Ø LINE INPUT "Enter month ";MN$
98Ø FOR I=1 TO 8
99Ø PRINT @214,ACCT$(I);"
1ØØØ PRINT @256,"$"
1Ø1Ø PRINT @192,""
1Ø2Ø LINE INPUT "$";BUD$(I)
1Ø3Ø IF INSTR(BUD$(I),".") = Ø THEN BUD$(I)=BUD$(I)+".ØØ"
1Ø4Ø IF LEN(BUD$(I))<9 THEN BUD$(I)=" "+BUD$(I):GOTO 1Ø4Ø
1Ø5Ø TBUD$=STR$(VAL(TBUD$)+VAL(BUD$(I)))
1Ø6Ø NEXT I
1Ø7Ø IF INSTR(TBUD$,".")=Ø THEN TBUD$=TBUD$+".ØØ"
1Ø8Ø IF LEN(TBUD$)<9 THEN TBUD$=" "+TBUD$:GOTO 1Ø8Ø
1Ø9Ø FOR I=1 TO 8
11ØØ PER(I)=VAL(BUD$(I))/VAL(TBUD$)*1ØØ
111Ø NEXT I
112Ø SCREENØ
113Ø '
114Ø ' Draw the circle and calculate the location of the lines and
115Ø ' the line numbers
116Ø '
117Ø CIRCLE(41Ø,12Ø),R
118Ø FOR I=Ø TO 8
119Ø TØ=TWOPI/1ØØ*PER(I)+TØ
12ØØ XØ=41Ø+R*COS(TØ)
121Ø YØ=12Ø-R*SIN(TØ)*Ø.5
122Ø T1=TWOPI/1ØØ*PER(I+1)+TØ
123Ø THALF(I)=(TØ+T1)/2
124Ø HXCOL=(41Ø+R*1.15*COS(THALF(I)))
125Ø HYRW=(12Ø-R*1.15*SIN(THALF(I))*Ø.5)
126Ø IF PER(I)>1 THEN LINE (41Ø,12Ø)-(XØ,YØ)
127Ø GLOCATE (HXCOL,HYRW),Ø
128Ø IF I<8 AND PER(I+1)>1 THEN PRINT #-3,I+1
129Ø NEXT I
13ØØ '
131Ø ' Paint the appropriate sections of the pie
132Ø '
133Ø FOR I=Ø TO 7
134Ø XP=41Ø+R*Ø.5*COS(THALF(I))
135Ø YP=12Ø-R*Ø.5*SIN(THALF(I))*Ø.5
136Ø IF PER(I+1)<=1 THEN 138Ø
137Ø PAINT (XP,YP),TILE$(I),1
138Ø NEXT I
```

```
139Ø '
14ØØ ' Print the key for the graph
141Ø '
142Ø GLOCATE(Ø,1Ø),Ø
143Ø PRINT #-3,"Expenditures for"
144Ø GLOCATE(Ø,25),Ø
145Ø PRINT #-3,MN$
146Ø GLOCATE(Ø,4Ø),Ø
147Ø PRINT #-3,"#    Description   Amount"
148Ø FOR I=1 TO 8
149Ø GLOCATE(Ø,(4+I)*15),Ø
15ØØ PRINT #-3,I
151Ø GLOCATE(4Ø,(4+I)*15),Ø
152Ø PRINT #-3,ACCT$(I)
153Ø GLOCATE(13Ø,(I+4)*15),Ø
154Ø PRINT #-3,DS$;BUD$(I)
155Ø DS$=" "
156Ø NEXT I
157Ø GLOCATE(Ø,195),Ø
158Ø PRINT #-3,STRING$(26,"-")
159Ø GLOCATE(4Ø,21Ø),Ø
16ØØ PRINT #-3,"Total       ";TBUD$
161Ø FOR I=1 TO 1ØØØØ
162Ø NEXT I
163Ø SCREEN1
164Ø END
```

```
1Ø ' "THREEDEE/GRA"
2Ø '
3Ø ' Object
4Ø '    The object of this program is to produce a three
5Ø ' dimensional bar graph representation of the gross
6Ø ' income for a company over a one year period.
7Ø '
8Ø ' Variables
9Ø '   A  Vertical alphanumeric character
1ØØ '  BMSG$ Bottom message
11Ø '  CHAR$ Disk file input field
12Ø '  GI$  Gross income
13Ø '  I  Counter
14Ø '  J  Counter
15Ø '  MN$  Month
16Ø '  REC  Record number of vertical character
17Ø '  S1$  Single character of vertical message
18Ø '  TILE$ Tile pattern for painting
19Ø '  TTINC Total income for the year
2ØØ '  X  X-coordinate of bar
21Ø '  Y(i) Y-coordinate of bar
22Ø '
23Ø 'Input/output
24Ø '  The program prompts you to enter the gross income, in millions.
25Ø 'for each month.  The program requires these values to be between one
26Ø 'and nine.
27Ø '
28Ø 'Set initial values
29Ø '
3ØØ CLS
31Ø DIM Y(12),A(8),MN$(12)
32Ø DEFINT A
33Ø VMSG$=" Millions of dollars "
34Ø TMSG$="G r o s s  I n c o m e  F o r  1 9 8 Ø "
35Ø BMSG$="M o n t h"
36Ø MN$(1)="January"
37Ø MN$(2)="February"
38Ø MN$(3)="March"
39Ø MN$(4)="April"
4ØØ MN$(5)="May"
41Ø MN$(6)="June"
42Ø MN$(7)="July"
43Ø MN$(8)="August"
44Ø MN$(9)="September"
45Ø MN$(1Ø)="October"
46Ø MN$(11)="November"
47Ø MN$(12)="December"
48Ø TILE$=CHR$(&H99)+CHR$(&H66)
49Ø X=-1Ø
```

