



A11106 037071

NBSIR 85-3235

A Device Independent Graphics Kernel

Reference

**NBS
PUBLICATIONS**

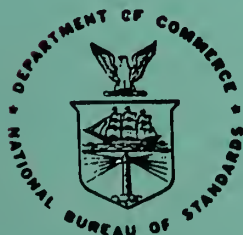
Walter W. Jones and
Alicia B. Fadell*

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Fire Research
Gaithersburg, MD 20899

*Current Address - Department of Mathematics
University of Maryland
College Park, Maryland

May 1985

Issued October 1985



U.S. DEPARTMENT OF COMMERCE

NATIONAL BUREAU OF STANDARDS

QC
100
.U56
85-3235
1985

00
100
1034
85-3235
1985

NBSIR 85-3235

**A DEVICE INDEPENDENT GRAPHICS
KERNEL**

Walter W. Jones and
Alicia B. Fadell*

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
National Engineering Laboratory
Center for Fire Research
Gaithersburg, MD 20899

*Current Address - Department of Mathematics
University of Maryland
College Park, Maryland

May 1985

Issued October 1985

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

TABLE OF CONTENTS

	Page
List of Figures.....	iv
List of Tables.....	v
1. Introduction.....	1
2. Overview.....	1
3. Schematic of the transform from the application window to a device dependent world space.....	7
4. Structure of the Data Files.....	9
5. Input/Output Devices.....	11
6. Implementation.....	12
6.1 BUILD.....	12
6.2 TITLE.....	17
6.3 ADDMAP.....	19
Appendix	
A. Description of Routines and Their Arguments.....	22
B. Control Sequences for WDDRAW.....	60
C. Character Sets (1-24).....	61
D. Description of BUILD commands.....	86
E. Description of TITLES commands.....	100
F. Examples of Usage of Routines.....	105
G. Listing of Device.....	139

List of Figures

	Page
1. Interface between application program and devices or processors.....	2
2. Schematic of the transform from the application window to a device dependent world space.....	6
3. Schematic of the transform from the application window to a viewpoint on the viewing surface.....	7
4. Filling pattern for "FILTYP".....	36
5. Default color lookup table for Lexidata 8100.....	52
6. Plot geometry used in "SURFAC"; this shows the naming convention used for the skirts and surfaces.....	53
7. Overlapping spheroids with hidden lines removed.....	58

List of Tables

	Page
1. Sample of a hardware interface protocol.....	2
2. General format and grouping of graphics routines.....	4
3. Device coordinates.....	8
4. Listing from BUILD - corresponding to data file shown in Table 3.....	10
5. Coordinate limits of the hardware.....	10

A DEVICE INDEPENDENT GRAPHICS KERNEL

Walter W. Jones
Alicia B. Fadell

Abstract

This paper describes an interface for programs which allows one to write graphics primitives to several devices without regard for the type of device. The most salient features are that it has low overhead, is transportable and can be expanded as the nature of the input/output devices changes. A conscious effort has been made to include all normal graphics primitives together with the most useful high level routines without compromising the use of special features of custom display units.

Keywords: device independence, display devices, graphics

1. INTRODUCTION

This paper describes a graphics package which is intended to ease the use of input/output devices in acquiring and displaying information graphically. The intent is to reduce the problem to its simplest level by allowing one to describe graphs in much the same way one thinks of them. A further intent is to make the user language truly device independent and allow a programmer or other user to switch devices interactively simply by identifying the desired input and output devices. As much as possible, the similarity in instructions to different devices is maintained. There are some limitations of course. Pen plotters do not normally come with erasers and most storage scopes (e.g., Tektronics) are not erasable on the individual pixel level. Beyond this no function which is available for a particular device, but not supported because there is no commonality amongst the devices, is rendered unavailable. Thus, even for specialized usage, this package will take care of normal initialization and setup.

The devices which are currently supported are Plot-10 emulators, CALCOMP pen plotters, a line printer and the Lexidata 3400/8100 series¹. These devices encompass most protocols and slots have been left in the package for future expansion. We currently support these devices since those are the ones which are available. Any suggestions for expansion or additional functions are welcome.

2. OVERVIEW

Graphics application programs may be required to send output to or accept input from several devices. Since each output device is supported by different sets of graphics routines, writing application programs is normally

¹Certain commercial equipment is identified in this paper in order to illustrate adequately certain device specific characteristics. Such identification does not imply recommendation or indorsement by the National Bureau of Standards, nor does it imply that the equipment is necessarily the best available for the purpose.

burdensome since applications may require several device-dependent versions of each program. Alternatively, each application could include all the device-dependent routines in one version of each program, and call only the routines required for a specific device, e.g., by means of GO TO statements. In either case, when a new device is added to the work environment, application programmers must learn another set of graphics routines and write new codes for each application. This reprogramming can be time-consuming and costly.

A graphics system which supports several input and output devices, while hiding the device-dependent routines from the user, is desirable. DEVICE is a package of FORTRAN subroutines for producing graphics output on several devices. The package consists of one standard routine for each output primitive (line, polygon, character, etc.). Once a device has been identified, a primitive is displayed through a call to the appropriate routine. This routine is also used to generate the same output on other devices. Hence DEVICE is an interface between the application program and the input/output devices since each standard routine is device-independent, as shown in Figure 1.

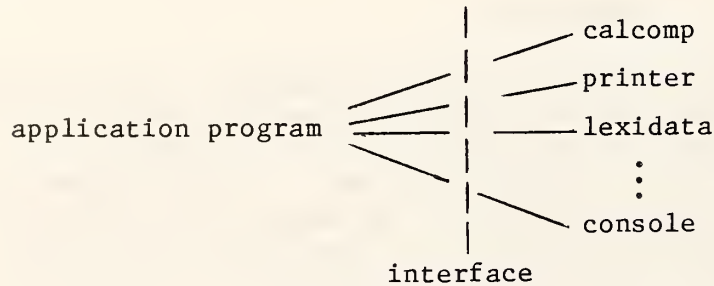


FIGURE 1

Every hardware interface routine is divided into sections. Each of these sections is devoted to a device. Once a primitive is called by a program, command goes to the section designed for the accessed device, as shown in table 1.

Table 1
A Sample of the Hardware Interface Protocol

```

SUBROUTINE SAMPLE
:
:
COMMON/DEV/IDEV
GO TO (1,2,...n) IDEV

C    CALCOMP SECTION
1    :    graphics subroutines
      RETURN

C    PRINTER SECTION
2    :    graphics subroutines
      RETURN
:
:
      END

```

The coordinates used in labeling a viewing area differ for each device. Therefore, the user's data points must be converted into device coordinates (see section 5). This conversion is done by DEVICE before device-dependent routines are called. The format of a typical graphics subroutine is

```

SUBROUTINE SAMPLE
:
:
COMMON/DEV/IDEV
GO TO (1,2,...n) IDEV

C    CALCOMP SECTION
1    convert to CALCOMP coordinates
      call CALCOMP graphics routines
      RETURN

C    PRINTER SECTION
2    convert to PRINTER coordinates
      call PRINTER graphics routines
      RETURN
:
:
C    CONSOLE SECTION
n    convert to CONSOLE coordinates
      call CONSOLE graphics routines
      RETURN
      END

```

2.1 Structure of the Package

These routines can be grouped into 6 categories summarized below and defined in table 2. More detail is given in the appendixes.

1. Device Control - utility routines which include initialization and termination procedures for the device
2. Viewing - specify the part of the user's coordinate system to display and where to place the display on the viewing surface
3. Output Primitives - define objects and display them on the viewing surface
4. Attributes - define the appearance of the output primitives
5. Auxiliary - miscellaneous routines
6. Input - acquire data.

Table 2. Graphics Subroutines - General Format

1. <u>Device Control</u>	<u>Function</u>
DEVICE (n)	initialize a graphics device
NEWFRM	clear the screen and initialize a new frame
HDCOPY	generate a hard copy
FRAME	write out the buffer
ERASE	clear the screen (also done in "NEWFRM")
ENDFRM	close the graphics device
2. <u>Viewing</u>	<u>Function</u>
SCALNG (X1, Y1, X2, Y2, X1H, Y1H, X2H, Y2H, X1S, Y1S, X2S, Y2S)	define the window and viewport
DEFINE (X1, Y1, X2, Y2)	define the window (viewport defaults to entire viewing surface)
3. <u>Output Primitives</u>	<u>Function</u>
LINE (X1, Y1, X2, Y2)	draw a line between 2 points
LINES (X1, Y1, X2, Y2, N)	draw a line between points in an array of length n
LNPLLOT (X, Y, I1, I2, I3)	draw lines between selected points of an array
PLYGON (X, Y, N)	draw a closed polygon
BOXPLT (X1, Y1, X2, Y2)	draw a rectangle
CIRCLE (X, Y, R)	draw a circle
SURFAC (Z, NX, NY, MODE)	performs surface plotting
CONTUR (F, TEST, NX, NY, FL)	performs contour plotting
VOLUME (MODE, F, NX, NY, NZ, CLEVEL, NCL, T, NT)	performs volume plotting (3 dimensional contouring)
SYMBOL (X, Y, CHAR)	draws a specified hardware character at a given point
CHPLOT (X, Y, CHAR, I1, I2, I3)	draws a specified hardware character at selected points in an array
HHDRAW (X, Y, SX, SXY, ICHAR, NSET, NSET, IERR)	draws a particular character of the specified character set
WDDRAW (X, Y, DX, DY, SX, SXY, SY, CHARS)	draws a character string

LABEL (CHARS, X1, Y1, X2, Y2, ANGLE)	draws a character string
ALABEL (CHARS, X1, X2, Y1, Y2, ANGLE)	draws a character string with aspect ratio 4/3
FNUMBR (X, Y, DX, DY, SX, SXY, SY, XNUMBR, WIDTH, DIGITS)	draws a real number in FORTRAN F-type format
ENUMBR (XNUMBR, X1, Y1, X2, Y2)	draws a real number in exponential (E) format (E - type)
GRAFIT (NPLT, X1, X2, X1R, X2R, XX1, XX2, Y1, Y2, Y1R, Y2R, YY1, YY2, XTIT, NDVX, YTIT, NDVY)	sets up a graph (x and y axis) for plotting data
PLYPLT (F, IS)	read in a "BUILD" formatted structure file, F and display according to IS
MAPIN (FW, V, NV, E, ES, NE, P, PS, NP)	read a "BUILD" file and return the vertices, edges and polygons. Files are appended after the initial call.
MAPOUT (W, V, NV, E, ES, NE, P, PS, NP)	display the specified edges as given
VIEWTR (X, W, V, NV, E, ES, NE, P, PS, NP)	display the specified edges and polygons using the transform matrix X.

4. Attributes

	<u>Function</u>
COLOR (N)	defines the color for drawing
LINWID (N)	defines the line width
FILTYP (N)	defines the appearance of the interior of polygons and circles
CHRSIZ (CHFRZ, GCHFRZ)	sets the size of the hardware characters
CHRSET (N)	changes the default character set

5. Auxiliary

	<u>Function</u>
IOWAIT (N)	puts a pause in the program
SETDEV (N1, N2)	changes logical units
DELAY (N)	delays for hard copy units
SETLUT	sets up the color look-up table

6. Input

DEVINP (I, status, X,Y,Z)	read input coordinates triplet from a device
---------------------------	---

3. COORDINATE SYSTEMS

The application must define the environment in which it will operate. For instance, temperature may be calculated in degrees celsius for one study, while length is measured in feet for another project. Data can be sent to the DEVICE routines directly from the application; no conversion to a standard unit is required of the user. Within the graphics package, however, the data coordinates must be converted to device-dependent coordinates for the individual hardware.

The user space is a 3-dimensional left-handed coordinate system (positive x-axis to the right, positive y-axis up, and positive z-axis into the viewing surface). Data points are added to the space in "application-dependent" coordinates. Before these points can be displayed, a window and a viewport must be specified.

The window, a rectangle in the x-y plane of the user space, encloses the points to be displayed. This window is mapped into the device's world space. The world space is labeled differently for each device. The world space is determined by the device's fixed points (X1S, Y1S) and (X2S, Y2S). The window is defined by the user who specifies the diagonal endpoints (X1, Y1) and (X2, Y2). The window is mapped to the world space by the scale factors:

$$\begin{aligned}XYCOORD(1) &= (X2S - X1S)/(X2 - X1) \\XYCOORD(2) &= X1S - X1 * XYCOORD(1) \\XYCOORD(3) &= (Y2S - Y1S)/(Y2 - Y1) \\XYCOORD(4) &= Y1S - Y1 * XYCOORD(3)\end{aligned}$$

A point (X,Y) in the user space is converted to the world coordinate (XS,YS) by the transformation:

$$\begin{aligned}XS &= X * XYCOORD(1) + XYCOORD(2) \\YS &= Y * XYCOORD(3) + XYCOORD(4)\end{aligned}$$

Any point within the user's window will be converted into valid world coordinates (i.e., $X1S \leq XS \leq X2S$ and $Y1S \leq YS \leq Y2S$) and can be plotted. Other points will be mapped out of the device's range and should not be displayed on the viewing surface.

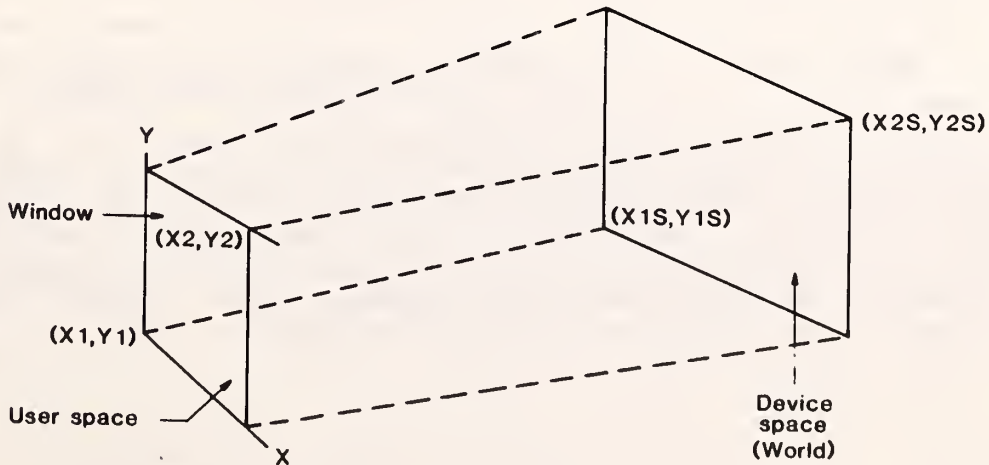


FIGURE 2

The part of the viewing surface which will contain the display is called the viewport. The viewport defaults to the entire viewing surface. In order to place the display on a smaller portion of the viewing surface, the desired rectangular area must be specified. The device coordinates $(X1H, Y1H)$ and $(X2H, Y2H)$ are used to define this viewport. The world space is then mapped onto the viewport. Hence, all data points in the user space are converted to device coordinates through 2 transformations. Only points in the user's window will be transformed into valid device coordinates (i.e., coordinates within the device's viewport).

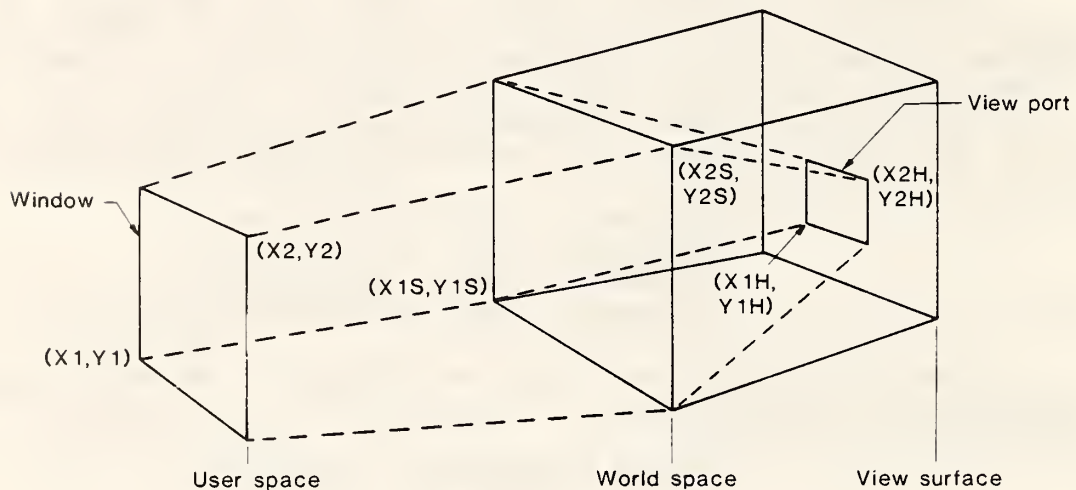


FIGURE 3

Usual usage is to transform the viewport to the entire viewable space (view surface). In this case a call to DEFINE is sufficient. The default values for the device world coordinate space are used, and the defined window is mapped to the maximum normalized device coordinates. Special applications might require other transformations. Such a facility is provided by SCALNG. This routine specifies the transform from user space to world coordinate space, from world coordinate space to normalized device coordinates and finally to the device (or hardware) address units. Table 3 shows the default values for the current list of output device.

TABLE 3
DEVICE COORDINATES

<u>OUTPUT</u>	<u>WORLD</u>		<u>NORMALIZED</u>		<u>PHYSICAL</u>	
	X	Y	X	Y	X	Y
Lexidata	128000	102300	32700	26160	1279	1023
Tek 40XX	1280	1024	1029	1029	1023	768
CALCOMP	-	-	1029	1023	10.75	8.75
<u>Printer</u>	<u>1279</u>	<u>1023</u>	<u>1279</u>	<u>1023</u>	<u>128</u>	<u>50</u>

The physical (device) coordinates map windows to the full viewing space. Some applications require use of only a portion of this space. An example is the reduction of about 15% of the linear dimensions in order to fit into a 512 line raster scan television frame.

4. STRUCTURE DATA FILES

Several programs create and modify structure files for graphical display. The elements used to compose these structures are vertices, edges, and polygons. The elements are stored in user coordinates in the data file. Attributes for these elements along with the specifications of the user space, are also stored in this file.

Listed at the beginning of the data file are the dimensions of the user space and the window space. The boundaries of the window are put into the WINDOW array: WINDOW (1 → 4) {(left - bottom), and (right - top), respectively}, while the boundaries of the user space are sent to the WORLD array: WORLD (1 → 6) {left, bottom, front, and right, top, back}, respectively. This latter three dimensional space specification is not currently used.

The following two images in the data files specify the number of vertices, elements (defined as polygons + edges) and the number of edges, and a description of the elements by groups. This latter is currently implemented only in ADDMAP, described in Section 6.3.

The data for the elements and attributes comprise the remaining part of the file. The vertex coordinates (x,y,z) are stored next, followed by element pointers. The input routines read the data for the vertices into the array VERTEX (i,j). Here, the row index, i, (possible values = 1,2,3) refers to the ith coordinate (x, y, or z, respectively) and the column vector j refers to the vertex number. For instance VERTEX (2,5) stores the value of the y coordinate of the 5th vertex. In the file, each coordinate triplet is preceded by zero (0) or one (1). A zero indicates that the vertex was deleted from the user space (i.e., the vertex is not used for displaying or for constructing edges and polygons) but was not deleted from the data file. A one (1) specifies that the vertex exists in the user space. An example would be deleting a polygon without removing the corresponding vertices.