```
5ØØ '
51Ø 'Input gross income, and calculate the Y-coordinate
52Ø '
53Ø FOR I=1 TO 12
54Ø CLS
55Ø PRINT "Enter gross income in millions (1-9) for ";MN$(I)
56Ø LINE INPUT "$";GI$
57Ø Y(I)=2Ø5-2Ø*VAL(GI$)
58Ø TTINC=TTINC+VAL(GI$)
59Ø NEXT I
6ØØ CLR
61Ø SCREENØ
62Ø '
63Ø 'Draw the graph and bars
64Ø '
65Ø FOR I=1 TO 12
66Ø CLS
67Ø X=X+5Ø
68Ø LINE (X,Y(I))-(X+2Ø,2Ø5),1,BF
69Ø LINE -(X+4Ø,195)
7ØØ LINE -(X+4Ø,Y(I)-1Ø)
71Ø LINE -(X+2Ø,Y(I)-1Ø)
72Ø LINE -(X,Y(I))
73Ø LINE (X+2Ø,Y(I))-(X+4Ø,Y(I)-1Ø)
74Ø PAINT(X+21,Y(I)+2),TILE$,1
75Ø NEXT I
76Ø GLOCATE(4Ø,215),Ø
77Ø PRINT #-3,"Jan   Feb   Mar    Apr   May   June  July   Aug    Sept  Oct
Nov   Dec"
78Ø GLOCATE(29Ø,23Ø),Ø
79Ø PRINT #-3,BMSG$
8ØØ FOR I=1 TO 1Ø
81Ø IF I>9 THEN C=1 ELSE C=2
82Ø GLOCATE((C*1Ø)-5,(2Ø-I*2)*1Ø),Ø
83Ø PRINT #-3,STR$(I);"-"
84Ø NEXT I
85Ø LINE (35,Ø)-(35,2Ø5)
86Ø LINE -(639,2Ø5)
87Ø GLOCATE(Ø,18Ø),3
88Ø PRINT #-3,VMSG$
89Ø GLOCATE(22Ø,Ø),Ø
9ØØ PRINT #-3,TMSG$
91Ø GLOCATE(26Ø,1Ø),Ø
92Ø PRINT #-3,"(Total income is";TTINC;" million)"
93Ø FOR I=1 TO 1ØØØØ
94Ø NEXT I
95Ø SCREEN1
96Ø END
```

Printing Graphics Displays

There are many ways to use the stand-alone utilities (described in Graphic Utilities). The following discussion demonstrates one way to use the utilities with graphic displays generated under BASICG.

To print graphics, follow these steps:

1. When TRSDOS Ready appears, set FORMS to FORMS (WIDTH=255, LINES=6Ø). (See your **Model III Disk System Owner's Manual.**)
2. Set the printer into Graphic Mode, if possible, and set the printer's other parameters (elongation, non-elongated, etc.), if applicable, according to instructions in your printer owner's manual.
3. Write, run and save your program as a BASICG program file.
4. Save the graphics memory to diskette using GSAVE.
5. Load the file into memory using GLOAD.
6. Enter the print command GPRINT.

Example

1. Set FORMS with your printer's printing parameters.

2. Load BASICG and type in this program:

```
   5 SCREEN Ø
  1Ø DEFDBL Y
  2Ø CLR
  3Ø LINE (Ø,12Ø)-(64Ø,12Ø)
  4Ø LINE (32Ø,Ø)-(32Ø,24Ø)
  5Ø FOR X=Ø TO 64Ø
  6Ø PI=3.141259
  7Ø X1=X/64Ø*2*PI-PI
  8Ø Y=SIN(X1)*1ØØ
  9Ø IF Y>1ØØ THEN X=X+7
 1ØØ PSET (X,-Y+12Ø)
 11Ø NEXT X
 12Ø GLOCATE(Ø,Ø),Ø
 13Ø PRINT #-3,"THIS IS A SINE WAVE."
```

3. RUN the program.

   The program draws a sine wave on the Graphics Screen (graphics memory) and prints the statement in line 13Ø ("THIS IS A SINE WAVE.") on the Graphics Screen.

4. SINE (for sine wave) is the name we are giving this TRSDOS file. To save the contents of the graphics memory (which now includes the converted video memory) to diskette, type: CMD"I","GSAVE SINE" <ENTER>.

5. The graphics memory is saved as a TRSDOS file on your diskette and you will return to TRSDOS Ready.

6. Type: GCLS <ENTER>

   The graphics memory is now cleared.

7. To load the file back into memory, type:

   GLOAD SINE <ENTER>

   The display is now on the Graphics Screen.

8. To print, type: GPRINT <ENTER>.

FORTRAN Sample Programs

```
ØØ1ØØ   C       HIGH RESOLUTION GRAPHICS TEST - MAIN PROGRAM
ØØ2ØØ   C
ØØ3ØØ           CALL GRPINI(Ø)
ØØ4ØØ           CALL SCREEN(Ø)
ØØ5ØØ   C
ØØ6ØØ   C       CIRCLE TEST
ØØ7ØØ   C
ØØ8ØØ           CALL CTEST
ØØ9ØØ   C
Ø1ØØØ   C       LINE TEST
Ø11ØØ   C
Ø12ØØ           CALL LTEST
Ø13ØØ   C
Ø14ØØ   C       LINEB TEST
Ø15ØØ   C
Ø16ØØ           CALL LBTST
Ø17ØØ   C
Ø18ØØ   C       LINEBF TEST
Ø19ØØ   C
Ø2ØØØ           CALL LBFTST
Ø21ØØ   C
Ø22ØØ   C       PAINTT TEST
Ø23ØØ   C
Ø24ØØ           CALL PTTTST
Ø25ØØ   C
Ø26ØØ   C       GET AND PUT TEST
Ø27ØØ   C
Ø28ØØ           CALL GPTST
Ø29ØØ   C
Ø3ØØØ   C       PSET/POINT TEST
Ø31ØØ   C
Ø32ØØ           CALL PPTST
Ø33ØØ   C
Ø34ØØ   C       PRESET/POINT TEST
Ø35ØØ   C
Ø36ØØ           CALL PRETST
Ø37ØØ   C
Ø38ØØ   C       SCREEN TEST
Ø39ØØ   C
Ø4ØØØ           CALL SCRTST
Ø41ØØ   C
Ø42ØØ   C       VIEW/FVIEW TEST
Ø43ØØ   C
Ø44ØØ           CALL VTEST
```

```
Ø45ØØ           CALL CLS(2)
Ø46ØØ           END
```

```
ØØ1ØØ           SUBROUTINE CTEST
ØØ2ØØ   C
ØØ3ØØ   C       THIS SUBROUTINE TESTS CIRCLE, SETXY, AND PAINT
ØØ4ØØ   C
ØØ5ØØ           LOGICAL MSG(29)
ØØ6ØØ           CALL CLS
ØØ7ØØ           ENCODE(MSG,1ØØ)
ØØ8ØØ   1ØØ     FORMAT('TEST CIRCLE, SETXY, AND PAINT')
ØØ9ØØ           CALL SETXY(Ø,Ø)
Ø1ØØØ           CALL LOCATE(Ø)
Ø11ØØ           CALL GPRINT(29,MSG)
Ø12ØØ           CALL WAIT
Ø13ØØ           CALL VIEW(Ø,3Ø,639,239,Ø,Ø)
Ø14ØØ           DO 1Ø I=1,1ØØ
Ø15ØØ           IX=MOD(I*17,64Ø)
Ø16ØØ           IY=MOD(I*13,21Ø)
Ø17ØØ           IR=I*1.5
Ø18ØØ           START=MOD(I,13)-6.Ø
Ø19ØØ           END=MOD(I*3,13)-6.Ø
Ø2ØØØ           IF (START.LT.END) GOTO 1
Ø21ØØ           T=START
Ø22ØØ           START=END
Ø23ØØ           END=T
Ø24ØØ   1       CONTINUE
Ø25ØØ           RATIO=MOD(I*3,1ØØ)
Ø26ØØ           IF (RATIO.GT.Ø) RATIO=RATIO/4Ø.
Ø27ØØ           CALL SETXY(IX,IY)
Ø28ØØ           CALL CIRCLE(IR,1,START,END,RATIO)
Ø29ØØ   1Ø      CONTINUE
Ø3ØØØ   C
Ø31ØØ   C       RANDOMLY FILL IN THE AREAS
Ø32ØØ   C
Ø33ØØ           DO 11 I=1,5Ø
Ø34ØØ           IX=MOD(I*23,64Ø)
Ø35ØØ           IY=MOD(I*11,21Ø)
Ø36ØØ           CALL SETXY(IX,IY)
Ø37ØØ           CALL PAINT(1,1)
Ø38ØØ   11      CONTINUE
Ø39ØØ           CALL WAIT
Ø4ØØØ           CALL VIEW(Ø,Ø,639,239,-1,-1)
Ø41ØØ           RETURN
Ø42ØØ           END
```

```
              SUBROUTINE LTEST
C
C             THIS ROUTINE EXERCISES LINE
C
              LOGICAL MSG(19)
              CALL CLS(Ø)
              ENCODE(MSG,1ØØ)
1ØØ           FORMAT('LINE AND PAINT TEST')
              CALL SETXY(Ø,Ø)
              CALL LOCATE(Ø)
              CALL GPRINT(19,MSG)
              CALL WAIT
              J=1ØØ
              DO 1Ø I=1,639,2
              CALL SETXY(I,15)
              CALL SETXY(I,239)
              CALL LINE(1,J)
              J=J-1
1Ø            CONTINUE
              CALL WAIT
              CALL VIEW(Ø,15,639,239,Ø,Ø)
              CALL CLS
C
C             DRAW WHITE LINES AND FILL IN RANDOMLY
C
              IX=MOD(I*19,639)
              IY=MOD(I*17,223)
              CALL SETXY(IX,IY)
              DO 11 I=1,1ØØ
              IX=MOD(I*23,639)
              IY=MOD(I*29,223)
              CALL SETXY(IX,IY)
              CALL LINE(1,-1)
11            CONTINUE
              DO 12 I=1,5Ø
              IX=MOD(I*31,639)
              IY=MOD(I*37,223)
              CALL SETXY(IX,IY)
              CALL PAINT(1,1)
12            CONTINUE
              CALL WAIT
C
C     WHITE OUT SCREEN, DRAW BLACK LINES, PAINT BLACK RANDOMLY
C
              CALL VIEW(Ø,15,639,239,1,1)
              DO 15 I=1,1ØØ
              IX=MOD(I*11,639)
              IY=MOD(I*13,223)
              CALL SETXY(IX,IY)
```

```
Ø5ØØØ          CALL LINE(Ø,-1)
Ø51ØØ    15    CONTINUE
Ø52ØØ          DO 16 I=1,5Ø
Ø53ØØ          IX=MOD(I*17,639)
Ø54ØØ          IY=MOD(I*19,223)
Ø55ØØ          CALL SETXY(IX,IY)
Ø56ØØ          CALL PAINT(Ø,Ø)
Ø57ØØ    16    CONTINUE
Ø58ØØ          CALL WAIT
Ø59ØØ          CALL VIEW(Ø,Ø,639,239,Ø,Ø)
Ø6ØØØ          RETURN
Ø61ØØ          END
```

```
ØØ1ØØ          SUBROUTINE LBFTST
ØØ2ØØ    C
ØØ3ØØ    C     LINEBF TEST
ØØ4ØØ    C
ØØ5ØØ          LOGICAL MSG(11)
ØØ6ØØ          CALL CLS
ØØ7ØØ          ENCODE(MSG,1ØØ)
ØØ8ØØ    1ØØ   FORMAT('LINEBF TEST')
ØØ9ØØ          CALL SETXY(Ø,Ø)
Ø1ØØØ          CALL LOCATE(Ø)
Ø11ØØ          CALL GPRINT(11,MSG)
Ø12ØØ          CALL WAIT
Ø13ØØ          IXP=639
Ø14ØØ          ICLR=1
Ø15ØØ          DO 1Ø IX=Ø,12Ø
Ø16ØØ          CALL SETXY(IX*2,IX+3Ø)
Ø17ØØ          CALL SETXY(IXP,IXP-4ØØ)
Ø18ØØ          CALL LINEBF(ICLR)
Ø19ØØ          IXP=IXP-3
Ø2ØØØ          ICLR=ICLR-1
Ø21ØØ          IF (ICLR.LT.Ø) ICLR=1
Ø22ØØ    1Ø    CONTINUE
Ø23ØØ          CALL WAIT
Ø24ØØ          RETURN
Ø25ØØ          END
```