The vertices are grouped together to form edges and polygons. Following the list of vertex coordinates is the list of elements. Each element, either an edge or a polygon, is identified by its endpoints which are in

EDGE (i,j) or POLY(i,j),

where i specifies which of the two endpoints and j indicates the element number. For example, if the fourth edge is composed of vertices 6 and 10, we have

EDGE (1,4) = 6 and EDGE (2,4) = 10

Polygons are stored in a similar manner. POLY(i,j) contains the vertex number of the ith vertex of the jth polygon. Polygons have no more than NPVERT (currently 8) vertices.

Data for each edge or polygon fills one row of the data file. The element's vertices are listed first and are followed by the element's attributes. The attributes are read into the arrays ESPEC (i,j) or PSPEC (i,j) for edges and polygons, respectively. Here the i specifies the attribute and j indicates the element number.

<u>i</u>	<u>ESPEC (i,j)</u>	<u>i</u>	<u>PSPEC (i,j)</u>
1	line width	1	line width
2	color	2	color
3	line attribute	3	fill type
4	<unused>	4	polygon (>0) or polyline (<0)

A sample data file is shown in Table 3. In this file there are 9 valid vertices, 1 "deleted" vertex, 1 edge, and 3 polygons. (Note that the last polygon is actually a polyline.) The same data as listed by the BUILD program is shown in Table 4.

Table 3

```

BUILD 0.00000E+00 0.00000E+00 1.28000E+03 1.02400E+03
-1.00000E+09-1.00000E+09-1.00000E+09 1.00000E+09 1.00000E+09 1.00000E+09
1          10      4      1
1          4
1          1.55608E+02  7.37531E+02  0.00000E+00
1          4.03576E+02  9.39818E+02  0.00000E+00
1          5.00956E+02  7.45070E+02  0.00000E+00
1          2.60015E+02  6.44554E+02  0.00000E+00
0          4.36470E+02  3.88245E+02  0.00000E+00
1          9.64768E+02  6.44554E+02  0.00000E+00
1          5.30060E+02  3.68136E+02  0.00000E+00
1          5.30070E+02  5.88014E+02  0.00000E+00
1          8.03136E+02  5.88014E+02  0.00000E+00
1          8.03136E+02  4.63627E+02  0.00000E+00
4  6  0  0  0  0  0  0  0  2  1  0  0
1  2  3  0  0  0  0  0  0  2  1  0  0
6  7  8  9  0  0  0  0  0  2  8  1  0
1  4  7  0  0  0  0  0  0  2  1  0 -1

```

The edge, polygon and polyline entries in Table 3 are integer format. In this case there are 13 entries for each element. For polygons and polylines, NPVERT entries, a zero for compatibility with "MOVIE.BYU" and the 4 NSPEC entries. For edges, here are two entries, seven zeros and the 4 ESPEC entries.

Table 4

	WORLD			WINDOW		
LEFT,BOTTOM,FRONT:	-1.0E+09	-1.0E+09	-1.0E+09	LEFT BOTTOM:	0.0	0.00
RIGHT, TOP, BACK:	1.0E+09	1.0E+09	1.0E+09	RIGHT,TOP:	1.3E+03	1.0E+03
	VERTEX(X,Y,Z)					
1)	155.61	737.53	0.00000			
2)	403.58	939.82	0.00000			
3)	500.96	745.07	0.00000			
4)	260.02	644.55	0.00000			
6)	964.77	644.55	0.00000			
7)	530.07	368.14	0.00000			
8)	530.07	588.01	0.00000			
9)	803.14	588.01	0.00000			
10)	803.14	463.63	0.00000			
	EDGES			SPECS		
1)	4	6		2	1	0 0
	POLYGONS					
1)	1	2	3	0	0	0 0 2 1 0 0
2)	6	7	8	9	0	0 0 0 0 2 8 1 0
3)	1	4	7	0	0	0 0 0 0 2 1 0 -1

5. INPUT/OUTPUT DEVICES

Currently four separate input and output devices are supported. These encompass all the common standard interfaces and the flexibility exists to include any future input/output device for which a transformation between the hardware and mathematical space can be given. The devices for output are a CALCOMP pen plotter (1012), Tektronix 40XX or emulator, Lexidata Displays (8100 and 3700) and a line printer. Devices for input are a digitizing tablet, a joystick, segments in the Lexidata picture descriptor lists which can be "picked" and ASCII input from a keyboard or file.

The software maintains three dimensional picture descriptors. Although only the Lexidata 3700 can utilize this capability directly, perspective and visual cuing can be shown on the other devices. The coordinate limits of the hardware are shown in Table 5.

TABLE 5

<u>DEVICE</u>	<u>Xmin/Xmax</u>	<u>Ymin/Ymax</u>	<u>Zmin/Zmax</u>
Tektronix 40XX	0/768	0/1023	-
Printer	1/128	format limited	-
Joystick	0/32767	0/26160	-
Lexidata 8100	0/1279	0/1023	-
Lexidata 3700	0/1279	0/1023	0/4095
Tablet	0/1219 ₉	0/9142 ₉	-
Segments	0/~ 10 ⁹	0/~ 10 ⁹	
Calcomp plotter	0/11.5	0/8.5	

These are the local address modes for pixels and not the normalized device coordinates (see section 3). This group of devices includes almost all standard protocols for graphics interfaces. These are the CORE standard, escape sequences, direct (DMA) input/output of pixels, line oriented input/output (ASCII) and the very early protocols formulated by Tektronix and CALCOMP. Slots have been left in the software to accomodate new devices. At the time of this writing, the GKS standard has not been formalized. However, if it is similar to the proposed standard then inclusion of devices which follow this protocol is straightforward.

6. IMPLEMENTATION

6.1 BUILD

The BUILD program is used to create, modify, or display files consisting of vertices, edges, and polygons. These files also contain the elements' characteristics: line width, color, and polygon fill type. The first step in creating a file is to specify the user space and the window. The user space defaults to a cube centered at the origin and having sides of length 6×10^9 . Prompts are used by the programs when needed. There are no default values for the window space. The x and y coordinates of the window must be specified with a call to

WINDOW

At this point, data for the various structures may be entered and subsequently modified or displayed. The work file, however, need not be a new file. It is possible to alter or display data of an existing file. The command to access an old file is

GET [filename]

where the file defaults to INFILE, the last file accessed. Similarly, a work file may be saved by

SAVE [filename]

In this case, the default file is OUTFILE which is the last file that was saved. A check of the filenames stored in INFILE and OUTFILE is possible by calling

STATUS

This command also indicates the number of vertices, edges, and polygons in the work file along with the maximum number of each element allowed.

The work file is modified in several ways. Elements are added to the file with

ADD { VERTEX }
 { EDGE } (default = VERTEX)
 { POLYGON }

Vertices are input by specifying three (x,y,z) coordinates. Edges are specified by pointing to two vertices and polygons are defined by indicating a minimum of 3 vertices to a maximum of NPVERT vertices.

Elements can be added from any of the input devices. The command

SET { JOYSTICK }
 { TABLET } (default = KEYBOARD)
 { KEYBOARD }

sets the default input device. At the start of program execution, the keyboard is the data input device. To add data from a different device, SET must be used before ADD.

If the attributes of new edges or polygons are to be different than the default characteristics, the command

SELECT

must be used before the ADD command. SELECT allows the specification of the line width, color, polygon fill type and a user specification parameter, the fourth element in ESPEC and NSPEC. These new values then become the default attributes.

Elements are deleted from the file with the command

DELETE { VERTEX }
 { EDGE } (default = VERTEX)
 { POLYGON }

When a vertex is deleted, any edge or polygon containing that vertex is also deleted. It should be noted that the vertex is not deleted from the permanent data file; instead, this point is marked with a flag to indicate deletion from the user space. In order to remove vertices from the data file, the

SQUEEZE

command is used. All "flagged" vertices, along with any vertices in the user space which are not used by an edge or a polygon, are deleted from the data file by SQUEEZE. When the DELETE command is used for polygons and edges, however, these elements are removed from the user space and the data file.

Once added to the user space, the elements can be modified. Elements are moved to new locations through the command

MOVE { VERTEX }
 { EDGE } (default = VERTEX)
 { POLYGON }

A vertex can be moved by specifying its new (x,y,z) coordinates. Alternatively, a single vertex or a group of vertices can be moved a distance given by an (x,y,z) displacement vector. MOVE EDGE allows a single edge or a group of edges to change location by a specified (x,y,z) distance. Polygons are moved in the same manner as edges. "MOVE" makes new vertices when moving edges or polygons. In addition, duplicates of elements can be created with

DUPLICATE { EDGE } (default = VERTEX)
 { POLYGON }

The new elements are positioned at the designated displacement from the original position.

In addition to modifying portions of the model, the entire structure can be transformed. The model is moved to another part of the user space with the command

TRANSLATE

accompanied by an (x,y,z) displacement vector. A rotation of the model about a specified point (default is the body center of gravity for the model) is possible with

ROTATE

The rotations occur about axes parallel to the x-, y-, or z-axes of coordinate system. The axis and angle of rotation are specified. The structure is scaled about its center point by

SCALE

which requires the input of a positive scale factor. If the scale factor is greater than one, the model will be magnified. If the factor is between zero and one, the structure will be reduced. Each scaling, rotation, and translation, along with the order of application, is recorded in a matrix called the transformation matrix. This matrix contains the information needed to directly convert the structure from the original configuration to the final position determined by the series of transformations.

A listing of the elements in the work file or of the transformation matrix can be obtained at the terminal with

```
LIST      { VERTEX
           EDGE
           POLYGON
           MATRIX }      (default = VERTEX)
```

Heading the vertex list are the world and window coordinates of the work file. The (x,y,z) coordinates of a designated group of vertices follow. These vertices are listed by numbers; missing numbers correspond to vertices which were removed from the user space with DELETE. The LIST EDGE command also requires the specification of a group of edges. Each edge's number and endpoint vertices are listed on a row. Similarly, the polygon numbers and component vertices are listed for a group of polygons by the LIST POLYGON command. Finally the transformation matrix is displayed with LIST MATRIX.

The lists can be sent to the printer instead of the terminal with the command

```
PRINT     { VERTEX
           EDGE
           POLYGON
           ALL
           MATRIX }      (default = ALL)
```

The PRINT commands produce lists in the same format as the corresponding LIST commands. However, groups of elements are not specified by the user. The PRINT element command lists the first through the last "element" stored in the work file. The additional command, PRINT ALL, lists the world and window coordinates; all of the vertices, edges, and polygons; and the transformation matrix.

The contents of the work file can be displayed graphically. At the start of program execution, the output device for the display defaults to the Lexidata. A different device is selected with

```
DEVICE    { CALCOMP
           PRINTER
           LEXIDATA
           CONSOLE }      (default = LEXIDATA)
```

The specified device becomes the new default for graphical display.

If the output device has associated hard copies, these copies can be generated with

```
COPY      { ON
           OFF }          (default = OFF)
```

After COPY ON has been entered, a specified number of hard copies of each subsequent graphics display will be produced. To discontinue this automatic duplication, COPY OFF must be entered. Note that the copy switch is in the OFF position at the start of program execution.

The model is displayed with

DISPLAY $\left\{ \begin{array}{l} \text{VERTEX} \\ \text{EDGE} \\ \text{POLYGON} \\ \text{ALL} \end{array} \right\}$ (default = ALL)

If the output device is a screen, the screen is cleared before the display is generated. Only the elements within the window are displayed. DISPLAY VERTEX plots the model's vertices and labels each vertex with its number. DISPLAY EDGE or DISPLAY POLYGON draws the models' edges or polygons, respectively without numbering them. All edges and polygons are shown with DISPLAY ALL. A screen can be cleared with

ERASE

Otherwise, the image will remain on the screen until DISPLAY is used again or until termination of the program.

The field of view for a display is altered with

FIELD

The point of observation can be moved relative to the x-y plane of the user's system. A distance of zero places the observer at the origin of the coordinate system. A positive distance, d, positions the observer d units on the negative z-axis. The perspective of the display changes as the viewer's distance is altered. In addition, the angle of view can be changed. This angle, originating at the observer and bisected by the z-axis, determines the scope of vision. For instance, as the angle is decreased, the scope of vision becomes narrower. At the start of program execution, the observer is positioned 10,000 units from the origin and the angle of view is 90°.

At times it is useful to modify a structure throughout a sequence of frames. The number of frames and the changes desired are specified with

ANIMATE

The structure can be translated, scaled about its center, or rotated about a particular point. Edges and polygons may be moved. The observer can be moved along the z-axis. Any combination of these modifications is possible. Note that ANIMATE only receives the parameters for each change. The sequence of frames is viewed with DISPLAY.

Some transformations may cause the model to be moved outside the window. If the location of the structure becomes unknown, there may be difficulties in retrieving the model. The model is found with

AUTO

The window and the observer are automatically moved so that the entire model can be seen. As a result of AUTO, the scale of the model may not be optimum for viewing. This situation is easily modified with SCALE and FIELD.

The commands and the user's responses to the subsequent prompts do not have to be entered at the console. The input device is changed to a specified file with

INPUT

All input is read from this file until another input file is specified. The

EXIT

command changes the input device to the console.

Finally, the program is terminated with

END

While the program is running, the

HELP

command is used to list all the commands together with a brief description of them.

6.2 TITLES

The TITLES program is used to create and modify a series of colored pictures consisting of character strings, lines, and circles. The strings are positioned on the screen by means of a joystick. Characters from any combination of the 24 character sets (see Appendix C) are used to form a string. Lines, circles, and bullets also may be added to the picture. Elements are added in various colors and sizes. Once the pictures (usually called slides) have been created, hard copies may be generated. TITLES is commonly used to produce slide presentations.

At the start of program execution, there is the option of accessing an existing data file or composing a new file. The desired option is specified by entering the filename, or, to indicate the creation of a new file, by hitting the <RETURN> key.

Once an existing file has been read, the user is asked for a command with the prompt FUNCTION=. Prompts are given when needed. If the file is new, the command defaults to

NEW,

a new slide is created and added to the work file. The default color and character set for the text on this slide is specified. Lines of text are entered and then positioned with the cursor. Colors and character sets may be changed within a character string by means of control sequences. Control sequences are also used to add subscripts and superscripts and to change the justification of the text (see Appendix B). These attributes, however, are returned to the default values for the next line of text.

Once a character string is placed on the screen, modifications may be made. The string may be deleted, moved, centered on the line, and scaled. Note that the character size resulting from a scaling becomes the default size for all characters subsequently entered. This process of adding and modifying text is continued until the <RETURN> key is entered instead of text characters. At this point, the user is informed of the number of slides in the work file.

Slides are modified in several ways. Lines of text can be corrected one line at a time with the command

CORRECT

A character string can be deleted, moved, centered on the line and scaled. When all corrections have been made to one line of text, the process is repeated for the remaining character strings on the slide.

Various elements may be added to a slide. Text is added with the command

ADD

The default color and character set are selected. Lines of text are entered and modified as with the NEW command. Other types of elements are drawn in specified colors and at given locations. Bullets (filled circles) are added with the command

BULLET

A straight line is drawn between two specified points with

LINE

Finally, with a center point and a radial distance designated, a circle is added to the slide with

CIRCLE

Once drawn, each element may be corrected in the usual manner. It may be deleted, moved, centered, or scaled.

Specified slides in the work file may be displayed on the Lexidata with the command

VIEW

Hard copies are generated with the command

PROCESS

Slides are specified along with the desired number of copies of each slide. The delay time for the hard copy device is changed with

DELAY

and the work file may be written and saved as a data file with the command

SAVE

If the work file is new, a filename is specified by the user. Otherwise, the slides are automatically written to the file last accessed. In order to access another data file, the

REREAD

command is used.

The various character sets and colors may be referenced. The command

SET

displays a specified character set. Each set has been arranged to correspond to the 96 displayable ASCII characters. When adding text to a slide, a desired character is placed in the character string by entering the corresponding ASCII character. The command

COLORS

(default = 1)

displays all the available colors in rows. The colors are numbered to correspond with the rows. The color in the top row is color number one, the next row is color number two, and so on.

The two remaining commands are HELP and END. The commands are listed (but not described) with

HELP

Finally, the program is terminated with

END

6.3 ADDMAP

This is a structure manipulation program to add pieces of structure file together. The commands are similar to those found in BUILD and TITLE. There are some additional commands to manipulate the individual pieces. The commands are

SET, NEW, GET, WINDOW, ADD, SAVE, DISPLAY, SPECS, DRAG, GROUP, TRANSLATE, ROTATE, SCALE, FIELD, DEVICE, HELP and END.

The only commands described here are those which differ from the explanation in sections 6.1 and 6.2. The implementation of several commands, such as SET and GET, differs in BUILD and TITLES, but accomplishes the same task.

The special commands to deal with adding structures involve those which force actions on only a portion of the total file. These are NEW, DRAG and

GROUP.

In general, all manipulation commands are applied to all extant vertices. However, as pieces are added with repeated GET requests, a table is maintained which points to each of these GROUPS of vertices, polygons, edges and polylines. In order to apply an operation to a single group, the command,

GROUP

is used. The response will be an integer from zero to a number no larger than the number of GETs which have been applied. If 0 is entered, then all groups are affected. If a non-zero number is used, then only that group is affected. An error is returned and the request repeated if a negative integer or a number which exceeds the maximum number of groups is entered.

DRAG

is similar to translate, but is done by a pointing device. When this command is invoked, a point to drag is specified. This is the initial point. Then a final or destination position is requested. The action is to move the appropriate vertices by this change. Essentially, it allows the user to specify a translation without knowing the actual coordinates of a map position on the screen. This program maintains the group designations internally whereas BUILD does not. The grouping is given on the third line (image) of the structures file.

The command

NEW

resets pointers and counters and is equivalent to restarting the program.

ACKNOWLEDGMENTS

As with any good software product, many people have contributed to the development of improvement of the package over several years. Special thanks to Jay Boris and Richard Peacock who wrote routines which appear here, and to the many users who have suffered through initial releases of the software, making sometimes pointed but useful comments on improvements which were needed, and usually made. Also to the numerous readers who caught poor explanations and unjustified assumptions.

APPENDIX A

The following is a description of each routine, together with its arguments. All routines are listed, although emphasis is on the higher level routines. For the "hackers", low level routines are included since special effects are sometimes desirable. The usual FORTRAN convention for integer and real (floating) numbers is maintained. Type is given only for arrays and character variables.

ALABEL

Purpose: to draw a character string with an aspect ratio of 4/3 given the width and an angle of rotation.

Usage: Call ALABEL (CHARS, X1, Y1, X2, Y2, ANGLE)

Description of Parameters:

CHARS - CHARACTER * 1 ARRAY - character string to be drawn

X1, Y1 - lower left starting point of the string

X2 - X coordinate of the right ending point of the string

Y2 - not used

ANGLE - angle (radians) by which to rotate the string. - [A positive (negative) angle causes a counter-clockwise (clockwise) rotation about the lower left starting point.

Control Sequences: See Appendix B

Method: This routine calls the subroutine WDCOUNT to determine if the string contains characters. If characters exist, LABEL calculates the space size of the characters in order to create a 4/3 aspect ratio. Finally, the subroutine WDDRAW is called to plot the text.

BOXPLT

Purpose: to draw a rectangle given the endpoints of one of the diagonals.

Usage: Call BOXPLT (X1, Y1, X2, Y2)

Description of Parameters:

X1, Y1 - coordinates of lower left corner of the box

X2, Y2 - coordinates of the upper right corner of the box

Method: The routine determines the endpoints of each side of the rectangle. These endpoints are placed in an array and sent to the subroutine LINES which plots the four sides.