```
ØØ1ØØ          SUBROUTINE PTTTST
ØØ2ØØ    C
ØØ3ØØ    C     PAINT WITH TILES TEST
ØØ4ØØ    C
ØØ5ØØ          LOGICAL A(65),B(4),IS(16),MSG(23)
ØØ6ØØ          DATA A(1)/8/
ØØ7ØØ    C     X
ØØ8ØØ          DATA A(2),A(3),A(4),A(5)/X'41',X'22',X'14',X'Ø8'/
ØØ9ØØ          DATA A(6),A(7),A(8),A(9)/X'14',X'22',X'41',X'ØØ'/
Ø1ØØØ    C     FINE HORIZONTAL LINES
Ø11ØØ          DATA A(1Ø),A(11),A(12)/2,X'FF',X'ØØ'/
Ø12ØØ    C     MEDIUM HORIZONTAL LINES
Ø13ØØ          DATA A(13)/4/
Ø14ØØ          DATA A(14),A(15),A(16),A(17)/X'FF',X'FF',X'ØØ',X'ØØ'/
Ø15ØØ    C     DIAGONAL LINES
Ø16ØØ          DATA A(18)/4/
Ø17ØØ          DATA A(19),A(2Ø),A(21),A(22)/X'Ø3',X'ØC',X'3Ø',X'CØ'/
Ø18ØØ    C     LEFT TO RIGHT DIAGONALS
Ø19ØØ          DATA A(23)/4/
Ø2ØØØ          DATA A(24),A(25),A(26),A(27)/X'CØ',X'3Ø',X'ØC',X'Ø3'/
Ø21ØØ    C     FINE VERTICAL LINES
Ø22ØØ          DATA A(28),A(29)/1,X'AA'/
Ø23ØØ    C     MEDIUM VERTICAL LINES
Ø24ØØ          DATA A(3Ø),A(31)/1,X'CC'/
Ø25ØØ    C     COARSE VERTICAL LINES
Ø26ØØ          DATA A(32),A(33)/1,X'FØ'/
Ø27ØØ    C     ONE PIXEL DOTS
Ø28ØØ          DATA A(34),A(35),A(36)/2,X'22',X'ØØ'/
Ø29ØØ    C     TWO PIXEL DOTS
Ø3ØØØ          DATA A(37),A(38),A(39)/2,X'99',X'66'/
Ø31ØØ    C     PLUSES
Ø32ØØ          DATA A(4Ø),A(41),A(42),A(43)/3,X'3C',X'3C',X'FF'/
Ø33ØØ    C     SOLID
Ø34ØØ          DATA A(44),A(45)/1,X'FF'/
Ø35ØØ    C     BROAD CROSS HATCH
Ø36ØØ          DATA A(46),A(47),A(48),A(49)/3,X'92',X'92',X'FF'/
Ø37ØØ    C     THICK CROSS HATCH
Ø38ØØ          DATA A(5Ø)/4/
Ø39ØØ          DATA A(51),A(52),A(53),A(54)/X'FF',X'FF',X'DB',X'DB'/
Ø4ØØØ    C     FINE CROSS HATCH
Ø41ØØ          DATA A(54),A(55),A(56)/2,X'92',X'FF'/
Ø42ØØ    C     ALTERNATING PIXELS
Ø43ØØ          DATA A(57),A(58),A(59)/2,X'55',X'AA'/
Ø44ØØ          DATA B(1),B(2),B(3),B(4)/1,Ø,1,X'FF'/
Ø45ØØ          DATA IS(1),IS(2),IS(3),IS(4),IS(5),IS(6)/1,1Ø,13,18,23,28/
Ø46ØØ          DATA IS(7),IS(8),IS(9),IS(1Ø),IS(11)/3Ø,32,34,37,4Ø/
Ø47ØØ          DATA IS(12),IS(13),IS(14),IS(15),IS(16)/44,46,5Ø,54,57/
Ø48ØØ          CALL CLS
Ø49ØØ          ENCODE(MSG,1ØØ)
```

```
Ø5ØØØ   1ØØ     FORMAT('PAINTT AND SETXYR TESTS')
Ø51ØØ           CALL SETXY(Ø,Ø)
Ø52ØØ           CALL LOCATE(Ø)
Ø53ØØ           CALL GPRINT(23,MSG)
Ø54ØØ           CALL WAIT
Ø55ØØ   C
Ø56ØØ   C       PAINT ON A BLACK BACKGROUND
Ø57ØØ   C
Ø58ØØ           DO 1Ø I=1,16
Ø59ØØ           CALL SETXY(Ø,4Ø)
Ø6ØØØ           CALL SETXYR(639,199)
Ø61ØØ           CALL LINEB(1,-1)
Ø62ØØ           CALL SETXYR(-3ØØ,-1ØØ)
Ø63ØØ           ITMP=IS(I)
Ø64ØØ           CALL PAINTT(A(ITMP),1,B)
Ø65ØØ           CALL WAIT
Ø66ØØ           CALL VIEW(Ø,4Ø,639,239,Ø,Ø)
Ø67ØØ           CALL VIEW(Ø,Ø,639,239,-1,-1)
Ø68ØØ   1Ø      CONTINUE
Ø69ØØ   C
Ø7ØØØ   C       PAINT ON A WHITE BACKGROUND
Ø71ØØ   C
Ø72ØØ           DO 11 I=1,16
Ø73ØØ           IF(I.EQ.12) GOTO 11
Ø74ØØ           CALL VIEW(Ø,4Ø,639,239,Ø,Ø)
Ø75ØØ           CALL VIEW(Ø,Ø,639,239,-1,-1)
Ø76ØØ           CALL SETXY(Ø,4Ø)
Ø77ØØ           CALL SETXYR(639,199)
Ø78ØØ           CALL LINEBF(1)
Ø79ØØ           CALL SETXYR(-3ØØ,-1ØØ)
Ø8ØØØ           ITMP=IS(I)
Ø81ØØ           CALL PAINTT(A(ITMP),Ø,B(3))
Ø82ØØ           CALL WAIT
Ø83ØØ   11      CONTINUE
Ø84ØØ           RETURN
Ø85ØØ           END
```

```
ØØ1ØØ           SUBROUTINE GPTST
ØØ2ØØ     C
ØØ3ØØ     C     GET AND PUT TEST
ØØ4ØØ     C
ØØ5ØØ           LOGICAL A(1ØØØ),MSG(16)
ØØ6ØØ           CALL CLS
ØØ7ØØ           ENCODE(MSG,1ØØ)
ØØ8ØØ     1ØØ   FORMAT('GET AND PUT TEST')
ØØ9ØØ           CALL SETXY(Ø,Ø)
Ø1ØØØ           CALL LOCATE(Ø)
Ø11ØØ           CALL GPRINT(16,MSG)
Ø12ØØ           CALL VIEW(Ø,3Ø,639,239,Ø,Ø)
Ø13ØØ           CALL SETXY(1ØØ,1ØØ)
Ø14ØØ           CALL SETXYR(3Ø,3Ø)
Ø15ØØ           CALL LINEBF(1)
Ø16ØØ           CALL GET(A,1ØØØ)
Ø17ØØ           CALL CLS
Ø18ØØ           CALL WAIT
Ø19ØØ           CALL SETXY(1ØØ,1ØØ)
Ø2ØØØ           CALL PUT(A,1)
Ø21ØØ           CALL WAIT
Ø22ØØ           CALL VIEW(Ø,Ø,639,239,Ø,-1)
Ø23ØØ           RETURN
Ø24ØØ           END
```

```
          SUBROUTINE PPTST
C
C         PSET AND POINT TEST
C
          LOGICAL POINT,MSG(21)
          CALL CLS
          ENCODE(MSG,1ØØ)
1ØØ       FORMAT('PSET AND POINT TEST')
          CALL SETXY(Ø,Ø)
          CALL LOCATE(Ø)
          CALL GPRINT(19,MSG)
          CALL WAIT
          CALL CLS
C
C         SET AND CHECK ALL PIXELS
C
          DO 1Ø I=Ø,639
          DO 11 J=Ø,239
          CALL SETXY(I,J)
          CALL PSET(1)
          K=POINT(L)
          IF(K.EQ.Ø) GOTO 999
11        CONTINUE
1Ø        CONTINUE
C
C         RESET AND CHECK ALL PIXELS
C
          DO 12 I=Ø,639
          DO 13 J=Ø,239
          CALL SETXY(I,J)
          CALL PSET(Ø)
          K=POINT(L)
          IF (K.EQ.1) GOTO 999
13        CONTINUE
12        CONTINUE
          CALL CLS
          ENCODE(MSG,1Ø1)
1Ø1       FORMAT('PSET AND POINT PASSED')
          CALL SETXY(Ø,Ø)
          CALL LOCATE(Ø)
          CALL GPRINT(21,MSG)
          GOTO 1ØØØ
999       CALL CLS
          ENCODE(MSG,1Ø2)
1Ø2       FORMAT('PSET AND POINT FAILED')
          CALL SETXY(Ø,Ø)
          CALL LOCATE(Ø)
          CALL GPRINT(21,MSG)
1ØØØ      CALL WAIT
```

```
Ø5ØØØ             RETURN
Ø51ØØ             END
```

```
ØØ1ØØ          SUBROUTINE PRETST
ØØ2ØØ   C
ØØ3ØØ   C      PRESET AND POINT TEST
ØØ4ØØ   C
ØØ5ØØ          LOGICAL POINT,MSG(23)
ØØ6ØØ          CALL CLS
ØØ7ØØ          ENCODE(MSG,1ØØ)
ØØ8ØØ   1ØØ    FORMAT('PRESET AND POINT TEST')
ØØ9ØØ          CALL SETXY(Ø,Ø)
Ø1ØØØ          CALL LOCATE(Ø)
Ø11ØØ          CALL GPRINT(23,MSG)
Ø12ØØ          CALL WAIT
Ø13ØØ          CALL CLS
Ø14ØØ   C
Ø15ØØ   C      SET AND CHECK ALL PIXELS
Ø16ØØ   C
Ø17ØØ          DO 1Ø I=Ø,639
Ø18ØØ          DO 11 J=Ø,239
Ø19ØØ          CALL SETXY(I,J)
Ø2ØØØ          CALL PRESET(1)
Ø21ØØ          K=POINT(L)
Ø22ØØ          IF(K.EQ.Ø) GOTO 999
Ø23ØØ   11     CONTINUE
Ø24ØØ   1Ø     CONTINUE
Ø25ØØ   C
Ø26ØØ   C      RESET AND CHECK ALL PIXELS
Ø27ØØ   C
Ø28ØØ          DO 12 I=Ø,639
Ø29ØØ          DO 13 J=Ø,239
Ø3ØØØ          CALL SETXY(I,J)
Ø31ØØ          CALL PRESET(Ø)
Ø32ØØ          K=POINT(L)
Ø33ØØ          IF (K.EQ.1) GOTO 999
Ø34ØØ   13     CONTINUE
Ø35ØØ   12     CONTINUE
Ø36ØØ          CALL CLS
Ø37ØØ          ENCODE(MSG,1Ø1)
Ø38ØØ   1Ø1    FORMAT('PRESET AND POINT PASSED')
Ø39ØØ          CALL SETXY(Ø,Ø)
Ø4ØØØ          CALL LOCATE(Ø)
Ø41ØØ          CALL GPRINT(23,MSG)
Ø42ØØ          GOTO 1ØØØ
Ø43ØØ   999    CALL CLS
Ø44ØØ          ENCODE(MSG,1Ø2)
Ø45ØØ   1Ø2    FORMAT('PRESET AND POINT FAILED')
Ø46ØØ          CALL SETXY(Ø,Ø)
Ø47ØØ          CALL LOCATE(Ø)
Ø48ØØ          CALL GPRINT(23,MSG)
Ø49ØØ   1ØØØ   CALL WAIT
```