CHPLOT

Purpose: to draw a specified hardware character centered at selected points in an array.

Usage: Call CHPLOT (X, Y, CHAR, I1, I2, I3)

Description of Parameters:

X, Y - ARRAY - points at which to plot the character

CHAR - CHARACTER * 1 - character to be plotted

I1 - index of first point to plot

I2 - increment at which points are to be selected to plot

I3 - index of last point to plot

Method: The routine selects the points at which to plot the character. The subroutine SYMBOL is called to draw the character at each of these chosen points.

CHRSET

Purpose: to change the default character set

Usage: Call CHRSET (N)

Description of Parameters:

N - number (1-24) corresponding to a character set

CHRSIZ

Purpose: to set the size of the hardware character

Usage: Call CHRSIZ (CHFRZ, GCHFRZ)

Description of Parameters:

CHFRZ - a fraction specifying the size of the characters relative to the screen (default = .03)

GCHFRZ - a fraction specifying the size of the characters relative to the variables Y1R and Y2R in subroutine GRAFIT (default = .04)

CIRCLE

Purpose: to draw a circle of a given radius about a specified center point.

Usage: Call CIRCLE (X,Y,R)

Description of Parameters:

X,Y - center of the circle

R - radius of the circle

Method: The routine generates commands to transform the points from the user space to the raster space. It then generates commands to plot a circle centered at the point (X,Y) with a radius of R.

COLOR

Purpose: to define the color for drawing

Usage: Call COLOR(N)

Description of Parameters:

N - the number corresponding to a desired color

Method: This command is ignored for the printer and Tektronix. For the other devices, an integer is normalized between 1 and NUMCOLOR, where NUMCOLOR is the number of colors the device can display. The Calcomp and Lexidata possess 4 and 16 colors, respectively. The numbers are normalized by modular arithmetic and numbers less than 1 are set equal to 1:

$$\begin{aligned} \text{COLOR} &= \text{MOD}(N-1, \text{NUMCOLOR}) + 1 \\ \text{COLOR} &= \text{MAX}\emptyset(\text{COLOR}, 1) \end{aligned}$$

CONTUR

Purpose: to perform surface plotting

Usage: Call CONTUR (F, TEST, NX, NY, FL)

Description of Parameters:

F - ARRAY (NX, NY) - real values of the function to be contoured

TEST - ARRAY (NX, NY) - user supplied scratch array having the same dimension as F

NX - range and dimension of i in F(i,j)

NY - range and dimension of j in F(i,j)

FL - value of F(i,j) for contouring

Method: Contours are plotted by looking at a projection of the function F(I,J) and determining if the function crosses the contouring interval within their box. Six interpolations are done. First I to I+1, then J to J+1 parallel crossings are considered. Finally, the four cases of diagonal crossings are considered by triangular interpolation of each tessellation of the box by the corner points with the center. The corner points are considered in pairs moving counter-clockwise around the projected box.

DEFINE

Purpose: to define the correspondence between the user's window and the device's world space

Usage: Call DEFINE (X1, Y1, X2, Y2)

Description of Parameters:

X1, Y1 - coordinates of the window's lower left vertex

X2, Y2 - coordinates of the window's upper right vertex

Method: The differences $X2-X1$ and $Y2-Y1$ are checked. If either of the differences is equal to zero, the difference is set equal to one. The routine then calculates the transformation which maps objects from the user's window to the device's world space. The viewport defaults to the entire viewing surface.

DELAY

Purpose: to delay for hard copy units

Usage: Call DELAY(N)

Description of Parameters:

N - the number of seconds to delay

Method: This routine calls the system routing WAIT for the Tektronix and the Matrix camera.

DEVICE

Purpose: to initialize a device

Usage: Call DEVICE(N)

Description of Parameters:

N - defines a device

= 1 - Calcomp

= 2 - Printer

= 3 - Lexidata

= 4 - TeXtronix Console

= 5 - Empty Slot

Method: If a device is already open, execution of the program is terminated. Otherwise, the desired (device) file is connected to a unit. The device defaults to the printer if the specified number "n" does not correspond to an existing device.

DEVINP

Purpose: to get a coordinate triplet from an input device.

Usage: CALL DEVINP (INPUT, MASK, STATUS, X, Y, Z)

Description of Parameters:

INPUT - input device: 1 = joystick; 2 = tablet; 3 = keyboard.

MASK - button select - by power of 2.

1 = button 1

2 = button 2

4 = button 3

8 = button 4

STATUS - returns a value corresponding to MASK for the button pushed.

X,Y,Z - coordinate values returned. For two-dimensional devices, Z is always 0.

ENDFRM

Purpose: to close the graphics device

Usage: Call ENDFRM

Method: Closes logical units associate with DEVICE. Normally these are units 7, 8, and 9. This routine does not empty the I/O buffer but does send disconnect sequences and, for screen devices, erases the screen.

ENUMBR

Purpose: to draw a real decimal number expressed with powers of 10 such that the mantissa is between 1 and 10, i.e., $y \cdot 10^n$ where $1 < |y| < 10$

Usage: Call ENUMBR (XNUMBR, X1, Y1, X2, Y2)

Description of Parameters:

XNUMBR - number to be plotted in exponential format

X1, Y1 - starting point (lower left corner) of the first character drawn

X2, Y2 - terminating point (upper right corner) of the last character drawn

Method: A real number, not equal to zero, is normalized between 0 and 10. Then FNUMBR plots this portion as scaled by the starting and terminating points (X1, Y1) and (X2, Y2). The characters "10*" are then plotted, followed by a superscript containing the normalizing power of 10.

ERASE

Purpose: to erase the screen

Usage: Call ERASE

Method: This command is ignored for the Calcomp. For display devices a simple erase sequence is sent.

FILTYP

Purpose: to define the appearance of the interior of polygons and circles

Usage: Call FILTYP(N)

Description of Parameters:

N - the number corresponding to a desired fill type

Method: There are 9 fill types (0-8) for devices supporting filling patterns. The current devices, except for the Calcomp, support these patterns. Figure (4) shows the fill pattern numbered from 0 on the lower left to 8 at the upper right of the figure.

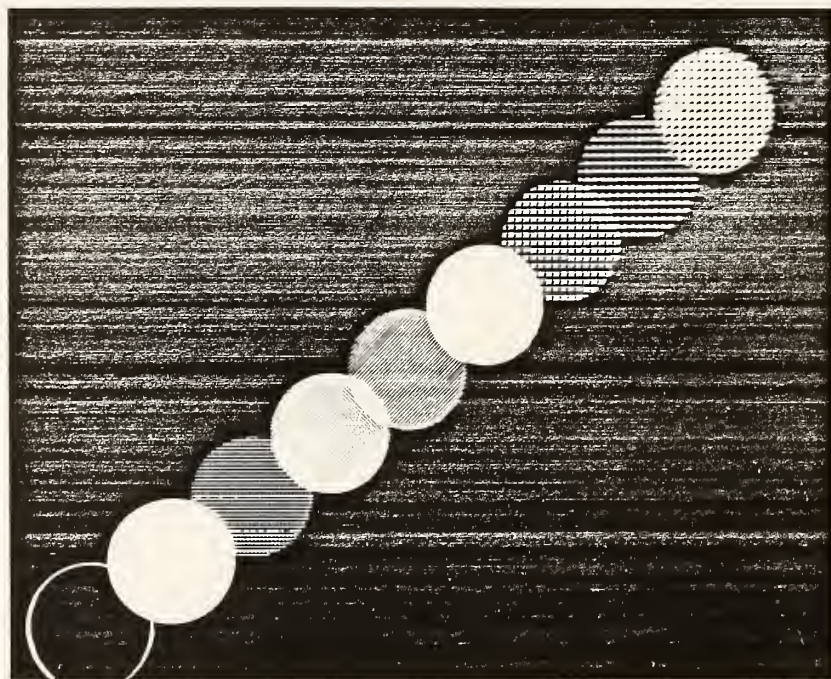


FIGURE 4

FNUMBR

Purpose: to draw a real number in FORTRAN F format

Usage: Call FNUMBR (X, Y, DX, DY, SX, SXY, SY, XNUMBR, IWIDTH, NDIGIT)

Description of Parameters:

X, Y - starting point (lower left corner) of the first character to be drawn

DX - increment added to the X-coordinate for each character drawn

DY - increment added to the Y-coordinate for each character drawn

SX - X space size for the characters

SXY - slant modifier for characters

SY - Y space size for the characters

XNUMBR - number to be plotted in F format

IWIDTH - total width of field including decimal point

NDIGIT - number of places to the right of the decimal point to be drawn

Method: The input "XNUMBR" is scaled to include the specified number of places to the right of the decimal point. Each digit in the number is converted to a literal character. Leading zeros are converted to blanks and the decimal point is inserted if NDIGIT is greater than zero. The subroutine WDDRAW is called to plot the literal data string. The parameters X, Y, DX, DY, SX, SXY, and SY have the same meaning as in WDDRAW.

FRAME

Purpose: to force out the contents of the buffer

Usage: Call FRAME

Method: This call should be used to terminate each series of graphics sequences. It is used to insure that the I/O buffers are empty and to synchronize the timing of the I/O channel.

GRAFIT

Purpose: to set up a graph (X and Y axis) for plotting data

Usage: Call GRAFIT (NPLT, X1, X2, X1R, X2R, XX1, XX2, Y1, Y2, Y1R, Y2R, YY1, YY2, XTIT, NDVX, YTIT, NDVY)

Description of Parameters:

- NPLT - number of the plot referenced (up to four graphs may be placed on one graph)
- X1,X2 - labels of X-axis minimum and maximum, respectively. (The literal string 'NONE' produces no label.)
- X1R,X2R - user space value of X-axis minimum and maximum, respectively
- XX1,XX2 - minimum and maximum values, respectively, of X in data to be plotted
- Y1, Y2 - labels of Y-axis minimum and maximum, respectively. (The literal string 'NONE' produces no label.)
- Y1R,Y2R - user space value of y-axis minimum and maximum, respectively
- YY1,YY2 - minimum and maximum values, respectively, of Y in data to be plotted.
- XTIT - char - title for X-axis (less than 30 characters and terminated by '|'.')
- NDVX - number of intervals to be drawn on X-Axis
- YTIT - char - title for Y-axis (less than 30 characters and terminated by '|'.')
- NDVY - number of intervals to be drawn on Y-axis

There are three additional entry points: PLOTCH, PLOTLN and GRISET.

- PLOTCH (NPLT, X, Y, NP CHAR) - where NPLT is the corresponding plot number
- PLOTLN (NPLT, X, Y, NP) - (See above), X and Y are coordinate arrays, NP is the number of points to plot (or connect) and CHAR is a character in CHARACTER *1 format
- GRISET (XL, YL, XR, YT) - see DEFINE

HDCOPY

Purpose: to generate a hard copy

Usage: Call HDCOPY

<u>Device</u>	<u>Associated Hard Copy</u>
Calcomp	-
Printer	-
Lexidata	Camera
Tektronix	Activates screen copy unit

Method: This command is ignored for the Calcomp and the printer.

HHDRAW

Purpose: to draw a particular character of a specified character set

Usage: Call HHDRAW (X, Y, SX, SXY, SY, ICHAR, NSET, IERR)

Description of Parameters:

X, Y - starting point (lower left) of the character to be plotted

SX - X space size for character (user coordinate system)

SXY - slant modifier for character

SY - Y space size for character (user coordinate system)

ICHAR - index to specify characters within the chosen set

NSET - number specifying a particular character set

IERR - error flag

= 0 - no errors

= 1 - error in accessing set or character

Method: If the desired character is from the hardware set, then the subroutine SYMBOL is called to plot the character. For characters from other sets, HSETS is called to place the character's coordinates and pen values in arrays and then LINES is called to plot the character as a set of connected strokes.

IOWAIT

Purpose: to put a pause in the program

Usage: Call IOWAIT (N)

Description of Parameters:

N - number of milliseconds to pause

Method: This routine calls the system routine WAIT.

LABEL

Purpose: to draw a character string of a specified width and height (i.e., a rectangular area) at a specified location

Usage: Call LABEL (CHARS, X1, Y1, X2, Y2, ANGLE)

Description of Parameters:

CHARS - CHARACTER * 1 ARRAY	- character string to be drawn
X1, Y1	- lower left starting point of string
X2, Y2	- upper right ending point of string
ANGLE	- angle (radians) by which to rotate the string. [A positive (negative) angle causes a counter-clockwise (clockwise) rotation about the point (X1, Y1)].

Control Sequences: See Appendix B

Method: This routine calls the subroutine WDCOUNT to determine if the string contains characters. If characters exist, LABEL calculates the space size of the characters and calls the subroutine WDDRAW to plot the text.

LINE

Purpose: to draw a line between two given points

Usage: Call LINE (X1, Y1, X2, Y2)

Description of Parameters:

X1, Y1 - coordinates of first point

X2, Y2 - coordinates of second point

Method: The routine transforms the points from the user space to the raster space. Subroutine PUTLVN is then called to generate a line segment to connect the points.

LINES

Purpose: to draw lines between points in an array

Usage: Call LINES (X1, Y1, X2, Y2,N)

Description of Parameters:

X1, Y1-array - starting coordinates of the lines

X2, Y2-array - ending coordinates of the lines

N - number of line pairs: [X1(i), Y1(i)] → [X2(i), Y2(i)]

Method: The routine transforms the points from the user space to the raster space. Subroutine PUTLNV is then called to generate line segments to connect corresponding points.

LINWID

Purpose: to define the line width for plotting

Usage: Call LINWID(N)

Description of Parameters:

N - the number of strokes when drawing a line - not implemented
for CALCOMP

Method: This routine sets the number of strokes for each line. (The
strokes are drawn one pixel apart.)

LNPLLOT

Purpose: to draw lines between selected points of an array

Usage: Call LNPLLOT (X, Y, I1, I2, I3)

Description of Parameters:

X - array - X coordinates of data

Y - array - Y coordinates of data

I1 . - index of first point to be connected

I2 - increment at which points are to be selected to plot

I3 - index of the last point in the data array

Method: The selected points are stored in new arrays and the subroutine LINES is called to generate each of the lines.

NEWFRM

Purpose: to erase the screen and initialize a new frame

Usage: Call NEWFRM

Method: Device dependent but in general clears buffers and initializes pointers.

PLYGON

Purpose: to draw a closed polygon

Usage: Call PLYGON (X,Y,N)

Description of Parameters:

X - array - X coordinates of the data
Y - array - Y coordinates of the data
N - number of vertices in the polygon

Method: The routine generates commands to transform coordinates from the user space to the raster space. It then generates commands to plot the polygon starting and ending at vertex (X(1), Y(1)).

SCALNG

Purpose: to define the window and the viewport

Usage: Call SCALNG (X1, Y1, X2, Y2, X1H, Y1H, X2H, Y2H, X1S, Y1S, X2S, Y2S)

Description of Parameters:

X1, Y1 - lower left corner of the window (user coordinates)

X2, Y2 - upper right corner of the window (user coordinates)

X1H, Y1H - lower left corner of viewport (device coordinates)

X2H, Y2H - upper right corner of viewport (device coordinates)

X1S, Y1S - lower left corner of world (world coordinates)

X2S, Y2S - upper right corner of world (world coordinates)

Method: Calculates the transformations to map objects from the user's window into the device's world and then into the viewport.

SETDEV

Purpose: to change logical units

Usage: Call SETDEV (N1, N2)

Description of Parameters:

N1 - logical unit for graphics output (0- ; default = 7)

N2 - logical unit for diagnostic output (1- ; default = 0)

Default Device Assignments:

<u>Type</u>	<u>LU</u>	<u>Result</u>
Diagnostics	0	no output
Graphics	7	normal - assigned
Camera*	8	assigned with Lexidata
Character set	0	unit zero-assigned
Calcomp	L7:	RS232 line
Printer	PR:	whatever
Lexidata	LEX:	DMA (L34DVR in system)
Matrix camera	L14:	RS232
Tektronix	C:	console
Open slot	NULL:	bit bucket
Console input	5	
Character set	9 (closed then opened)	
Tablet input*	8 (closed then opened)	

*Note: Camera and tablet cannot be active simultaneously

SETLUT

Purpose: to set color look-up table

Usage: Call SETLUT

Method: The default color look-up table is shown in Fig. (5) for the Lexidata 8100 . SETLUT is not currently used. It exists for users who must define look-up tables on other systems.



FIGURE 5

SURFAC

Purpose: construct and plot data surface

Usage: Call SURFAC (Z, NX, NY, MODE)

Description of Parameters:

Z - ARRAY (NX, NY) - data to be plotted as a surface

NX - dimension and range of i in Z(i,j)

NY - dimension and range of j in Z(i,j)

MODE - specifies the surface to plot
 MODE = 1: upper surface
 MODE = -1: lower surface

This subroutine and its entries construct and display plots of a surface function (two dimensional) with the hidden lines removed. The appearance of the plot is quite flexible and can include skirts around unobservable portions. It is possible to show just the upper or just the lower surface, or both together. The surface must be approximately horizontal.

SFRAME (MODE) - shows rectangular parallepiped plotting region.

SSKIRT (Z,NX,NY,MODE) - draw skirts around the plotting region (See figure 6) - uses hidden surface algorithm, so should be called last

SRFSET (X,Y,Z MIN, Z MAX, NX, NY) - initiates the plotting region

THE GEOMETRY FOR "SURFAC" PLOTS

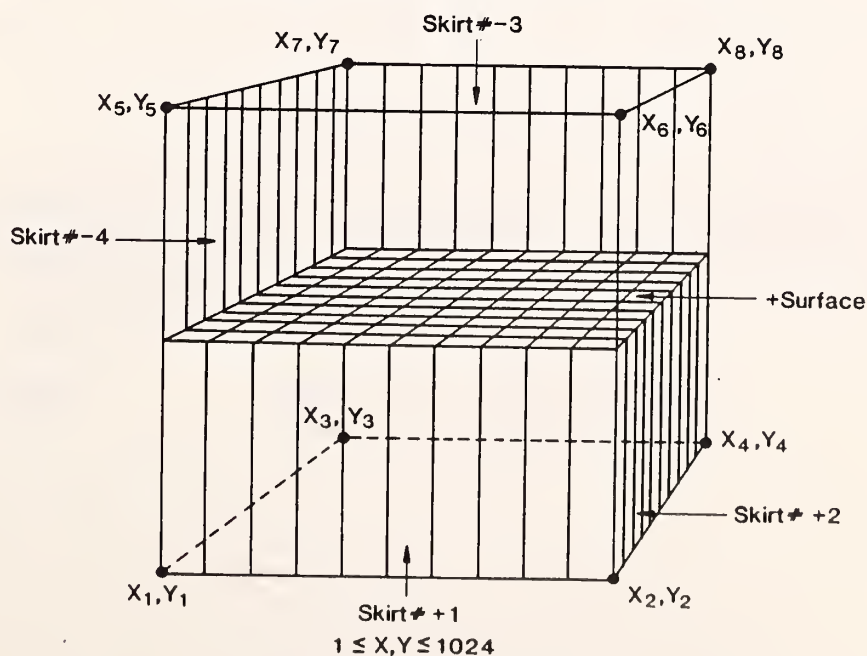


FIGURE 6

SYMBOL

Purpose: to draw a specified hardware character centred at a given point

Usage: Call SYMBOL (X, Y, CHAR)

Description of Parameters:

X, Y - location at which to plot the character

CHAR - CHARACTER*1 - character to be plotted

Method: The routine transforms the center point from the user space to the raster space. Subroutine PUTCH is then called to generate the commands to plot the character.

VOLUME

Purpose: To draw a two-dimension projection of a 3-dimensional array; simulates effect of "3D"

Usage: Call VOLUME (MODE, F, NX, NY, NZ, CLEVE, NCL, T, NT)

Description of Parameters:

F - ARRAY (NX, NY, NZ) - three-dimensional figure to be plotted

NX - range and dimension of i in F(i,j,k)

NY - range and dimension of j in F(i,j,k)

NZ - range and dimension of k in F(i,j,k)

T - ARRAY (NT, NT) - plotting array (boolean) to eliminate hidden lines

NT - dimension of the array T: T(NT,NT)

CLEVE - contour surface level for plotting

MODE - specifies the surface to be plotted

ABS(MODE) = 1: plot contour level "CLEVE" in each plane; fix hidden line matrix, T

ABS(MODE) = .2: find hidden line matrix, T, without plotting

MODE < 0: "outside" is less than "CLEVE"

MODE > 0: used when the value of the function on the "outside" is greater than "CLEVE"

NCL - number of pairs of (MODE, CLEVE) to be plotted

Method: Contours a Function F(See above) in three dimensions and projects the resultant plots onto a two dimensional plotting surface. Auxiliary entries are

VOLSET (X,Y,T,NT) - initialize the plot

VOLFRM (MODE) - MODE = 0 = corner vertices
- Mode = 1 = surrounding box

A three dimensional interpolation by triangular tessellation is used to find the contour crossing points. The technique is similar to that used in CONTUR. Once again, the hidden pixel (frame buffer) is filled as the figure is scanned back to front. A sample showing two overlapping spheroids and a cylinder is shown in Fig. 7. The functions plotted are

Cylinders: $F = (X-10.5)^2 + (Y-3.5)^2$

Sphere: $F_1 = \frac{10}{(X-8)^2 + (Y-10.5)^2 + (Z-10.5)^2}$

$$F_2 = \frac{20}{(X-13)^2 + (Y-10.5)^2 + (Z-10.5)^2}$$

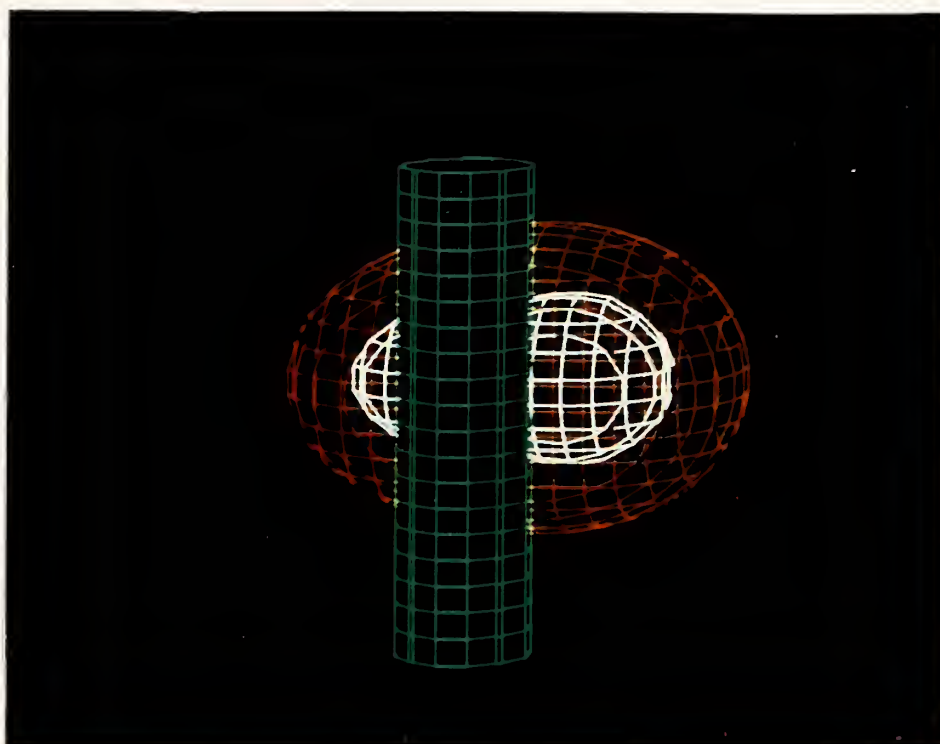


FIGURE 7

WDDRAW

Purpose: to draw a character string

Usage: Call WDDRAW (X, Y, DX, DY, SX, SXY, SY, CHARS)

Description of Parameters:

- | | |
|------|--|
| X, Y | - starting point of the character string (the string is plotted to the right of (X,Y) unless otherwise indicated by a control character) |
| DX | - increment to be added to the X-coordinate for each character drawn |
| DY | - increment to be added to the Y-coordinate for each character drawn |
| SX | - X space size for characters (user coordinate system) |
| SXY | - slant modifier for characters |
| SY | - Y space size for characters (user coordinate system) |

CHARS - CHARACTER * 1 ARRAY - character string to be drawn

Control Sequences - See Appendix B

Method: This routine scans the string for control characters and text characters. When a text character is found, the corresponding character number is obtained by the function IDCHAR. The set number assumes the default value unless the number was changed by a control character within the string. The routine also executes scaling, shifting, and rotating transformations in order to determine the starting positions of the characters. Finally, the subroutine HHDRAW is called to plot the text characters.

APPENDIX B

Control Sequences for WDDRAW, LABEL, and ALABEL

The control sequence is "|" followed by an editing character which is one of the following:

- L - justify the text to the left
- M - center the text
- R - justify the text to the right (default)
- Hnn - change from default character set to set nn
- U - draw the following characters in superscript
- D - draw the following characters in subscript
- O - reset character size and placement from superscript or subscript to original size
- B - backspace over last character drawn (works only for one character, multiple backspaces will produce unpredictable results.)
- Cnn - change from the default color to color nn
- . - end of character string

APPENDIX C

<u>Set Number</u>	<u>Alphabet</u>	<u>Style</u>	<u>Size</u>
1	Roman (Hardware characters)		
2	Roman		Cartographic
3	Greek		Cartographic
4	Roman	Simplex	Print
5	Greek	Simplex	Print
6	Script	Simplex	Print
7	Roman	Complex	Index
8	Greek	Complex	Index
9	Italic	Complex	Index
10	Roman	Complex	Print
11	Greek	Complex	Print
12	Italic	Complex	Print
13	Script	Complex	Print
14	Roman	Duplex	Print
15	Roman	Triplex	Print
16	Italic	Triplex	Print
17	German	Gothic	Print
18	English	Gothic	Print
19	Italian	Gothic	Print
20	Cyrillic	Complex	Print
21	Miscellaneous		
22	Miscellaneous		
23	Miscellaneous		
24	Miscellaneous		

Set Number 1

0	A	K	U	e	o	y	:)	,
1	B	L	V	f	p	z	;	[
2	C	M	W	g	q		!]	-
3	D	N	X	h	r	%	?	(.
4	E	O	Y	i	s	&	-)	_
5	F	P	Z	j	t	@	+		
6	G	Q	a	k	u	\$	=	"	
7	H	R	b	l	v	#	*	<	
8	I	S	c	m	w	.	/	>	
9	J	T	d	n	x	.	(o	

9	J	T	+		'	
8	I	S	&	-	.	
7	H	R	?		x	
6	G	Q	!			
5	F	P	Z	;	□	
4	E	O	Y	:)	
3	D	N	X	,	(o
2	C	M	W	.	/	-
1	B	L	V	#	*	↑
0	A	K	U	\$	=	=

Set Number 3

A	B	Ξ	Δ	E	Γ	H	I	Θ
K	Λ	M	N	O	Π	P	Σ	T
Υ	Φ	Ω	X	Ψ	Z			

9 J T d n x ;)

8 I S c m w . / >

7 H R b l v # * v

6 G Q a k u \$ = =

5 F P Z j t @ + |

4 E O Y i s & - } |

3 D N X h r % ? { c

2 C M W g q !] |

1 B L V f p z . [/

0 A K U e o y :) ;

A K r ε o ψ ;)
B Λ φ π ζ ; [/
E O ψ ı σ & - } |
Δ N X η ρ % ? ~ c
Ξ M Ω γ ;] |
H P β λ φ # * v
Γ α κ ı \$ = =
I Σ ξ μ ω · / >
Θ T δ ı χ ;)

0	1	2	3	4	5	6	7	8	9
A	B	C	D	E	F	G	H	I	J
K	L	M	N	O	P	Q	R	S	T
U	V	W	X	Y	Z	a	b	c	d
e	f	g	h	i	j	k	l	m	n
o	p	q	r	s	t	u	v	w	x
y	z		%	&	@	\$	#	.	,
:	;	!	?	-	+	=	*	/	(
)	[]	{	}		"	<	>	,
,	/	—	—	—					

A	B	Ξ	Δ	E	Γ	H	I	Ⓜ
K	Λ	M	N	O	Π	P	Σ	T
Υ	Φ	Ω	X	Ψ	Z	β	ξ	δ
ε	γ	η	ι	ι	ϑ	λ	μ	ν
ο	π	ρ	σ	τ	υ	φ	ω	χ
ψ	ζ							

A	B	C	D	E	F	G	H	I	J
K	L	M	N	O	P	Q	R	S	T
U	V	W	X	Y	Z	a	b	c	d
e	f	g	h	i	j	k	l	m	n
o	p	q	r	s	t	u	v	w	x
y	z	!	%	&	@	\$	#	.	,
:	;	!	?	-	+	=	*	/	(
)	[]	{	}		"	<	>	,
'	/	—	—	—	—	—	—	—	—

0 A K U e o y ;) ,
 1 B L V f p z ; [/
 2 C M W g q !] _
 3 D N X h r % ? { (
 4 E O Y i s & - } -
 5 F P Z j t @ + |
 6 G Q a k u \$ = "
 7 H R b l v # * <
 8 I S c m n w . / >
 9 J T d n x ; (,

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\] ^ _ ` a b c d e f g h i j k l m n o p q r s t u v w x y z { | } ~

A B C D E F G H I J
K L M N O P Q R S T
U V W X Y Z a b c d
e f g h i j k l m n
o p q r s & t u v w x
y z ! [] ^ _ `
 ; : ; [] ^ _
) , ; / >

0	1	2	3	4	5	6	7	8	9
A	B	C	D	E	F	G	H	I	J
K	L	M	N	O	P	Q	R	S	T
U	V	W	X	Y	Z	a	b	c	d
e	f	g	h	i	j	k	l	m	n
o	p	q	r	s	t	u	v	w	x
y	z	!	%	&	@	\$	#	.	,
:	;	!	?	&	+	=	*	/	(
)	[]	{	}		"	<	>)
,	/	—	—	—	—	—	—	—	—

9 J T d n x ;)
 8 I S c m w . / >
 7 H R b l v # * >
 6 G Q a k u \$ = "
 5 F P Z i t @ + |
 4 E O Y i s & - } |
 3 D N X h r % ? ~ ()
 2 C M W g q i j |
 1 B L V f p z ; [/
 0 A K U e o y ;) ,

0	A	K	U	e	o	y	:)
1	B	L	V	f	p	z	:]
2	C	M	W	g	q		!]
3	D	N	X	h	r	%	?	{
4	E	O	Y	i	s	&	-	}
5	F	P	Z	j	t	@	+	
6	G	Q	a	k	u	\$	=	"
7	H	R	b	l	v	#	*	<
8	I	S	c	m	w	.	/	>
9	J	T	d	n	x		,	(

0	1	2	3	4	5	6	7	8	9
A	B	C	D	E	F	G	H	I	J
K	L	M	N	O	P	Q	R	S	T
U	V	W	X	Y	Z	a	b	c	d
e	f	g	h	i	j	k	l	m	n
o	p	q	r	s	t	u	v	w	x
y	z	!	%	&	@	\$	#	.	,
;	:	!	?	-	+	=	*	/	;
)]	[{	}		"	<	>	
/									

h n d p n x ;

B H G r m w . / >

7 K R b l u # * >

H Q O a k u \$ = :

5 R Q G i t @ + |

4 Q O B i s & - } |

E D C X h r % ? { c

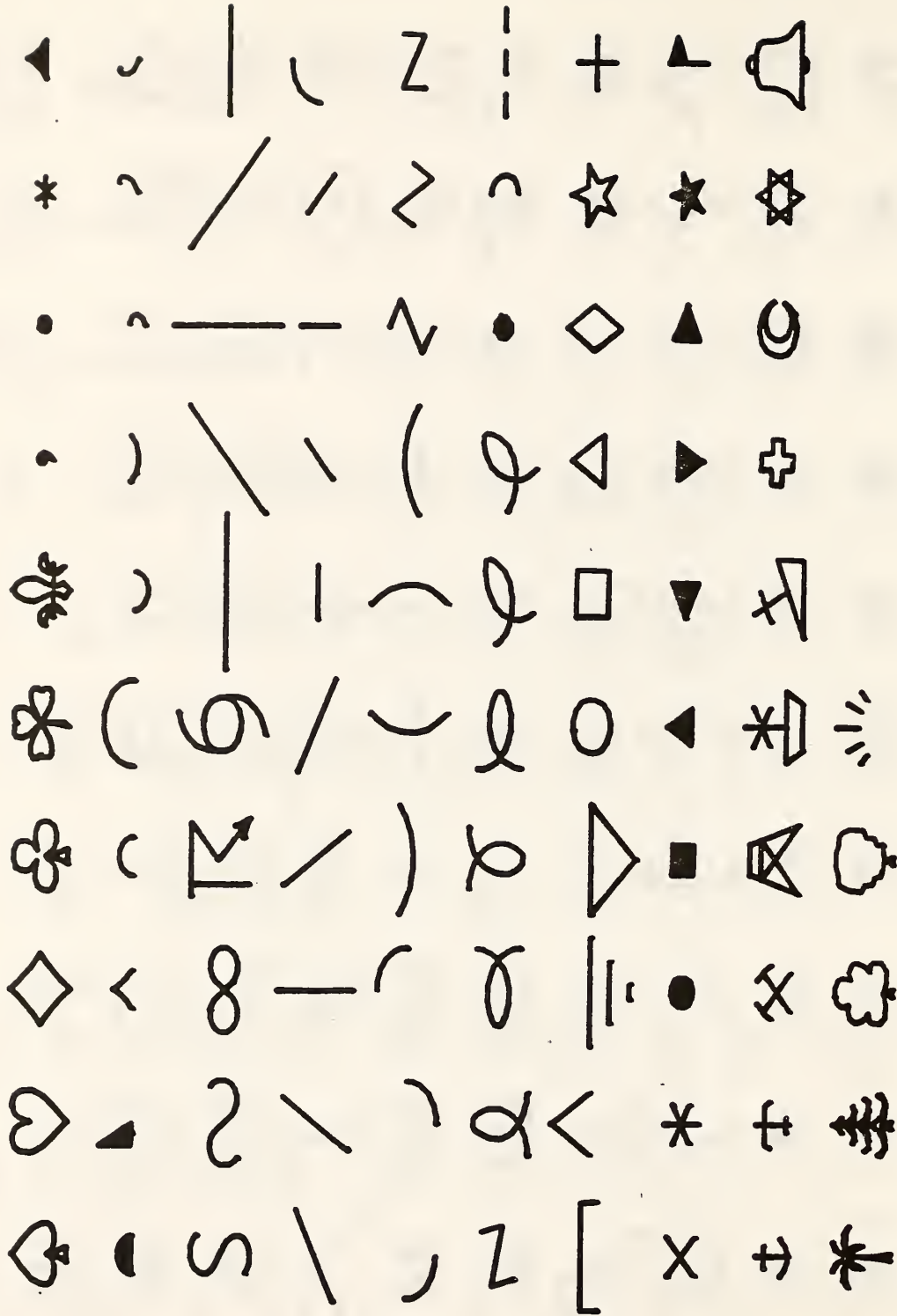
2 Q R W g q ! [|

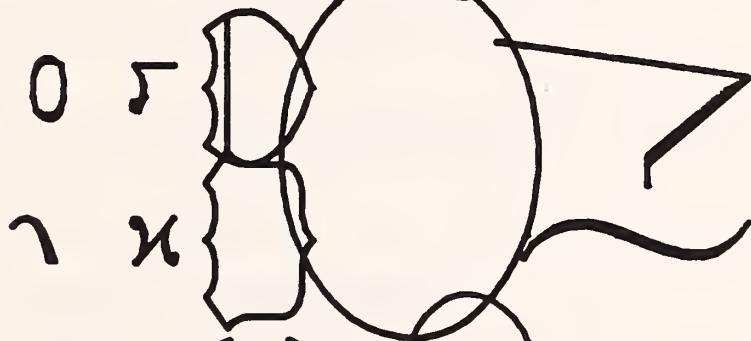
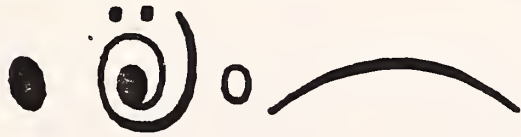
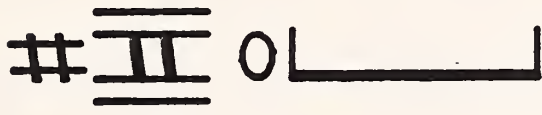
1 B H V f p 3 : [/

H R U e o y :)

□	=	≡)	↑	∨	·	∠	∩
&	—	≠	∠	∩	8	·	∩	∩
#	>	≡	∠	∩	∩	X	∩	>
↑	∨	∩	∠	∩	∩	∩	∩	·
,	o	·	∩	∩	>	∩	∩	·
,	·	X	∩	∩	∩	+	∩	·
·	∩	∩	∩	∩	∩	∩	∩	·
*	∩	∩	∩	∩	→	≡	≡)
o	∩	+	∩	∩	∩	∩	∩	∩
-	≡	∩	∩	∩	←	∩	∩	∩

§	h	Σ	∫	h	h	∇	♀	*	m
†	♁	(∇	m	m	∫	⊕	∞	↗
‡	♁)	ℓ	m	m	♁	♂	♂	♁
⊚	⊚	⊚	⊚	⊚	⊚	⊚	⊚	⊚	⊚
♁	*	~	φ	φ	←	‡	♁	♁	♁
⊕	∞	∫	∫	∫	→	⊚	⊚	⊚	⊚
♂	♂	~	∫	∫	∂	⊚	⊚	⊚	⊚
24	∫	∇	h	h	∇	♂	♂	⊚	⊚





APPENDIX D

The following is a description of each BUILD command. The prompts associated with each command and the format of the user's response are indicated.

Note: When specifying a command, at least two letters of each word must be entered (e.g. AD PO means ADD POLYGON).

1. ADD

Elements are added to the arrays.

1.1 ADD VERTEX

ENTER (X,Y,Z) COORDINATES AND SELECT BUTTON

BUTTON 1 (KEYBOARD = '1')-- ENTER VERTEX

BUTTON 2 (KEYBOARD = '2')-- QUIT

If the input device is the joystick or the tablet, the vertex is selected with the picking implement and then button 1 is pressed. If the keyboard is used, the prompt

X,Y,Z, IDV=>

appears. Three real numbers, followed by the number 1, are entered. These numbers are separated by commas or spaces.