```
Ø5ØØØ          RETURN
Ø51ØØ          END
```

```
ØØ1ØØ           SUBROUTINE SCRTST
ØØ2ØØ   C
ØØ3ØØ   C       SCREEN TEST
ØØ4ØØ   C
ØØ5ØØ           LOGICAL MSG(11)
ØØ6ØØ           CALL CLS
ØØ7ØØ           ENCODE(MSG,1ØØ)
ØØ8ØØ   1ØØ     FORMAT('SCREEN TEST')
ØØ9ØØ           CALL SETXY(Ø,Ø)
Ø1ØØØ           CALL LOCATE(Ø)
Ø11ØØ           CALL GPRINT(11,MSG)
Ø12ØØ           CALL WAIT
Ø13ØØ           CALL SETXY(3ØØ,12Ø)
Ø14ØØ           CALL CIRCLE(1ØØ,1,Ø.Ø,6.28,Ø.5)
Ø15ØØ           CALL CIRCLE(1ØØ,1,Ø.Ø,6.28,Ø.25)
Ø16ØØ           CALL CIRCLE(5Ø,1,Ø.Ø,6.28,Ø.5)
Ø17ØØ           CALL PAINT(1,1)
Ø18ØØ   C
Ø19ØØ   C       GRAPHICS SCREEN
Ø2ØØØ   C
Ø21ØØ           CALL SCREEN(Ø)
Ø22ØØ           CALL WAIT
Ø23ØØ           CALL WAIT
Ø24ØØ           CALL WAIT
Ø25ØØ   C
Ø26ØØ   C       TEXT SCREEN
Ø27ØØ   C
Ø28ØØ           CALL SCREEN(1)
Ø29ØØ           CALL WAIT
Ø3ØØØ           CALL WAIT
Ø31ØØ           CALL WAIT
Ø32ØØ   C
Ø33ØØ   C       GRAPHICS SCREEN
Ø34ØØ   C
Ø35ØØ           CALL SCREEN(Ø)
Ø36ØØ           CALL WAIT
Ø37ØØ           CALL WAIT
Ø38ØØ           CALL WAIT
Ø39ØØ           RETURN
Ø4ØØØ           END
```

```
ØØ1ØØ               SUBROUTINE VTEST
ØØ2ØØ   C
ØØ3ØØ   C           VIEW AND FVIEW TEST
ØØ4ØØ   C
ØØ5ØØ               INTEGER FVIEW
ØØ6ØØ               LOGICAL MSG(19)
ØØ7ØØ               CALL CLS
ØØ8ØØ               ENCODE(MSG,1ØØ)
ØØ9ØØ   1ØØ         FORMAT('VIEW AND FVIEW TEST')
Ø1ØØØ               CALL SETXY(Ø,Ø)
Ø11ØØ               CALL LOCATE(Ø)
Ø12ØØ               CALL GPRINT(19,MSG)
Ø13ØØ               CALL WAIT
Ø14ØØ   C
Ø15ØØ   C           DRAW VIEWPORT AND CIRCLES
Ø16ØØ   C
Ø17ØØ               CALL VIEW(Ø,4Ø,639,239,Ø,1)
Ø18ØØ               CALL DCIRCL(1)
Ø19ØØ   C
Ø2ØØØ   C           DRAW VIEWPORT AND LINES
Ø21ØØ   C
Ø22ØØ               CALL VIEW(2Ø,5Ø,619,229,1,Ø)
Ø23ØØ               CALL DLINE(Ø)
Ø24ØØ   C
Ø25ØØ   C           DRAW VIEWPORT AND CIRCLES
Ø26ØØ   C
Ø27ØØ               CALL VIEW(4Ø,6Ø,599,2Ø9,Ø,Ø)
Ø28ØØ               CALL DCIRCL(1)
Ø29ØØ   C
Ø3ØØØ   C           DRAW VIEWPORT AND LINES
Ø31ØØ   C
Ø32ØØ               CALL VIEW(6Ø,7Ø,579,199,1,1)
Ø33ØØ               CALL DLINE(Ø)
Ø34ØØ   C
Ø35ØØ   C           CLEAR SCREEN
Ø36ØØ   C
Ø37ØØ               IX1=FVIEW(Ø)
Ø38ØØ               IY1=FVIEW(1)
Ø39ØØ               IX2=FVIEW(2)
Ø4ØØØ               IY2=FVIEW(3)
Ø41ØØ               CALL VIEW(6Ø-IX1,7Ø-IY1,6Ø+IX2,4Ø+IY2,Ø,1)
Ø42ØØ               CALL CLS
Ø43ØØ               RETURN
Ø44ØØ               END
```

```
Ø45ØØ          SUBROUTINE DCIRCL(ICLR)
Ø46ØØ          CALL SETXY(1ØØ,1ØØ)
Ø47ØØ          DO 1Ø I=5,3ØØ,5
Ø48ØØ          CALL CIRCLE(I,ICLR,Ø.Ø,6.28,Ø.5)
Ø49ØØ    1Ø    CONTINUE
Ø5ØØØ          CALL WAIT
Ø51ØØ          RETURN
Ø52ØØ          END

Ø53ØØ          SUBROUTINE DLINE(ICLR)
Ø54ØØ          DO 11 I=2,2ØØ,4
Ø55ØØ          CALL SETXY(-1Ø,-1Ø)
Ø56ØØ          CALL SETXY(I+2ØØ,I)
Ø57ØØ          CALL LINE(ICLR,-1)
Ø58ØØ    11    CONTINUE
Ø59ØØ          CALL WAIT
Ø6ØØØ          RETURN
Ø61ØØ          END
```