In both cases, the X,Y,Z coordinates of the vertex are stored in the VERTEX array. Vertices are added until the VERTEX array is filled or the user wants to quit. In order to quit, button 2 is pressed. At the keyboard, any three numbers, followed by the number 2, will terminate the process.

(Alternatively, three commas followed by the number 2 can be entered: ,,,2)

1.2 ADD EDGE

BY VERTEX NO. (1) OR COORDINATES (2)?

An integer is entered. The number 1 is entered if the edges are to be composed of vertices in the VERTEX array. The number 2 is entered if the edges will be created from new vertices.

If the first method is chosen, there will be the prompt

ENTER ENDPOINTS -- 2 VERTEX NUMBERS

TO QUIT HIT <RETURN> OR ENTER 'QUIT'

VERTEX NUMBERS:

Two integers, separated by a comma or space, are entered. These numbers, indicating the vertices which define the edge, are stored in the EDGE array. The user is prompted for another edge by

VERTEX NUMBERS:

The process continues until the EDGE array is filled or the user has finished adding edges.

If the second method is selected (i.e. new vertices will be chosen), the user is prompted

ENTER 2 ENDPOINT COORDINATES:

BUTTON 1 (KEYBOARD = '1') -- ENTER VERTEX FOR EDGE

BUTTON 2 (KEYBOARD = '2') -- LAST VERTEX FOR EDGE

BUTTON 3 (KEYBOARD = '4') -- QUIT

The vertices are specified as in ADD VERTEX. The first vertex of an edge is followed by button 1 ('1' on the keyboard) and the second vertex is followed by button 2 ('2' on the keyboard). The new vertices are added to the VERTEX array and the new edge is added to the EDGE array. Additional edges can be created until the VERTEX or EDGE array is filled or the user wants to quit. Button 3 ('4' on the keyboard) is used to terminate the process.

Note: In both cases, if a newly formed edge exists in the EDGE array, this element will not be stored again. A message will appear to inform the user of the condition.

1.3 ADD POLYGON

Polygons are added in the same manner as edges except that at least three vertices are needed to define a polygon. Also, there is a limit of NPVERT vertices for a polygon. Polygons can be created from existing vertices or from new vertices. When entering new vertices, button 1 ('1' on the keyboard) is pressed after each vertex except the last one in a polygon. This last vertex is followed by button 2 ('2' on the keyboard). Finally, button 3 ('4' on the keyboard) terminates the process.

New polygons are added to the POLY array provided that they don't already exist in the array. (Messages will appear to indicate existing polygons.) Any new vertices will be added to the VERTEX array.

2. ANIMATE

A sequence of frames is specified. This sequence is viewed with DISPLAY.

NUMBER OF FRAMES

An integer is entered to indicate the number of frames.

TOTAL TRANSLATION CHANGE (DX,DY,DZ)

Three real numbers separated by commas, are entered. These numbers indicate the total displacement of the model in the X,Y,Z directions.

TOTAL ROTATION CHANGE (DX,DY,DZ)

Three real numbers, separated by commas, are specified. The model can be rotated about a point (specified in the next step) with respect to X',Y',Z' axes. These axes pass through the specified point and are parallel to the X,Y,Z axes of the coordinate system. The three numbers entered indicate the angles (in degrees) of rotation about the X',Y',Z' axes. Note that the rotations will be made in the X',Y',Z' order.

RELATIVE ORIGIN

The point about which rotations occur is specified. The center for the model is chosen by hitting the <RETURN> key. A different point is chosen by entering the appropriate X,Y,Z coordinates. The coordinates are entered as three real numbers separated by commas.

DISTANCE TO ORIGIN (DELTA)

A real number indicating the total change in the observer's position, is entered. A positive number moves the observer along the Z-axis in the negative direction (i.e. away from the viewing surface). A negative number causes the observer to move in the positive direction along the Z-axis.

SCALE FACTOR

A positive real number is entered. If the number is greater than one, the model is magnified. The model is reduced if the number is less than one.

MOVE EDGE(S)

There are several acceptable formats for the response:

```
n
n-m
n-
n BY X,Y,Z
n-m BY X,Y,Z
n- BY X,Y,Z
```

One edge (n) or a group of edges (n-m) may be moved. Note n- represents all the edges from n through the last element in the EDGE array. n and m are entered as positive integers. The relative displacement vector is specified by the real numbers X,Y,Z. If this displacement vector is not specified, the user is prompted for it.

MOVE POLYGON(S)

One polygon (n) or a group of polygons (n-m) may be moved by a distance X,Y,Z. The specifications are entered in the same format as for MOVE EDGE(S).

Note: Any of these transformations are omitted from the animated sequence by hitting <RETURN> after a particular prompt.

3. AUTO

The window and observer are moved so that the entire model appears in the field of view. This command is particularly useful if the location of the model outside the window becomes unknown to the user. New parameters are calculated by AUTO. A display does not automatically follow. The DISPLAY command must be used to view the model.

Note: The scaling of the model may be altered as a result of using AUTO. It may be necessary to compensate for this change by using the SCALE command.

4. COPY

The switch for hard copy of the graphics display is set.

4.1 COPY ON

NO. OF COPIES

An integer is entered to indicate the number of hard copies of each display to produce. This number defaults to one (1) if the <RETURN> key is pressed. Following each graphics display on the output device, hard copies, associated with this device, are generated.

4.2 COPY OFF

The automatic generation of hard copies after each graphics display is discontinued.

Note: At the start of program execution, the copy switch is set off.

5. DELETE

Elements are deleted from the arrays.

5.1 DELETE VERTEX

Vertices are not actually deleted from the VERTEX array. Instead, these "deleted" vertices are made unavailable for displaying and for constructing edges or polygons. Vertices are deleted from the array with the SQUEEZE command.

ENTER VERTEX LINE NUMBERS
HIT <RETURN> OR ENTER 'QUIT' WHEN DONE
VERTEX n=

There are three formats for entering vertices:

n
n-m
n-

where n and m are positive integers. One vertex (n) or a group of vertices (n-m) may be deleted. n- represents all the vertices from n through the last element in the VERTEX array. Once these vertices are deleted, the user is prompted for other vertices to delete with

VERTEX n=

This process is repeated until all vertices are deleted or until the user has finished.

5.2 DELETE EDGE

ENTER LINE NUMBERS.
HIT <RETURN> OR ENTER 'QUIT' WHEN DONE
LINE NUMBER(S):

One edge (n) or a group of edges (n-m) may be deleted. These specifications are entered in the same format as the DELETE VERTEX. The user is prompted for LINE NUMBER(S): until the EDGE array is empty or the user is done.

5.3 DELETE POLYGON

ENTER LINE NUMBERS.
HIT <RETURN> OR ENTER 'QUIT' WHEN DONE
LINE NUMBER(S):

One polygon (n) or a group of polygons (n-m) may be deleted. These specifications are entered in the same format as the DELETE VERTEX. The user is prompted for LINE NUMBERS: until the POLY array is empty or the user is finished.

6. DEVICE

At the beginning of program execution, the output device for graphics display defaults to the Lexidata. DEVICE is used to select a new default device. The possible selections are

DEVICE	$\left\{ \begin{array}{l} \text{CALCOMP} \\ \text{PRINTER} \\ \text{LEXIDATA} \\ \text{CONSOLE} \end{array} \right\}$	(default = LEXIDATA)
--------	---	----------------------

7. DISPLAY

Array elements are displayed in graphics mode.

7.1 DISPLAY VERTEX

Each vertex in the model is displayed and labelled with its vertex number.

7.2 DISPLAY EDGE

The model's edges are displayed.

7.3 DISPLAY POLYGON

The model's polygons are displayed.

7.4 DISPLAY [ALL]

All the edges and polygons in the model are displayed.

8. DUPLICATE

Elements are duplicated at other locations. The new elements are added to the array.

8.1 DUPLICATE EDGE

WHICH EDGE(S)?

The specifications for duplicating edges are entered in one of several ways:

```
n
n-m
n-
n BY X,Y,Z
n-m BY X,Y,Z
n- BY X,Y,Z
```

where n and m are positive integers and X,Y,Z are real numbers. One edge (n) or a group of edges (n-m) may be duplicated. n- represents all the edges from n through the last element in the EDGE array. The displacement vector is specified by X,Y,Z. If this vector is not specified, the user is prompted for it. Edges can be duplicated until <RETURN> is pressed or QUIT is entered.

8.2 DUPLICATE POLYGON

WHICH POLYGON(S)?

One polygon (n) or a group of polygons (n-m) may be duplicated. The specifications are entered in the same format as for DUPLICATE EDGE. The process of duplicating polygons can be terminated by hitting <RETURN> or entering QUIT.

Note: In DUPLICATE, the displacement vector can be entered using a type of shorthand. A zero may be represented by a blank or by no character at all.

e.g.	<u>long form</u>	<u>short form</u>
	10,8,0	10,8
	0,7,0	,7
	0,0,6	,,6
	1,0,3	1,,3

9. END

The graphics display screen is erased. Execution of BUILD is terminated.

10. ERASE

The graphics display screen is erased. Note: this command is ignored for the CALCOMP.

11. EXIT

The command source is changed to the console.

12. FIELD

The field of view is altered by moving the observer and changing the angle of view.

CURRENT DISTANCE TO ORIGIN: x.xx
NEW DISTANCE TO ORIGIN?

A positive real number, X, is entered. The observer is positioned at the point (0,-X); i.e. X units in front of the viewing surface.

CURRENT ANGLE OF VIEW: x.xx
NEW ANGLE OF VIEW?

The angle is measured in degrees. A positive real number is entered.

13. GET

A structure file is opened. Elements are read into variables and arrays. This file becomes the new INFILE.

GET FILENAME

The specified file is opened and read.

GET

The INFILE is opened and read. If there is no INFILE, the user is prompted for a filename.

14. HELP

The BUILD commands and their functions are listed at the console. Only a portion of the list appears followed by

CONTINUE? Y/N

Either the letter Y (for yes) or the letter N (for no) is entered. The remaining commands are listed if Y is entered. Nothing else is listed if N is entered.

15. INPUT

The command source is changed to a specified file. Possible formats are

INPUT filename
or
INPUT

The user is prompted for a filename in the latter case.

16. LIST

The array elements and/or the transformation matrix are listed at the console.

16.1 LIST VERTEX

The world and window coordinates are listed. The user is prompted for the vertices to list:

WHICH VERTICES?

There are three formats for entering vertices:

n
n-m
n-

where n and m are positive integers. One vertex (n) or a group of vertices (n-m) may be listed. n- represents all the vertices from n through the last element in the VERTEX array. The user is prompted for more vertices with

WHICH VERTICES?

This process is repeated until the user hits <RETURN> or types QUIT.

16.2 LIST EDGE

Only the model's edges are listed.

WHICH ELEMENTS?

One edge (n) or a group of edges (n-m) may be listed. These specifications are entered in the same format as in LIST VERTEX. The user is prompted for edges until <RETURN> is pressed or QUIT is entered.

16.3 LIST POLYGON

The model's polygons are listed.

WHICH ELEMENTS?

One polygon (n) or a group of polygons (n-m) may be listed. These specifications are entered in the same format as in LIST VERTEX. The user is prompted for polygons until <RETURN> is pressed or QUIT is entered.

16.4 LIST MATRIX

The transformation matrix is listed.

17. MOVE

Elements are moved to new locations.

17.1 MOVE VERTEX

VERTEX n =

There are several possible formats for response:

```
n
n-m
n-
n TO X,Y,Z
n BY X,Y,Z
n- BY X,Y,Z
```

One vertex (n) or a group of vertices (n-m) may be moved. Note: n- represents all the vertices from n through the last element in the VERTEX array. n and m are entered as positive integers. Vertices may be moved by a distance relative to the origin; the displacement vector is given by (X,Y,Z). In addition, a single vertex can be moved to a specific point (X,Y,Z). In both cases, X,Y,Z are entered as real numbers. If the input consists of only the vertex number(s), the user will be prompted for a relative displacement vector. This process of moving vertices will continue until <RETURN> is pressed or QUIT is entered.

17.2 MOVE EDGE

MOVE EDGE(S)

The specifications for moving the edges are entered in one of several ways:

```
n
n-m
n-
n BY X,Y,Z
n-m BY X,Y,Z
n- BY X,Y,Z
```

where n and m are positive integers and X,Y,Z are real numbers. One edge (n) or a group of edges (n-m) may be moved. n- represents all the edges from n through the last element in the EDGE array. The displacement vector is specified by X,Y,Z. If this vector is not specified, the user is prompted for it. Edges can be moved until <RETURN> is pressed or QUIT is entered.

17.3 MOVE POLYGON

MOVE POLYGON(S)

One polygon (n) or a group of polygons (n-m) may be moved. The specifications are entered in the same format as for MOVE EDGE. The process of moving polygons can be terminated by hitting <RETURN> or entering QUIT.

Note: In MOVE, the displacement vector can be entered using a type of shorthand. A zero may be represented by a blank character or by no character at all.

e.g.	<u>long form</u>	<u>short form</u>
	10,8,0	10,8
	0,7,0	,7
	0,0,6	,,6
	1,0,3	1,,3

18. PRINT

The array elements and/or the transformation matrix are listed at the printer.

18.1 PRINT VERTEX

The world and window coordinates are printed. Then all the vertices are listed.

18.2 PRINT EDGE

All of the model's edges are listed.

18.3 PRINT POLYGON

All of the polygons are printed.

18.4 PRINT MATRIX

The transformation matrix is printed.

18.5 PRINT ALL

The world and window coordinates are printed. All of the vertices, edges, and polygons are listed. The transformation matrix is printed.

19. ROTATE

The model is rotated about a specified point. Rotations take place about X',Y',Z' axes. These axes passing through the given point, are parallel to the X,Y,Z axes, respectively, of the coordinate system.

ANGLES? (X,Y,Z)

Three real numbers, separated by commas, are entered. These numbers specify the angles (in degrees) of rotation about the X', Y', Z' axes, respectively. Note that the rotations will be made in the X', Y', Z' order.

RELATIVE ORIGIN? (X,Y,Z)

The point about which rotations occur is specified. The center of the model is chosen by hitting the <RETURN> key. A different point is chosen by entering the appropriate X,Y,Z coordinates. These coordinates are entered as three real numbers separated by commas.

Note: After both prompts, the response can be entered using a type of shorthand. A zero may be represented by a blank character or by no character at all.

e.g.	<u>long form</u>	<u>short form</u>
	10,8,0	10,8
	0,7,0	,7
	0,0,6	,,6
	1,0,3	1,,3

20. SAVE

Variables and array elements are written to a file. This structure file is closed and becomes the new OUTFILE.

SAVE filename

The current work file is saved in the specified file.

SAVE

The work file is saved in the OUTFILE. If there is no OUTFILE, the user is prompted for a filename.

21. SCALE

All the vertices are scaled about the center point of the structure.

SCALE FACTOR?

A positive real number is entered. If the number is greater than one, the model is magnified. The model is reduced if the number is less than one.

22. SELECT

The defaults for line width, color, and polygon filling are chosen.

LINE WIDTH (0-15)

An integer is entered. As the numbers increase, the lines become wider. At the start of program execution, the default value is 2.

COLOR (1-15)

An integer, corresponding to the desired color, is entered. Initially, the default value is 1.

FILL TYPE (0-4)

An integer, indicating a particular polygon filling, is entered. At the start, the default value is 0. (See FILTYP in Appendix A).

Note: If the default value for an attribute is not to be changed, the <RETURN> key is pressed after the prompt.

23. SET

At the beginning of program execution, the input device for data (vertices, edges, and polygons) defaults to the keyboard. SET is used to select a new default input device. The possible selections are

$$\text{SET } \left\{ \begin{array}{l} \text{KEYBOARD} \\ \text{JOYSTICK} \\ \text{TABLET} \end{array} \right\} \quad (\text{default} = \text{KEYBOARD})$$

24. SQUEEZE

Vertices which were deleted with the DELETE command and vertices which are not used in any polygon or edge are removed from the VERTEX array. The remaining vertices are shifted up in the array in order to fill empty slots. The EDGE and POLY arrays are adjusted to account for the new vertex numbers.

25. STATUS

Some general information concerning the work file appears at the console. The filenames of the INFILE and OUTFILE are listed. The number of elements in the VERTEX, EDGE, and POLY arrays are indicated along with the size of each array.

26. TRANSLATE

The vertices of the model are translated.

TRANSLATE VERTICES BY (DX,DY,DZ)

Three real numbers, separated by commas, are entered. These numbers indicate the total displacement of the model in the X,Y,Z directions. The numbers can be entered using a type of shorthand. A zero may be represented by a blank character or by no character at all.

e.g.	<u>long form</u>	<u>short form</u>
	10,8,0	10,8
	0,7,0	,7
	0,0,6	,,6
	1,0,3	1,,3

27. WINDOW

The boundaries of the window are specified.

ENTER THE LOWER LEFT VERTEX:

If the input device is the joystick or the tablet, the picking implement is used to point to the lower left corner of the window and then button 1 is pressed. If the keyboard is the input device, then the user is prompted

X, Y, Z, IDV=

Three real numbers, followed by the number 1, are entered. These numbers are separated by commas or spaces. The real numbers specify the X,Y,Z, coordinates, respectively, of the lower left corner of the window. Note: The Z-coordinate is assigned the value zero, regardless of the value entered by the user.

ENTER THE UPPER RIGHT VERTEX:

This vertex is entered in the same manner as the lower left vertex.

28. WORLD

The boundaries of the world space are specified.

ENTER THE LOWER LEFT FRONT VERTEX:

If the input device is the joystick or the tablet, the picking implement is used to point to the lower left front corner of the box which will be the world space. Button 1 is then pressed. If the keyboard is the input device, then the user is prompted

X,Y,Z, IDV=

Three real numbers, followed by the number 1, are entered. These numbers are separated by commas or spaces. The real numbers specify the X,Y,Z coordinates, respectively, of the lower left front corner of the box.

ENTER THE UPPER RIGHT BACK VERTEX:

This vertex is entered in the same manner as the lower left front vertex.

APPENDIX E

The following is a description of each TITLES command. The prompts corresponding to each command, together with the formats of the user's responses, are indicated.

1. ADD

Character strings are added to a specified slide.

FRAME=

An integer is entered to specify a slide

COLOR=

An integer (1-15) is entered to specify a default color for the new character strings.

SET=

An integer (1-24) is entered to specify a default character set for the new text.

INPUT THE STRING

The character string is entered. The character sets and colors used within the string are modified with control sequences. In addition, control sequences are used to place subscripts and superscripts in the string and to change the justification of the text. See Appendix B for a list of these sequences.

TOGGLE BUTTON FOR LOWER LEFT

The cursor is moved to select the location of the lower left corner of the string. Once the cursor is set in place, the text is displayed.

DEL(1), MOV(2), CENTER(3), SCALE(4)=

An integer (1-4) is entered:

- 1 - The string is deleted.
 - 2 - The text will be moved. TOGGLE BUTTON TO MOVE LL CORNER. The cursor is positioned at the desired location of the lower left corner of the text. Once the cursor is set in place, the text is moved.
 - 3 - The string is centered on the line.
 - 4 - The text is scaled by a SCALE FACTOR which is specified as a positive real number.
- <RETURN> - The string is not modified.

Only one correction can be made at a time. The prompt for corrections will reappear until the string is deleted or the <RETURN> key is entered.

INPUT THE STRING

Another string may be added to the slide. When no other strings are to be added, the <RETURN> key is entered.