```
ØØ1ØØ          SUBROUTINE WAIT
ØØ2ØØ    C
ØØ3ØØ    C     THIS SUBROUTINE INTRODUCES A TIME DELAY
ØØ4ØØ    C
ØØ5ØØ          DO 11 J=1,2Ø
ØØ6ØØ          DO 1Ø I=1,1ØØØØ
ØØ7ØØ    1Ø    CONTINUE
ØØ8ØØ    11    CONTINUE
ØØ9ØØ          RETURN
Ø1ØØØ          END
```

Appendix E/ Base Conversion Chart

DEC.	HEX.	BINARY	DEC.	HEX.	BINARY
0	00	00000000	40	28	00101000
1	01	00000001	41	29	00101001
2	02	00000010	42	2A	00101010
3	03	00000011	43	2B	00101011
4	04	00000100	44	2C	00101100
5	05	00000101	45	2D	00101101
6	06	00000110	46	2E	00101110
7	07	00000111	47	2F	00101111
8	08	00001000	48	30	00110000
9	09	00001001	49	31	00110001
10	0A	00001010	50	32	00110010
11	0B	00001011	51	33	00110011
12	0C	00001100	52	34	00110100
13	0D	00001101	53	35	00110101
14	0E	00001110	54	36	00110110
15	0F	00001111	55	37	00110111
16	10	00010000	56	38	00111000
17	11	00010001	57	39	00111001
18	12	00010010	58	3A	00111010
19	13	00010011	59	3B	00111011
20	14	00010100	60	3C	00111100
21	15	00010101	61	3D	00111101
22	16	00010110	62	3E	00111110
23	17	00010111	63	3F	00111111
24	18	00011000	64	40	01000000
25	19	00011001	65	41	01000001
26	1A	00011010	66	42	01000010
27	1B	00011011	67	43	01000011
28	1C	00011100	68	44	01000100
29	1D	00011101	69	45	01000101
30	1E	00011110	70	46	01000110
31	1F	00011111	71	47	01000111
32	20	00100000	72	48	01001000
33	21	00100001	73	49	01001001
34	22	00100010	74	4A	01001010
35	23	00100011	75	4B	01001011
36	24	00100100	76	4C	01001100
37	25	00100101	77	4D	01001101
38	26	00100110	78	4E	01001110
39	27	00100111	79	4F	01001111

DEC.	HEX.	BINARY	DEC.	HEX.	BINARY
80	50	01010000	120	78	01111000
81	51	01010001	121	79	01111001
82	52	01010010	122	7A	01111010
83	53	01010011	123	7B	01111011
84	54	01010100	124	7C	01111100
85	55	01010101	125	7D	01111101
86	56	01010110	126	7E	01111110
87	57	01010111	127	7F	01111111
88	58	01011000	128	80	10000000
89	59	01011001	129	81	10000001
90	5A	01011010	130	82	10000010
91	5B	01011011	131	83	10000011
92	5C	01011100	132	84	10000100
93	5D	01011101	133	85	10000101
94	5E	01011110	134	86	10000110
95	5F	01011111	135	87	10000111
96	60	01100000	136	88	10001000
97	61	01100001	137	89	10001001
98	62	01100010	138	8A	10001010
99	63	01100011	139	8B	10001011
100	64	01100100	140	8C	10001100
101	65	01100101	141	8D	10001101
102	66	01100110	142	8E	10001110
103	67	01100111	143	8F	10001111
104	68	01101000	144	90	10010000
105	69	01101001	145	91	10010001
106	6A	01101010	146	92	10010010
107	6B	01101011	147	93	10010011
108	6C	01101100	148	94	10010100
109	6D	01101101	149	95	10010101
110	6E	01101110	150	96	10010110
111	6F	01101111	151	97	10010111
112	70	01110000	152	98	10011000
113	71	01110001	153	99	10011001
114	72	01110010	154	9A	10011010
115	73	01110011	155	9B	10011011
116	74	01110100	156	9C	10011100
117	75	01110101	157	9D	10011101
118	76	01110110	158	9E	10011110
119	77	01110111	159	9F	10011111

DEC.	HEX.	BINARY
160	A0	10100000
161	A1	10100001
162	A2	10100010
163	A3	10100011
164	A4	10100100
165	A5	10100101
166	A6	10100110
167	A7	10100111
168	A8	10101000
169	A9	10101001
170	AA	10101010
171	AB	10101011
172	AC	10101100
173	AD	10101101
174	AE	10101110
175	AF	10101111
176	B0	10110000
177	B1	10110001
178	B2	10110010
179	B3	10110011
180	B4	10110100
181	B5	10110101
182	B6	10110110
183	B7	10110111
184	B8	10111000
185	B9	10111001
186	BA	10111010
187	BB	10111011
188	BC	10111100
189	BD	10111101
190	BE	10111110
191	BF	10111111
192	C0	11000000
193	C1	11000001
194	C2	11000010
195	C3	11000011
196	C4	11000100
197	C5	11000101
198	C6	11000110
199	C7	11000111

DEC.	HEX.	BINARY
200	C8	11001000
201	C9	11001001
202	CA	11001010
203	CB	11001011
204	CC	11001100
205	CD	11001101
206	CE	11001110
207	CF	11001111
208	D0	11010000
209	D1	11010001
210	D2	11010010
211	D3	11010011
212	D4	11010100
213	D5	11010101
214	D6	11010110
215	D7	11010111
216	D8	11011000
217	D9	11011001
218	DA	11011010
219	DB	11011011
220	DC	11011100
221	DD	11011101
222	DE	11011110
223	DF	11011111
224	E0	11100000
225	E1	11100001
226	E2	11100010
227	E3	11100011
228	E4	11100100
229	E5	11100101
230	E6	11100110
231	E7	11100111
232	E8	11101000
233	E9	11101001
234	EA	11101010
235	EB	11101011
236	EC	11101100
237	ED	11101101
238	EE	11101110
239	EF	11101111

DEC.	HEX.	BINARY
240	F0	11110000
241	F1	11110001
242	F2	11110010
243	F3	11110011
244	F4	11110100
245	F5	11110101
246	F6	11110110
247	F7	11110111
248	F8	11111000
249	F9	11111001
250	FA	11111010
251	FB	11111011
252	FC	11111100
253	FD	11111101
254	FE	11111110
255	FF	11111111

Appendix F/ Pixel Grid Reference

The following hexadecimal numbers include commonly used tiling designs.

Important Note: You cannot use more than two empty rows of tiles when tiling or you'll get an Illegal Function Call error.

Example (four rows of empty tiles):

```
CHR$(&HFF)+CHR$(&HFF)+CHR$(&HØØ)+CHR$(&HØØ)+CHR$(&HØØ)+CHR$(&HØØ)
```

gives you an Illegal Function Call error.

1. "X"