2. BULLET

A bullet is drawn on a specified slide.

FRAME=

An integer is entered to specify the slide on which the bullet will be drawn.

COLOR=

An integer (1-15) is entered to select the color of the bullet.

TOGGLE FOR CENTER

The cursor is positioned at the desired location of the bullet. Once the cursor is set, the bullet is displayed.

3. CIRCLE

A circle is drawn on a specified slide.

FRAME=

An integer is entered to specify the slide on which the circle will be drawn.

COLOR=

An integer (1-15) is entered to select the color of the bullet.

TOGGLE FOR CENTER

The cursor is positioned at the location of the circle's center.

TOGGLE FOR RADIUS

The cursor is positioned at the location of any point on the circle. Once the cursor is set, the circle is drawn.

4. COLORS

All of the available colors are displayed.

5. CORRECT

The text on a slide is corrected one string at a time.

FRAME=

An integer is entered to specify the slide which will be corrected. A line of text is printed at the console, followed by

DEL(1), MOV(2), CENTER(3), SCALE(4)=

An integer (1-4) is entered:

- 1 - The string is deleted.
- 2 - The text will be moved. TOGGLE BUTTON TO MOVE LL CORNER. The cursor is positioned at the desired location of the lower left corner of the text. Once the cursor is set in place, the text is moved.
- 3 - The string is centered on the line.
- 4 - The text is scaled by a SCALE FACTOR which is specified as a positive real number.
- <RETURN> - The string is not modified

Only one correction can be made at a time. This prompt for corrections will reappear until the string is deleted or the <RETURN> key is entered.

The next line of text, along with the prompt for corrections, is displayed on the console. This process is repeated for each line of the text on the slide.

6. DELAY

The delay time for the hard copy device is specified. An integer is entered for this delay time.

7. END

The program is terminated.

8. HELP

The TITLES commands are listed.

9. NEW

A new slide is created. The frame number is incremented. New character strings are added to this slide as with the ADD command.

10. LINE

A line is drawn on a specified slide.

FRAME=

An integer is entered to specify the slide on which the line will be drawn.

COLOR=

An integer (1-15) is entered to select the color of the line.

TOGGLE FOR START OF LINE

The cursor is positioned at the location of one of the line's endpoints.

TOGGLE FOR END OF LINE

The cursor is positioned at the location of the other endpoint of the line. Once the cursor is set, the line is drawn.

11. PROCESS

Hard copies of specified slides are generated.

INPUT SLIDE # (0 TO PROCESS.) AND FRAME COUNT =

Two integers are entered to indicate the slide number and the number of hard copies, respectively.

INPUT SLIDE# (0 TO PROCESS,) AND FRAME COUNT=

Another slide may be specified for copying. When all the desired slides have been specified, a zero is entered. The hard copies are then processed and the program is terminated.

12. REREAD

A file is opened and data is read into variables.

RESTART FILE

The name of a data file is entered. A new file may be started by just hitting

<RETURN>.

n SLIDES
FUNCTION=

The user is informed of the number of slides in the the data file. A command is then specified by the user.

13. SAVE

The slides in the work file are written to a data file. If a data file has been opened, the slides are written to this file. Otherwise, the user is prompted for

FILE NAME.

The name of the new data file is entered.

14. SET

A character set is displayed.

SET NO.=

A integer (1-24) is entered to select a character set.

15. VIEW

A slide is displayed.

NUMBER=

An integer is entered. This number indicates the slide to be displayed. Once the image has been generated, the process is repeated with the NUMBER= prompt. When no other slides are to be displayed, a zero or the <RETURN> key is entered.

Appendix F

PROGRAM G2TEST	page	1
SUBROUTINE ANGTST	page	2
SUBROUTINE CIRC	page	4
SUBROUTINE CUBE	page	5
SUBROUTINE MM	page	7
SUBROUTINE MM3	page	8
SUBROUTINE SETMAT	page	9
SUBROUTINE TRANS	page	10
MAIN PROGRAM	page	11
SUBROUTINE GRAFIT1	page	12
SUBROUTINE GRAFIT2	page	13
SUBROUTINE RDCNL	page	15
SUBROUTINE GRAFIT3	page	16
SUBROUTINE GRAFIT4	page	18
SUBROUTINE PLT	page	19
SUBROUTINE PHASE	page	20
SUBROUTINE RUMTST	page	21
SUBROUTINE VENTV	page	22
SUBROUTINE LAYER	page	23
SUBROUTINE FIRE	page	24
SUBROUTINE PLUME	page	25
SUBROUTINE ROOM	page	26
SUBROUTINE SADDLE	page	27
SUBROUTINE SETCOL	page	29
SUBROUTINE SYMTST	page	30
SUBROUTINE VOLTST	page	31
SUBROUTINE FILNGT	page	33

```

1          PROGRAM G2TEST
2          INTEGER PB(6)
3 C
4 C      TEST EACH FUNCTION OF THE DEVICE INDEPENDENT PCKAGE
5 C
6 C
7          WRITE(5,1) ' '
8          WRITE(5,1) 'ANGLE 1'
9          WRITE(5,1) 'CIRCLE 2'
10         WRITE(5,1) 'FLYING CUBE 3'
11         WRITE(5,1) 'GRAFIT(1) 4'
12         WRITE(5,1) 'GRAFIT(2) 5'
13         WRITE(5,1) 'GRAFIT(3) 6'
14         WRITE(5,1) 'GRAFIT(4) 7'
15         WRITE(5,1) 'PHASE SPACE PLOT (CONTOUR AND SURFACE PLOT) 8'
16         WRITE(5,1) 'SINGLE ROOM FIRE 9'
17         WRITE(5,1) 'SADDLE SURFACE 10'
18         WRITE(5,1) 'HARDWARE/HERSHEY SYMBOLS 11'
19         WRITE(5,1) '3D CONTOURING 12'
20         WRITE(5,1) 'POLYGON FILLING 13'
21         CALL SYSIO(PB,41,5,' TEST =',7,0)
22         READ(5,3,END=4,ERR=4) I
23 C
24 C      GET THE DEVICE
25 C
26         CALL SYSIO(PB,33,5,'DEVICE=',7,0)
27         READ(5,2) ID
28         2   FORMAT(I)
29         CALL DEVICE(ID)
30         GO TO (201,202,203,204,205,206,207,208,209,210,211,212,
31             .   213),I
32         STOP 'NO SELECTION'
33         201  CALL ANGTST
34         202  CALL CIRC(ID)
35         203  CALL CUBE
36         204  CALL GRAFIT1(ID)
37         205  CALL GRAFIT2(ID)
38         206  CALL GRAFIT3(ID)
39         207  CALL GRAFIT4(ID)
40         208  CALL PHASE
41         209  CALL RUMTST
42         210  CALL SADDLE(ID)
43         211  CALL SYMTST
44         212  CALL VOLTST(ID)
45         213  CALL FILNGT(ID)
46         1   FORMAT(1X,A50)
47         3   FORMAT(I)
48         4   STOP
49         END
50 C

```

```

51      SUBROUTINE ANGTST
52 C
53 C      A SIMPLE TEST OF THE CHARACTER STUFF
54 C
55      CHARACTER*1 TITLE(80)
56      INTEGER PB(6)
57      CHARACTER*80 STRING
58 C
59 C      FIRST A SIMPLE DEMONSTRATION
60 C
61      CALL GRISET (0.,0.,100.,100.)
62      CALL NEWFRM
63      CALL WDDRAW (50.,90.,6.,0.,8.,0.,8.,'QLLQC02EFT'.)
64      CALL WDDRAW (50.,75.,6.,0.,8.,0.,8.,'MQC01CQC02ENTERED'.)
65      CALL WDDRAW (50.,60.,6.,0.,8.,0.,8.,'RQC01RQC02IGHT'.)
66      STRING='MQC011QC022QC033QC044QC055QC066QC077QC088QC139QC140'.
67      .
68      CALL WDDRAW (50.,45.,6.,0.,8.,0.,8.,STRING)
69      STRING='MQC01ALSOQUQC05SUPERSSCRIPTSQC01ANDQC05DSDSUBSCRIPTS
70      .Q.'
71      CALL WDDRAW (50.,30.,6.,0.,4.,0.,5.,STRING)
72      STRING = 'MABCQU1.234Q.'
73      CALL LABEL(STRING,60.,10.,90.,20.,.50)
74 C      STRING='MQC01QH04AND QC03YOU QC07QH18CAN QH04CHANGE CHARACTER
75 C      . QC03QH18SETSQ.'
76 C      CALL WDDRAW (50.,20.,6.,0.,4.,0.,5.,STRING)
77      STRING = 'QC02QH04SQUSSUSQUSQ.'
78      CALL WDDRAW (10.,10.,6.,0.,4.,0.,5.,STRING)
79      STRING = 'QC03QODQDDQDDQDDQ.'
80      CALL WDDRAW(20., 10., 6., 0., 4., 0., 5., STRING)
81      STRING = 'QC08NOW TRY ONE YOURSELF!Q.'
82      CALL WDDRAW(10.,1.,6.,0.,4.,0.,5.,STRING)
83      CALL FRAME
84      PAUSE
85 C
86 C      THE RESTART POSITION - SCART WITH THE ANGLE OF THE TEXT
87 C
88      4      CALL SYSIO(PB,33,5,'ANGLE=',6,0)
89      READ(5,1) ANGLE
90      1      FORMAT(2F)
91      IF(ANGLE.LT.0.0) GO TO 5
92      THETA = ANGLE/57.295
93      DO 7 I = 1, 80
94      7      TITLE(I) = ' '
95 C
96 C      ERASE THE SCREEN AND PREPARE TO DRAW
97 C
98      CALL NEWFRM
99      CALL SYSIO(PB,33,5,'TEXT=',5,0)
100     READ(5,3) TITLE
101     3      FORMAT(80A1)
102     DO 11 I = 1, 80
103     II = 81 - I
104     IF(TITLE(II).NE.' ') GO TO 12
105     11     CONTINUE

```

```
106          GO TO 4
107      12    TITLE(II+1) = 'q'
108          TITLE(II+2) = '.'
109      C     CALL SYSIO(PB,33,5,'LINWID=',7,0)
110      C     READ(5,2) LW
111          CALL DEFINE (0., 0., 13., 13.)
112          CALL COLOR(4)
113      C     CALL LINWID(LW)
114          CALL BOXPLT (0.5, 0.5, 12.5, 12.5)
115          CALL COLOR(5)
116          CALL WDCOUNT(TITLE,NT)
117          IF(NT.EQ.0) GO TO 4
118      C
119      C     DRAW THE LABEL
120      C
121          YB = 5.0
122          XPOS = 5.0
123          XRIGHT = XPOS + MIN(FLOAT(NT),11.)
124          YTOP = YB + 1.
125          CALL LABEL(TITLE, XPOS, YB, XRIGHT, YTOP, THETA)
126          CALL FRAME
127          GO TO 4
128      C
129      C     CLOSE THE PLOTTING DEVICE AND ERASE NON-LOCAL SCREENS
130      C
131      5     CALL ENDFRM
132          STOP
133          END
```



```
134 SUBROUTINE CIRC(ID)
135 CALL NEWFRM
136 IF(ID.NE.1) THEN
137   CALL DEFINE (0.,0.,10.,10.)
138 ELSE
139   CALL SCALNG (0.,0.,10.,10.,0.,0.,7.9,7.9,0.,0.,0.,0.)
140 ENDIF
141 CALL COLOR(2)
142 CALL LINWID(4)
143 CALL FILTYP(1)
144 CALL CIRCLE(5.,5.,4.)
145 CALL FRAME
146 PAUSE
147 CALL ERASE
148 CALL ENDFRM
149 STOP
150 END
```

SUBROUTINE CUBE

```

151      SUBROUTINE CUBE
152      COMMON QNORM(3,6),VERT(3,8),IEDGV1(12),IEDGV2(12),
153      ISURF(6),IPEDG(24),NVERT,NEDGS,NSURF,F,SCALE(4)
154      INTEGER IV(2)/2,0/
155  C
156      DIMENSION T(4,4),QM(4,4),QMT(4,4)
157      DX = 4.
158      DY = 4.
159      XL = -1.5 - 0.1*4.
160      XR = +3. + .15*4.
161      XB = -1.5 - 0.1*4.
162      XT = +3. + 0.15*4.
163      CALL DEFINE (XL, XB, XR, XT)
164      CALL DELAY(26)
165      CALL NEWFRM
166      CALL COLOR(5)
167      CALL LINWID(3)
168      NVERT = 8
169      NEDGS = 12
170      NSURF = 6
171      F = 3.
172      XINCR = .5 / 95.
173      DO 100 I = 1, 4
174      DO 100 J = 1, 4
175      T(I,J) = 0.0
176      IF(I.EQ.J) T(I,I) = 1.0
177 100  CONTINUE
178      XC = 0.5
179      YC = 0.5
180      ZC = 0.5
181      CALL SETMAT (QM,1,1.0,0,0,0)
182      DO 1000 I = 1, 95
183      CALL TRANS(T)
184      XC1 = XC - XINCR
185      YC1 = YC - XINCR
186      ZC1 = ZC - XINCR
187      QM(1,4) = -(QM(1,1)*XC + QM(1,2)*YC) + XC1
188      QM(2,4) = -(QM(2,1)*XC + QM(2,2)*YC) + YC1
189      QM(3,4) = -ZC + ZC1
190      XC = XC1
191      YC = YC1
192      ZC = ZC1
193      CALL MM (T,QM,T)
194 1000  CONTINUE
195      CALL TRANS(T)
196      CALL SETMAT(QM,2,1.0,0,0,0)
197      DO 2000 I = 1, 192
198      F = F + .15
199 2000  CALL TRANS(T)
200      CALL SETMAT(QM,3,1.0,0,0,0)
201      DO 3000 I = 1, 192
202      F = F - .15
203      CALL MM(T,QM,T)
204 3000  CALL TRANS(T)
205      CALL SETMAT(QM,1,1.0,0,0,0)

```

```
206      CALL SETMAT(QMT,2,1.0,0,0,0)
207      CALL MM(QM,QMT,QM)
208      DO 4000 I = 1, 96
209      CALL MM(T,QM,T)
210 4000  CALL TRANS(T)
211      CALL SETMAT(QMT,3,1.0,0,0,0)
212      CALL MM (QM,QMT,QM)
213      THETA = 3.14159265/180.
214      ANGLE8 = 3.141592 + THETA
215      DO 5000 I = 1, 360
216      XC1 = SIN(ANGLE8)
217      ZC1 = COS(ANGLE8)+ 1.0
218      YC1 = FLOAT(I)/360.
219      QM(1,4) = -(QM(1,1)*XC + QM(1,2)*YC + QM(1,3)*ZC) + XC1
220      QM(2,4) = -(QM(2,1)*XC + QM(2,2)*YC + QM(2,3)*ZC) + YC1
221      QM(3,4) = -(QM(3,1)*XC + QM(3,2)*YC + QM(3,3)*ZC) + ZC1
222      XC = XC1
223      YC = YC1
224      ZC = ZC1
225      ANGLE8 = ANGLE8 + THETA
226      CALL MM (T,QM,T)
227 5000  CALL TRANS(T)
228      CALL ENDFRM
229      STOP
230      END
```

```
231      SUBROUTINE MM(A,B,C)
232      DIMENSION A(4,4),B(4,4),C(4,4),D(4,4)
233      DO 10 I = 1, 4
234      DO 9 J = 1, 4
235      D(I,J) = 0.0
236      DO 8 K = 1, 4
237 8      D(I,J) = D(I,J) + B(I,K) * C(K,J)
238 9      CONTINUE
239 10     CONTINUE
240      DO 20 I = 1, 4
241      DO 20 J = 1, 4
242 20     A(I,J) = D(I,J)
243      RETURN
244      END
```

```
245      SUBROUTINE MM3(A,B,C,K)
246      DIMENSION B(4,4),A(3),C(3),D(3)
247      DO 10 I = 1, 3
248      IF(K.EQ.1) D(I) = B(I,4)
249      IF(K.NE.1) D(I) = 0.0
250      DO 9 J = 1, 3
251 9      D(I) = D(I) + B(I,J)*C(J)
252 10     CONTINUE
253      DO 20 I = 1, 3
254 20     A(I) = D(I)
255      RETURN
256      END
```

```
257      SUBROUTINE SETMAT(A, ITYPE, THETA, XT, YT, ZT)
258      DIMENSION A(4,4)
259      RAD = 3.141592 / 180.
260      DO 10 I = 1, 4
261      DO 10 J = 1, 4
262 10     A(I,J) = 0.0
263      A(4,4) = 1.0
264      ANGLE = THETA * RAD
265      CT = COS(ANGLE)
266      ST = SIN(ANGLE)
267      GO TO (100,200,300,400), ITYPE
268 100     A(1,1) = CT
269         A(2,1) = -ST
270         A(1,2) = ST
271         A(2,2) = CT
272         A(3,3) = 1.0
273         RETURN
274 200     A(1,1) = 1.0
275         A(2,2) = CT
276         A(3,2) = ST
277         A(2,3) = - ST
278         A(3,3) = CT
279         RETURN
280 300     A(1,1) = CT
281         A(3,1) = -ST
282         A(1,3) = ST
283         A(3,3) = CT
284         A(2,2) = 1.0
285         RETURN
286 400     A(1,1) = 1.0
287         A(2,2) = 1.0
288         A(3,3) = 1.0
289         A(1,4) = XT
290         A(2,4) = YT
291         A(3,4) = ZT
292         RETURN
293     END
```

```
294     SUBROUTINE TRANS(R)
295     COMMON QNORM(3,6),VERT(3,8),IEDGV1(12),IEDGV2(12),
296     ISURF(6),IPEDG(24),NVERT,NEDGS,NSURF,F,SCALE(4)
297     DIMENSION R(4,4),RVERT(3,8),QN(4)
298     DIMENSION IEDG(12),XP(12),YP(12)
299     DO 10 I = 1, NEDGS
300 10    IEDG(I) = 0
301     DO 20 I = 1, NVERT
302     CALL MM3(RVERT(1,I),R,VERT(1,I),1)
303 20    CONTINUE
304     IPTS = 1
305     DO 100 I = 1, NSURF
306     K = ISURF(I)
307     IPTF = IPTS + K - 1
308     CALL MM3(QN,R,QNORM(1,I),0)
309     K1 = IPEDG(IPTS)
310     K2 = IEDGV1(K1)
311     XN2 = -RVERT(1,K2)
312     YN2 = -RVERT(2,K2)
313     ZN2 = -RVERT(3,K2) - F
314     DOT = QN(1) * XN2 + QN(2) * YN2 + QN(3) * ZN2
315     IF(DOT.LE.0.0) GO TO 100
316     DO 30 J = IPTS, IPTF
317     JJ = IPEDG(J)
318 30    IEDG(JJ) = IEDG(JJ) + 1
319 100   IPTS = IPTF + 1
320     DO 200 I = 1, NVERT
321     ZPF = RVERT(3,I) + F
322     XP(I) = F * RVERT(1,I) / ZPF
323     YP(I) = F * RVERT(2,I) / ZPF
324 200   CONTINUE
325     CALL ERASE
326     DO 300 I = 1, NEDGS
327     IF(IEDG(I).EQ.0) GO TO 300
328     I1 = IEDGV1(I)
329     I2 = IEDGV2(I)
330     CALL LINE (XP(I1), YP(I1), XP(I2), YP(I2))
331 300   CONTINUE
332     CALL FRAME
333 C    CALL HDCOPY
334     RETURN
335     END
```

```
336      BLOCKDATA BLK
337      COMMON QNORM(3,6),VERT(3,8),IEDGV1(12),IEDGV2(12),
338      . ISURF(6),IPEDG(24),NVERT,NEDGS,NSURF,F,SCALE(4)
339      DATA QNORM/0,0,-1.,-1.,4*0.,2*1.,3*0.,-1.,0,0,1.,0/
340      DATA VERT/4*0,1.,0,2*1.,0,1.,4*0,1.,0.,6*1.,0,1./
341      DATA IEDGV1/1,2,3,4,5,6,7,8,1,2,3,4/
342      DATA IEDGV2/2,3,4,1,6,7,8,5,5,6,7,8/
343      DATA ISURF/6*4/
344      DATA IPEDG/1,2,3,4,1,9,5,10,8,7,6,5,3,11,7,12,4,12,8,9,
345      . 2,10,6,11/
346      END
```



```
347      SUBROUTINE GRAFIT1(ID)
348 C
349      DIMENSION X(100), Y(100)
350      CHARACTER*31 YAXIS/' IGNITION TIME (MINUTES)¶.'/
351 C
352      CALL NEWFRM
353 C      CALL SETDEV(0,5)
354      CALL COLOR(2)
355      CALL GRISSET (0., -50., 1023., 1023.)
356      PIP = 3.1415
357      PIM = - PIP
358      CALL COLOR(3)
359      CALL GRILAB(1,PIM,-2.,PIP,+2.)
360      CALL GRAFIT (1,PIM, PIP, 200., 900., PIM, PIP, -2.0, +2.0,
361 .      100., 800., -2.0, +2.0, 'X AXIS¶.', 3, YAXIS, 5)
362      CALL COLOR(4)
363      CALL GRILAB(2,PIM,-2.,PIP,+2.)
364      CALL GRAFIT (2,PIM, PIP, 450., 650., PIM, PIP, -2.0, +2.0,
365 .      200., 500., -2.0, +2.0, 'X AXIS¶.', 3, 'Y AXIS¶.', 5)
366      XOFF = (PIP - PIM) / 100.
367      DO 2 I = 1, 100
368      XI = PIM + FLOAT(I-1) * XOFF
369      YI = COS(XI)
370      X(I) = XI
371      2 Y(I) = YI
372      CALL COLOR(1)
373      CALL PLOTLN (1, X, Y, 100)
374      CALL PLOTLN (2, X, Y, 100)
375      CALL FRAME
376      IF(ID.EQ.3) PAUSE
377      CALL ENDFRM
378      STOP
379      END
```

```

380     SUBROUTINE GRAFIT2(ID)
381     DIMENSION X(100),Y(100)
382 C
383 C     READ IN SOME DATA (X VS Y)
384 C
385     LIN = 3
386     OPEN(UNIT=1,FILE='G1TEST2.DAT')
387     CALL RDCNL(Y,I1,IY,I2)
388     CALL RDCNL(X,I1,IX,I2)
389     IZ = MINO(IX, IY)
390     I3 = IZ/3
391     3 IREV = 0
392     DO 4 I = 1, IZ-1
393     II = I + 1
394     IF(X(I).LE.X(II)) GO TO 4
395     IREV = IREV + 1
396     XP = X(I)
397     YP = Y(I)
398     X(I) = X(II)
399     Y(I) = Y(II)
400     X(II) = XP
401     Y(II) = YP
402     4 CONTINUE
403     IF (IREV.GT.0) GO TO 3
404 C
405 C     SCALE - WE NOW KNOW HOW BIG TO MAKE THE AXES
406 C
407     XMAX = 0.0
408     XMIN = 1.E+10
409     YMAX = 0.0
410     YMIN = 1.E+10
411     DO 1 I = 1, IZ
412     XMAX = AMAX1(XMAX,X(I))
413     XMIN = AMIN1(XMIN,X(I))
414     YMAX = AMAX1(YMAX,Y(I))
415     1 YMIN = AMIN1(YMIN,Y(I))
416     YMINC = 1.E+10
417     YMAXC = 0.0
418     DO 5 I = 1, I3
419     YMINC = AMIN1(YMINC,Y(I3-1+I))
420     5 YMAXC = AMAX1(YMAXC,Y(I3-1+I))
421 C
422 C     NOW ROUND OFF THE AXES TO APPROPRIATE WHOLE VALUES
423 C
424     YMIN = 0.0
425     DELTAX = (XMAX-XMIN)/9.
426     DELTAY = (YMAX-YMIN)/9.
427     XMAXA = XMAX - DELTAX * 0.8
428     XMINA = XMIN + DELTAX * 1.5
429     YMINA = YMIN + DELTAY * 1.5
430     YMAXA = YMAX - DELTAY * .8
431     XMINB = XMINA + DELTAX
432     XMAXB = XMAXA - 2.0*DELTAX
433     YMINB = YMINA + DELTAY * .76
434     YMAXB = YMAXA - 2.7*DELTAY

```

```
435     XMAX = IFIX(XMAX+DELTAX/2.)
436     YMAX = IFIX((YMAX+DELTAY)/100.)*100.
437     CALL NEWFRM
438     CALL LINWID(LIN)
439     CALL COLOR(1)
440     CALL GRISET (XMIN, YMIN, XMAX, YMAX)
441     CALL GRILAB(1, XMIN, YMIN, XMAX, YMAX)
442     CALL GRAFIT(1, XMIN, XMAX, XMINA, XMAXA, XMIN, XMAX, YMIN, YMAX,
443     . YMINA, YMAXA, YMIN, YMAX, ' TIME(MINUTES)¶.', 14,
444     . 'TEMPERATURE(K)¶.', 12)
445     CALL GRILAB(2, X(I3), YMINC, X(2*I3), YMAXC)
446     CALL GRAFIT(2, X(I3), X(2*I3), XMINB, XMAXB, X(I3), X(2*I3),
447     . YMINC, YMAXC, YMINB, YMAXB, YMINC, YMAXC,
448     . 'TIME(MINUTES)¶.', 5, 'FLUE TEMPERATURE(K)¶.', 5)
449     CALL COLOR(2)
450     CALL LINWID(4)
451     CALL PLOTLN(1, X, Y, I3)
452     CALL COLOR(3)
453     CALL PLOTLN(1, X(I3-1), Y(I3-1), I3)
454     CALL PLOTLN(2, X(I3), Y(I3), I3)
455     CALL COLOR(2)
456     CALL PLOTLN(1, X(2*(I3-1)), Y(2*(I3-1)), IZ-2*(I3-1))
457     CALL FRAME
458     IF(ID.EQ.3) PAUSE
459     CALL ENDFRM
460     STOP
461     END
```

```

462     SUBROUTINE RDCNL (REED, ICNL, MAXR, IEOF)
463   C
464   C     RRRRRRR   RRRRRRR   RRRRRR   RR   RR   RR
465   C     RRRRRRRR  RRRRRRRR  RRRRRRRR  RRR   RR   RR
466   C     RR   RR  RR   RR  RR   RR   RRRR  RR   RR
467   C     RR   RR  RR   RR  RR           RR RR RR   RR
468   C     RRRRRRR   RR   RR  RR   RR  RR   RRRR  RR
469   C     RR   RR  RRRRRRRR  RRRRRRRR  RR   RRR  RRRRRRRR
470   C     RR   RR  RRRRRRR   RRRRRR   RR   RR   RRRRRRRR
471   C
472     DIMENSION REED(100)
473     READ (1,10) MAXR, ICNL, IEND
474     IF (IEND.EQ.999) THEN
475         IEOF=1
476     ELSE
477         IEOF=0
478         READ (1,20) (REED(IR), IR=1, MAXR)
479     END IF
480     RETURN
481   C
482   C
483   10  FORMAT (2I6,T78,I3)
484   20  FORMAT (7E11.5)
485   C
486     END

```

```

487      SUBROUTINE GRAFIT3(ID)
488 C
489      DIMENSION X(100), Y(100)
490      CHARACTER*31 YAXIS/'TIME SINCE IGNITION(.MINUTES).'/
491      CHARACTER IC/'Z'/
492 C
493      CALL SETDEV(0,5)
494      XMIN = 0.
495      XMAX = 1000.
496      YMIN = 0.
497      YMAX = 1000.
498      XMAXG = 0.0
499      YMAXG = 0.0
500      DO 3 J = 1, 5
501      CALL NEWFRM
502      XMAXG = XMAXG + XMAX
503      YMAXG = YMAXG + YMAX
504      CALL GRISSET(XMIN, YMIN, XMAXG, YMAXG)
505      PIP = 3.1415
506      PIM = - PIP
507      CALL COLOR(3)
508      DX = XMAX - XMIN
509      DY = YMAX - YMIN
510      X200 = XMIN + .2 * DX
511      X900 = XMAX - .1 * DX
512      Y100 = YMIN + .2 * DY
513      Y800 = YMAX - .1 * DY
514      CALL GRILAB(1,PIM,-2.,PIP,+2.)
515      CALL GRAFIT (1,PIM, PIP, X200, X900, PIM, PIP, -2.0, +2.0,
516      . Y100, Y800, -2.0, +2.0, 'X AXIS.', 3, YAXIS, 5)
517      CALL COLOR(4)
518      X450 = XMIN + .45 * DX
519      X650 = XMAX - .35 * DX
520      Y200 = YMIN + .30 * DY
521      Y500 = YMAX - .40 * DY
522      CALL GRILAB(2,PIM,-2.,PIP,+2.)
523      CALL GRAFIT (2,PIM, PIP, X450, X650, PIM, PIP, -2.0, +2.0,
524      . Y200, Y500, -2.0, +2.0, 'X AXIS.', 3, 'Y AXIS.', 5)
525      XOFF = (PIP - PIM) / 10.
526      DO 2 I = 1, 10
527      XI = PIM + FLOAT(I-1) * XOFF
528      YI = COS(XI)
529      X(I) = XI
530      2 Y(I) = YI
531      CALL COLOR(1)
532      CALL PLOTLN (1, X, Y, 10)
533      CALL CHRISZ(0., .05)
534      CALL PLOTCH (1,X,Y,10,'*A')
535      CALL PLOTLN (2, X, Y, 10)
536      CALL PLOTCH (2,X,Y,10,IC)
537      CALL FRAME
538      IF(ID.EQ.3) PAUSE
539      3 CONTINUE
540      CALL ENDFRM
541      STOP

```

542

END

```
543     SUBROUTINE GRAFIT4(ID)
544     DIMENSION Q(500),VS(500),X(500),Y(500),TM(500),SQT(500)
545     CHARACTER*16 FILE
546     CHARACTER*40 TITLE/'GRAFIT(4)'/
547 C
548     FILE = 'GITEST4.DAT'
549     TSTAR = 10.
550     B = .321
551     OPEN (UNIT=8,FILE=FILE)
552     NP=0
553     N1=1
554     6 READ(8,3,END=5) N
555     3 FORMAT(I2)
556     READ(8,4) (VS(J),Q(J),TM(J),J=1,N)
557     DO 10 I=1,N
558     FT=B*SQRT(TM(I))
559     IF(TM(I).GT.TSTAR) FM=1.
560     10 SQT(I)=Q(I)*FT
561     4 FORMAT(50X,3F10.2)
562     N2=NP+(N-2)
563     J=1
564     DO 8, I=N1,N2
565     Y(I)=VS(J)
566     X(I)=SQT(J)
567     8 J=J+1
568     NP=N2
569     N1=NP+1
570     GO TO 6
571     5 CONTINUE
572     TYPE *, TITLE
573     NP=N2
574     CALL PLT(NP,X,Y)
575     IF(ID.EQ.1) GO TO 50
576     PAUSE
577     50 CONTINUE
578     CALL ENDFRM
579     STOP
580     END
581 C
```

```
582 SUBROUTINE PLT(N,X,Y)
583 DIMENSION X(550),Y(550)
584 CHARACTER*40 TITL
585 CHARACTER*25 YLABL
586 CHARACTER*25 XLABL
587 C
588 TITL = ' Rigid Foam¶.'
589 XLABL='q . F(t) (W/cm¶U2¶0)¶.'
590 YLABL='1/ V (s/mm)¶.'
591 ymax = 2.0
592 IYTIC = 10
593 CALL NEWFRM
594 CALL COLOR(1)
595 CALL GRISSET(0.,0.,1100.,1100.)
596 CALL COLOR(8)
597 CALL GRAFIT(1,0.,6.,200.,900.,0.,6.,0.,YMAX,200.,900.,0.,YMAX,
598 &XLABL,6,YLABL,IYTIC)
599 CALL COLOR(1)
600 CALL PLOTCH(1,X,Y,N,'o')
601 CALL COLOR(5)
602 CALL LABEL(TITL,250.,950.,850.,1000.,0.0)
603 CALL FRAME
604 RETURN
605 END
```



```

606     SUBROUTINE PHASE
607     DIMENSION Z(30,30), X(8),Y(8), FL(8), TEST(30,30)
608     DATA A/1./, B/-4./, C/5./, NINT/8/
609     DATA X/100., 800., 250., 950., 100., 800., 250., 950./
610     DATA Y/100., 100., 300., 300., 800., 800., 950., 950./
611 C
612     DO 1 I = 1,30
613     DO 1 J = 1, 30
614 1     Z(I,J) = A*COS(-.4+FLOAT(I)/5.)*
615     . (1.+B*EXP(-C*(FLOAT(J-10)/7.))**2))+5
616     ZMAX = -1.E+9
617     ZMIN = +1.E+9
618     DO 4 I = 1, 30
619     DO 4 J = 1, 30
620     ZMAX = MAX(ZMAX, Z(I,J))
621 4     ZMIN = MIN(ZMIN, Z(I,J))
622     CALL LINWID(1)
623     CALL NEWFRM
624     CALL DEFINE(0.,-100.,1000.,1000.)
625     CALL SRFSET(X,Y,0.,10.,30,30)
626     CALL COLOR(2)
627     CALL SURFAC(Z,30,30,1)
628     CALL COLOR(3)
629     CALL SURFAC(Z,30,30,-1)
630 C
631 C     SET UP A REASONABLE CONTOURING INTERVAL
632 C
633     CALL CNTSET(X(1),Y(1),X(2),Y(2),X(4),Y(4),X(3),Y(3))
634     DELTAZ = (ZMAX - ZMIN) * 0.9 / FLOAT (NINT)
635     FL(1) = ZMIN + 0.5 * DELTAZ
636     DO 2 I = 2, 8
637 2     FL(I) = FL(I-1) + DELTAZ
638     DO 3 I = 1, 8
639     CALL COLOR(I)
640 3     CALL CONTUR (Z, TEST, 30, 30, FL(I))
641     CALL FRAME
642     PAUSE
643     CALL ENDFRM
644     STOP
645     END