```
CHR$(&H41)+CHR$(&H22)+CHR$(&H14)+CHR$(&HØ8)+CHR$(&H14)
+CHR$(&H22)+CHR$(&H41)+CHR$(&HØØ)
```

								Hex	Decimal
Ø	1	Ø	Ø	Ø	Ø	Ø	1	41	65
Ø	Ø	1	Ø	Ø	Ø	1	Ø	22	34
Ø	Ø	Ø	1	Ø	1	Ø	Ø	14	2Ø
Ø	Ø	Ø	Ø	1	Ø	Ø	Ø	Ø8	8
Ø	Ø	Ø	1	Ø	1	Ø	Ø	14	2Ø
Ø	Ø	1	Ø	Ø	Ø	1	Ø	22	34
Ø	1	Ø	Ø	Ø	Ø	Ø	1	41	65
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	ØØ	Ø

2. "Fine" horizontal lines

```
CHR$(&HFF)+CHR$(&HØØ)
```

								Hex	Decimal
1	1	1	1	1	1	1	1	FF	255
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	ØØ	Ø

3. "Medium" horizontal lines

```
CHR$(&HFF)+CHR$(&HFF)+CHR$(&HØØ)+CHR$(&HØØ)
```

								Hex	Decimal
1	1	1	1	1	1	1	1	FF	255
1	1	1	1	1	1	1	1	FF	255
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	ØØ	Ø
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	ØØ	Ø

4. Diagonal lines

(Right to left):

```
CHR$(&HØ3)+CHR$(&HØC)+CHR$(&H3Ø)+CHR$(&HCØ)
```

								Hex	Decimal
Ø	Ø	Ø	Ø	Ø	Ø	1	1	Ø3	3
Ø	Ø	Ø	Ø	1	1	Ø	Ø	ØC	12
Ø	Ø	1	1	Ø	Ø	Ø	Ø	3Ø	48
1	1	Ø	Ø	Ø	Ø	Ø	Ø	CØ	192

(Left to right)

```
CHR$(&HCØ)+CHR$(&H3Ø)+CHR$(&HØC)+CHR$(&HØ3)
```

								Hex	Decimal
1	1	Ø	Ø	Ø	Ø	Ø	Ø	CØ	192
Ø	Ø	1	1	Ø	Ø	Ø	Ø	3Ø	48
Ø	Ø	Ø	Ø	1	1	Ø	Ø	ØC	12
Ø	Ø	Ø	Ø	Ø	Ø	1	1	Ø3	3

5. "Fine" vertical lines

```
CHR$(&HAA)
```

								Hex	Decimal
1	Ø	1	Ø	1	Ø	1	Ø	AA	17Ø

6. "Medium" vertical lines

```
CHR$(&HCC)
```

								Hex	Decimal
1	1	Ø	Ø	1	1	Ø	Ø	CC	2Ø4

7. "Coarse" vertical lines

CHR$(&HFØ)

								Hex	Decimal
1	1	1	1	Ø	Ø	Ø	Ø	FØ	24Ø

8. One-pixel dots

CHR$(&H22)+CHR$(&HØØ)

								Hex	Decimal
Ø	Ø	1	Ø	Ø	Ø	1	Ø	22	34
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	ØØ	Ø

9. Two-pixel dots

CHR$(&H99)+CHR$(&H66)

								Hex	Decimal
1	Ø	Ø	1	1	Ø	Ø	1	99	153
Ø	1	1	Ø	Ø	1	1	Ø	66	1Ø2

1Ø. Pluses ("+")

CHR$(&H3C)+CHR$(&H3C)+CHR$(&HFF)

								Hex	Decimal
Ø	Ø	1	1	1	1	Ø	Ø	3C	6Ø
Ø	Ø	1	1	1	1	Ø	Ø	3C	6Ø
1	1	1	1	1	1	1	1	FF	255

11. Solid (all pixels ON)

CHR$(&HFF)

								Hex	Decimal
1	1	1	1	1	1	1	1	FF	255

12. "Broad" cross-hatch

CHR$(&H92)+CHR$(&H92)+CHR$(&HFF)

								Hex	Decimal
1	Ø	Ø	1	Ø	Ø	1	Ø	92	146
1	Ø	Ø	1	Ø	Ø	1	Ø	92	146
1	1	1	1	1	1	1	1	FF	255

13. "Thick" cross-hatch

CHR$(&HFF)+CHR$(&HFF)+CHR$(&HDB)+CHR$(&HDB)

								Hex	Decimal
1	1	1	1	1	1	1	1	FF	255
1	1	1	1	1	1	1	1	FF	255
1	1	Ø	1	1	Ø	1	1	DB	219
1	1	Ø	1	1	Ø	1	1	DB	219

14. "Fine" cross-hatch

CHR$(&H92)+CHR$(&HFF)

								Hex	Decimal
1	Ø	Ø	1	Ø	Ø	1	Ø	92	146
1	1	1	1	1	1	1	1	FF	255

15. Alternating pixels

CHR$(&H55)+CHR$(&HAA)

								Hex	Decimal
Ø	1	Ø	1	Ø	1	Ø	1	55	85
1	Ø	1	Ø	1	Ø	1	Ø	AA	17Ø

Appendix G/ Line Style Reference

Type	Binary Numbers	Hex	Decimal
Long dash	ØØØØ ØØØØ 1111 1111	&HØØFF	255
Short dash	1111 ØØØØ 1111 ØØØØ	&HFØFØ	-3856
"Short-short" dash	11ØØ 11ØØ 11ØØ 11ØØ	&HCCCC	-131Ø8
Solid line	1111 1111 1111 1111	&HFFFF	-1
OFF/ON	Ø1Ø1 Ø1Ø1 Ø1Ø1 Ø1Ø1	&H5555	21845
"Wide" dots	ØØØØ 1ØØØ ØØØØ 1ØØØ	&HØ8Ø8	2Ø56
"Medium" dots	1ØØØ 1ØØØ 1ØØØ 1ØØØ	&H8888	-3Ø584
"Dot-dash"	1ØØØ 1111 1111 1ØØØ	&H8FF8	-2868Ø

Index