```

```
646      SUBROUTINE RUMTST
647      COMMON/ROOMSZ/RML,RMR,RMB,RMT,SCALE(4),WALLSZ,CEILSZ
648      REAL*4 LSIZE, DELTA
649      CALL NEWFRM
650      CALL FILTYP(2)
651      CALL ROOM (0.0, 5.0, 0.0, 2.5, .05, 1.0, 0.7)
652      CALL VENTV (0.1, 0.5, 0.0, -1.)
653      A = -1.5 / 20.
654      B = 2.2 - A
655      I = 1
656      FSIZE = FLOAT(I)*A + B
657      LSIZE = FSIZE + 0.2
658      PSIZE = FSIZE
659      CALL FIRE (4.0, 0.5, 0.2, FSIZE)
660      CALL PLUME (4.0, 0.5, 0.5, PSIZE, 0.2)
661      CALL LAYER (0.05, 2.45, 4.95, 2.48)
662      CALL FRAME
663      CALL IOWAIT(1000)
664      DELTA = -A
665      DO 101 I = 1, 21
666      CALL FIRE (4.0, 0.5, 0.2, FSIZE)
667      CALL PLUME (4.0, 0.5, 0.5, PSIZE, 0.2)
668      FSIZE = FLOAT(I)*A + B
669      LSIZE = FSIZE + 0.2
670      PSIZE = FSIZE
671      CALL FIRE (4.0, 0.5, 0.2, FSIZE-2*DELTA)
672      CALL PLUME (4.0, 0.5, 0.5, PSIZE-DELTA, 0.2)
673      CALL LAYER(0.05, LSIZE-DELTA, 4.95, 2.48)
674      CALL FRAME
675      CALL IOWAIT(1000)
676 101  CONTINUE
677      PAUSE
678      CALL ENDFRM
679      STOP
680      END
```

```
681      SUBROUTINE VENTV (BOTM, TOP, SIDE, THICK)
682      COMMON/ROOMSZ/RML,RMR,RMB,RMT,SCALE(4),WALLSZ,CEILSZ
683      REAL BOTM, TOP, SIDE
684      C
685      C      DRAW VERTICAL LINES IN BLACK AND HORIZONTAL LINES IN WHITE
686      C
687          Y1 = BOTM
688          Y2 = TOP
689          DO 2 I = 1, 2
690              X1 = SIDE + (I-1) * WALLSZ * THICK
691              X2 = X1
692          2  CALL LINE (X1, Y1, X2, Y2)
693              X1 = SIDE
694              X2 = WALLSZ * THICK
695          DO 1 I = 1, 2
696              Y1 = BOTM + (I-1) * (TOP-BOTM)
697              Y2 = Y1
698          1  CALL LINE (X1, Y1, X2, Y2)
699      RETURN
700      END
```

```
701          SUBROUTINE LAYER (LEFT, BOTTOM, RIGHT, TOP)
702 C
703 C      DRAW SQUIGGLE FROM LEFT TO RIGHT TO REPRESENT
704 C      THE LAYER (HOT) ABOVE THE PLUME
705 C
706          INTEGER DIVCNT
707          REAL LEFT, RIGHT, BOTTOM, TOP, X(4), Y(4)
708 C
709 C      NOW DRAW THE SQUIGGLE
710 C
711          CALL COLOR(2)
712          X(1) = LEFT
713          Y(1) = BOTTOM
714          X(2) = RIGHT
715          Y(2) = TOP
716          X(3) = LEFT
717          Y(3) = BOTTOM
718          X(4) = RIGHT
719          Y(4) = TOP
720          CALL PLYGON(X, Y, 4)
721          RETURN
722          END
```

```
723     SUBROUTINE FIRE (CENTER, WIDTH, HEIGHT, FSIZE)
724     COMMON/ROOMSZ/RML,RMR,RMB,RMT,SCALE(4),WALLSZ,CEILSZ
725     C
726     C PLOT THE FIRE SOURCE
727     C
728         CALL COLOR(3)
729     CALL VBXPLT (CENTER-.5*WIDTH, 0.0,
730     .           CENTER+.5*WIDTH,HEIGHT)
731     C
732     C START AT HEIGHT [HP=(1+EPS)*HEIGHT] AND TRACE THE PARABOLA
733     C
734     CALL COLOR(8)
735     HP = 1.2 * HEIGHT
736     W5 = WIDTH * .5
737     SLOPE = FSIZE / W5**2
738     W20 = WIDTH / 20.
739     YB = HP
740     XB = CENTER - W5
741     DO 1 I = 1, 21
742     XA = XB
743     YA = YB
744     XB = CENTER - W5 + W20* FLOAT(I-1)
745     YB = -(XB-CENTER)**2*SLOPE + HP + FSIZE
746     1 CALL LINE (XA, YA, XB, YB)
747     RETURN
748     END
```

```
749          SUBROUTINE PLUME (CENTER, WIDTH, PBOTM, PTOP, FSOURC)
750 C
751 C      TO PUT THE PLUME IN, NORMALLY ABOVE THE FIRE
752 C      THIS ROUTINE ASSUMES A POINT SOURCE PLUME AT A
753 C      VIRTUAL DISTANCE BELOW THE FIRE SUCH THAT THE
754 C      PLUME SUBTENDS AN ANGLE OF 11 DEGREES AT THE FIRE
755 C      SOURCE.
756 C
757          REAL CENTER, WIDTH, PBOTM, PTOP, FSOURC
758          COMMON/ROOMSZ/RML,RMR,RMB,RMT,SCALE(4),WALLSZ,CEILSZ
759 C
760 C      PLOT THE PLUME
761 C
762          RADIUS = 0.5 * WIDTH
763          THETA = 11. / 57.1
764 C      VIRTUAL = RADIUS / ATAN(THETA)
765          RTAN = ATAN (THETA)
766 C
767 C      DRAW TWO LINES FROM THE VIRTUAL POINT WITH AN ANGLE
768 C      OF THETA AND VISIBLE SEGMENTS
769 C      FROM PBOTM TO PTOP
770 C
771          CALL COLOR(7)
772          Y1 = PBOTM
773          Y2 = PTOP
774          DO 1 I = 1, 2
775          PM = (-1.0)**I
776          X1 = CENTER + PM*(RADIUS+(PBOTM-FSOURC)*RTAN)
777          X2 = CENTER + PM*(RADIUS+(PTOP -FSOURC)*RTAN)
778 1      CALL LINE (X1, Y1, X2, Y2)
779          RETURN
780          END
```

```
781          SUBROUTINE ROOM (L, R, B, T, DW, FW, FH)
782          REAL L, R, T, B, F, SCALE(4)
783          COMMON/ROOMSZ/RML,RMR,RMB,RMT,SCALE,WALLSZ,CEILSZ
784 C
785 C - L = LEFT SIDE OF ROOM
786 C - R = RIGHT SIDE OF ROOM
787 C - T = TOP OF THE ROOM
788 C - B = BOTTOM OF THE ROOM
789 C - DW = WIDTH OF WALL
790 C - F IS THE FRACTION OF THE SCREEN TO BE USED
791 C
792          XFW = AMIN1 (1., FW)
793          XFH = AMIN1 (1., FH)
794          DWMIN = AMIN1 (R-L, T-B)
795          WALLSZ = DWMIN * DW
796          CEILSZ = 0.5 * XFW/XFH * WALLSZ
797          DY = T - B
798          DX = R - L
799          XL = L - DX/3.5
800          XR = R + DX/3.5
801          XB = B - DY/3.5
802          XT = T + DY/3.5
803          CALL DEFINE (XL, XB, XR, XT)
804          CALL COLOR(1)
805          DO 1 IW = 1, 2
806          WALL = WALLSZ * (IW-1)
807          CEIL = CEILSZ * (IW-1)
808          XL = L - WALL
809          XR = R + WALL
810          XB = B
811          XT = T + CEIL
812          1 CALL BOXPLT (XL, XB, XR, XT)
813          RETURN
814          END
```

```

815          SUBROUTINE SADDLE(ID)
816 C
817 C          PROGRAM TO TEST THE SURFACE PLOTTING AND CONTOUR ROUTINES.
818 C
819          PARAMETER (NX=25,NY=13,NP=4)
820          REAL      Z(NX,NY), XP(8,9),YP(8,9), FL(16), TEST(NX,NY)
821          INTEGER   INDX(NP)
822          DATA     ZMIN, ZMAX /-0.75, 200./, MX, MY /1, 1/
823          DATA     INDX/1,3,7,9/
824          DATA     XP/
825          1         100., 800., 250., 900., 100., 800., 250., 900.,
826          2         100., 900., 200., 800., 100., 900., 200., 800.,
827          3         200., 900., 100., 750., 200., 900., 100., 750.,
828          4         100., 800., 250., 900., 100., 800., 250., 900.,
829          5         100., 900., 200., 800., 100., 900., 200., 800.,
830          6         200., 900., 100., 750., 200., 900., 100., 750.,
831          7         100., 800., 250., 900., 100., 800., 250., 900.,
832          8         100., 900., 200., 800., 100., 900., 200., 800.,
833          9         200., 900., 100., 750., 200., 900., 100., 750./
834          DATA     YP/
835          1         100., 100., 250., 250., 800., 800., 900., 900.,
836          2         100., 100., 250., 250., 800., 800., 900., 900.,
837          3         100., 100., 250., 250., 800., 800., 900., 900.,
838          4         100., 100., 200., 200., 900., 900., 800., 800.,
839          5         100., 100., 200., 200., 900., 900., 800., 800.,
840          6         100., 100., 200., 200., 900., 900., 800., 800.,
841          7         200., 200., 100., 100., 900., 900., 750., 750.,
842          8         200., 200., 100., 100., 900., 900., 750., 750.,
843          9         200., 200., 100., 100., 900., 900., 750., 750./
844 C
845 C          INITIALIZE THE DATA
846 C
847          DO 20 J = 1, NY
848          DO 20 I = 1, NX
849          Z(I,J) = ABS(FLOAT(J)-7.)**2 + (144.-ABS(FLOAT(I)-13.)**2)
850          20      CONTINUE
851          DO 2 I = 1, 8
852          FL(I) = -1.0 + FLOAT(I)*10.
853          2      FL(I+8) = 0.2 + FLOAT(I)*10.
854 C
855 C          INITIALIZE THE GRAPHICS PACKAGE
856 C
857          CALL DEFINE (0., 0., 1023., 1023.)
858          CALL NEWFRM
859 C
860 C          LOOP OVER THE VARIOUS PERSPECTIVE PLOTS.
861 C
862          IPP = 3
863          DO 100 IPP = 1, NP
864          IP = INDX(IPP)
865 C
866 C          PLOT THE UPPER PART OF THE SURFACE WITH FRONT SKIRT.
867 C
868          CALL SRFSET (XP(1,IP), YP(1,IP), ZMIN, ZMAX, NX, NY)
869          CALL ERASE

```



```
870      CALL SETLUT
871      CALL COLOR(5)
872      CALL SURFAC (Z, NX, NY, +1)
873      CALL COLOR(4)
874      CALL SFRAME (2)
875      CALL FRAME
876      CALL IOWAIT(2000)
877 C
878 C      PLOT THE LOWER PART OF THE SURFACE.
879 C
880      CALL SRFSET (XP(1,IP), YP(1,IP), ZMIN, ZMAX, NX, NY)
881      CALL ERASE
882      CALL COLOR(7)
883      CALL SURFAC (Z, NX, NY, -1)
884      CALL COLOR(4)
885      CALL SFRAME (2)
886      CALL FRAME
887      CALL IOWAIT(2000)
888 C
889 C      PLOT BOTH PARTS OF THE SURFACE WITH SIDE SKIRTS.
890 C
891      CALL SRFSET (XP(1,IP), YP(1,IP), ZMIN, ZMAX, NX, NY)
892      CALL ERASE
893      CALL COLOR(5)
894      CALL SURFAC (Z, NX, NY, +1)
895      CALL COLOR(7)
896      CALL SURFAC (Z, NX, NY, -1)
897      CALL SFRAME (2)
898      CALL FRAME
899      IF(ID.EQ.3) THEN
900          PAUSE
901      ELSE
902          CALL IOWAIT(2000)
903      ENDIF
904 100      CONTINUE
905      CALL ENDFRM
906 C
907      STOP
908      END
```

```
909      SUBROUTINE SETCOL(IC)
910      DIMENSION LUT(16,3)
911      C
912      DO 1 I = 1, 3
913      DO 1 J = 1, 16
914      1   LUT(J,I) = 0
915      DO 2 I = 1, 8
916      LUT(I+1,2) = I*2 - 1
917      2   LUT(I+1,3) = I*2 - 1
918      DO 3 I = 1, 7
919      LUT(I+9,1) = I*2 + 1
920      3   LUT(I+9,2) = I*2 + 1
921      CALL DSLWT (16, 48, LUT)
922      RETURN
923      END
```

```

924      SUBROUTINE SYMTST
925 C
926 C      A SIMPLE TEST OF THE CHARACTER STUFF
927 C
928      CHARACTER*28 TITLE1/'A SET OF HARDWARE CHARACTERS'/
929      CHARACTER*19 TITLE2/'q1 SET FROM SET 1q.'/
930      CHARACTER*32 TITLE3/'q:A SET ([$/*†]) FROM SET #10q.'/
931      ISC=0
932      3      XSZ = 11. * 2.**ISC - 11.
933      CALL NEWFRM
934      CALL DEFINE(-XSZ, -XSZ, 11.+XSZ, 11.+XSZ)
935      CALL COLOR(4)
936      CALL BOXPLT (0.5, 0.5, 8.5, 4.5)
937      CALL BOXPLT (0.5, 5.5, 8.5, 9.5)
938      CALL BOXPLT (9.0, 0.5, 10.7, 9.5)
939      DX = 5. / 28.
940      XL = 4.5 - 14.*DX
941      CALL COLOR(1)
942      CALL CHRSTZ (DX/(11.+2.*XSZ),0.)
943      DO 1 I = 1, 28
944      1      CALL SYMBOL (XL+DX*FLOAT(I), 7.5, TITLE1(I:I))
945      CALL COLOR(3)
946      CALL LABEL(TITLE2, 2.0, 2.0, 8.0, 2.5, 0.0)
947      CALL COLOR(2)
948      CALL LABEL(TITLE3, 10.3, 1., 17.3, 2., 3.14/2.)
949      CALL FRAME
950      CALL IOWAIT(2000)
951      ISC = ISC + 1
952      IF(ID.NE.3) GO TO 4
953      IF(ISC.GT.7) GO TO 4
954      GO TO 3
955      4      CALL ENDFRM
956      STOP
957      END

```

```

958      SUBROUTINE VOLTST(ID)
959 C
960 C      VOLUME TEST PROGRAM WITH REDUCED RESOLUTION HIDDEN LINES.
961 C
962 C      THIS PROGRAM DEMONSTRATES THE USE OF THE "VOLUME" PLOTTER.
963 C      IT USES A SIMPLE SOLID ( TWO OVERLAPPING SPHERES)
964 C      WITH A SOLID PILLAR PLACED IN FRONT TO OBSCURE PART OF THE
965 C      SPHERES AND DEMONSTRATE THE HIDDEN LINE FEATURE. THIS IS DONE
966 C      TOGETHER WITH THE NECESSARY INTERACTION WITH THE GRAPHICS
967 C      PACKAGE. THIS SHOULD BE USED AS A MODEL FOR DESIGNING PROGRAMS UN
968 C      THE USER IS FAMILIAR WITH VOLUME.
969 C
970      PARAMETER (NT=100, NT2=NT*NT)
971      REAL T(NT2), CLEVE(2)
972      REAL      XP(8,3), YP(8,3)
973      REAL      F(20, 20, 20)
974      INTEGER  MODE(2)
975      DATA     NX,NY,NZ /20, 20, 20/
976      DATA     MODE /-1, 2/, CLEVE /1.5, -1.0/
977      DATA     NP, ITAPE /3, 105/
978      DATA     XP/
979      1          100., 800., 250., 900., 100., 800., 250., 900.,
980      2          100., 900., 200., 800., 100., 900., 200., 800.,
981      3          200., 900., 100., 750., 200., 900., 100., 750./
982      DATA     YP/
983      1          100., 100., 250., 250., 800., 800., 900., 900.,
984      2          100., 100., 250., 250., 800., 800., 900., 900.,
985      3          100., 100., 250., 250., 800., 800., 900., 900./
986 C
987 C      INITIALIZE THE PACKAGE
988 C
989      CALL DEFINE(0., 0., 1024., 1024.)
990 C
991 C      LOOP OVER FOUR SIZES OF THE SPHERES
992 C
993      DO 100 IPIC = 1,901,300
994      NP = NP
995      IP = 1
996      FAC = 1.0 + 0.01*FLOAT(IPIC)
997 C
998 C      SET UP THE VALUE OF THE FUNCTION.
999 C      IT IS AN NRL PEANUT ( 2 OVERLAPPING SPHERES)
1000 C      (X-13)**2 + (Y-10.5)**2 + (Z-10.5)**2
1001 C      (X-8)**2 + (Y-10.5)**2 + (Z-10.5)**2
1002 C
1003      DO 1 I = 1, NX
1004      DO 1 J = 1, NY
1005      DO 1 K = 1, NZ
1006      R1SQ = (I-8.0)**2 + (J-10.5)**2 + (K-10.5)**2
1007      R2SQ = (I-13.)**2 + (J-10.5)**2 + (K-10.5)**2
1008      F(I,J,K) = FAC/R1SQ + 2.0*FAC/R2SQ
1009      1 CONTINUE
1010 C
1011 C      CONSTRUCT THE PILLAR
1012 C

```

```
1013      DO 2 I = 1, NX
1014      DO 2 J = 1, 5
1015      DO 2 K = 1, NZ
1016      RSQ = (I-10.5)**2 + (J-3.5)**2
1017      F(I,J,K) = -4.0/RSQ
1018      2    CONTINUE
1019      C
1020      C    INITIALIZE THE PLOTTING PACKAGE
1021      C
1022      C    IN THIS CASE, "VOLSET" IS CALLED FOR EACH CYCLE SO THAT THE
1023      C    VARIOUS CONTOUR LEVELS WILL BE PLOTTED.  THE PILLAR IS PLOTTED
1024      C    ONLY ONCE(THIRD CALL) BUT STUFF BEHIND IT IS ALWAYS HIDDEN
1025      C    (NCL=2, MODE=2, CLEVE=-1.).
1026      C
1027      CALL NEWFRM
1028      C
1029      C    DRAW THE FIGURE
1030      C
1031      CALL COLOR(1)
1032      CLEVE(1) = 1.5
1033      CALL VOLSET (XP(1,IP), YP(1,IP), T, NT)
1034      CALL VOLUME(MODE, F, NX,NY,NZ, CLEVE, 2, T, NT)
1035      C
1036      CALL COLOR(2)
1037      CLEVE(1) = 0.5
1038      CALL VOLSET (XP(1,IP), YP(1,IP), T, NT)
1039      CALL VOLUME(MODE, F, NX,NY,NZ, CLEVE, 2, T, NT)
1040      C
1041      CALL COLOR(3)
1042      CALL VOLSET (XP(1,IP), YP(1,IP), T, NT)
1043      CALL VOLUME (+1, F, NX,NY,NZ, -1.0, 1, T, NT)
1044      C
1045      CALL COLOR(4)
1046      CALL VOLUME (-2, F, NX,NY,NZ, 0.5, 1, T, NT)
1047      C    CALL VOLFRM (2)
1048      CALL FRAME
1049      CALL IOWAIT(2000)
1050      C
1051      100  CONTINUE
1052      C
1053      C    CLOSE-OUT GRAFIT.
1054      C
1055      IF(ID.EQ.3) PAUSE
1056      CALL ENDFRM
1057      C
1058      STOP
1059      END
```

```
1060     SUBROUTINE FILNGT
1061 C
1062     CALL NEWFRM
1063     CALL DEFINE(-11.,-11.,11.,11.)
1064     DO 2 I = 1, 9
1065     CALL FILTYP(I-1)
1066     X = -11. + FLOAT(I)*2.1
1067     Y = -11. + FLOAT(I)*2.1
1068     CALL CIRCLE(X, Y, 2.0)
1069 2     CONTINUE
1070     CALL FRAME
1071     PAUSE
1072     CALL ENDFRM
1073     STOP
1074     END
```

SUBROUTINE ALABEL	page	1
FUNCTION AND	page	2
SUBROUTINE BOXPLT	page	3
ENTRY VBXPLT	page	3
SUBROUTINE CHPLOT	page	4
SUBROUTINE CIRCLE	page	5
SUBROUTINE CONTUR	page	6
SUBROUTINE DEVICE	page	10
ENTRY SETDEV	page	12
ENTRY NEWFRM	page	12
ENTRY FRAME	page	12
ENTRY ERASE	page	12
ENTRY HDCOPY	page	12
ENTRY COLOR	page	12
ENTRY FILTYP	page	12
ENTRY CHRSTZ	page	13
ENTRY LINWID	page	13
ENTRY SCALNG	page	13
ENTRY DEFINE	page	13
ENTRY ENDFRM	page	14
ENTRY DELAY	page	20
ENTRY IOWAIT	page	20
ENTRY LEXDID	page	20
ENTRY TABLET	page	20
SUBROUTINE DEVINP	page	22
SUBROUTINE ENUMBR	page	24
SUBROUTINE FNUMBR	page	25
SUBROUTINE GRAFIT	page	28
ENTRY PLOTCH	page	31
ENTRY PLOTLN	page	31
ENTRY PLOTBAR	page	32
ENTRY GRISET	page	32
ENTRY GRILAB	page	32
SUBROUTINE HHDRAW	page	34
SUBROUTINE HSETS	page	36
INTEGER FUNCTION IDCHAR	page	38
ENTRY IREVCH	page	38
SUBROUTINE LABEL	page	39

SUBROUTINE LINE	page	40
SUBROUTINE LINES	page	41
SUBROUTINE LNPLOT	page	42
INTEGER FUNCTION LSHF	page	43
SUBROUTINE MAPIN	page	44
SUBROUTINE MAPOUT	page	45
SUBROUTINE PLYGON	page	46
SUBROUTINE PLYPL1	page	47
SUBROUTINE PLYPL2	page	49
SUBROUTINE PLYPL3	page	50
SUBROUTINE PLYPL4	page	52
SUBROUTINE PLYPL5	page	54
SUBROUTINE PLYPLT	page	56
ENTRY PLYGNS	page	59
SUBROUTINE PROPOL	page	61
ENTRY CNTSET	page	63
ENTRY CNTFRM	page	63
SUBROUTINE PUTCH	page	64
SUBROUTINE PUTLNV	page	66
SUBROUTINE SETLUT	page	68
SUBROUTINE SRFSET	page	69
ENTRY SFRAME	page	71
ENTRY SSKIRT	page	71
ENTRY SURFAC	page	73
SUBROUTINE SURF4	page	77
ENTRY SURF5	page	77
SUBROUTINE SYMBOL	page	78
SUBROUTINE VIEWTR	page	79
SUBROUTINE VOLFRM	page	83

SUBROUTINE VOLUMO	page 84
ENTRY VOLUM6	page 84
SUBROUTINE VOLUM7	page 85
SUBROUTINE VOLUM8	page 87
SUBROUTINE VOLUME	page 89
ENTRY VOLSET	page 92
SUBROUTINE WDCOUNT	page 94
SUBROUTINE WDDRAW	page 96 .
ENTRY CHRSET	page 99

```
1      SUBROUTINE ALABEL (CHARS, XL, XR, ANGLE)
2      COMMON /DEVTYP/ IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
3      COMMON/GRFTYP/XNGLE,IRVRSE,CHSIZE(9),ITKWIT,LUDIAG,LUHSET
4      CHARACTER*1 CHARS(*)
5      CALL WDCOUNT(CHARS, NC)
6      IF(NC.LE.0) THEN
7          IF(LUDIAG.GT.0) WRITE(LUDIAG,3) (CHARS(I),I=1,132)
8      3      FORMAT(' NO ESCAPE SEQUENCE IN ALABEL -',/,1X,132A1)
9          RETURN
10         ENDIF
11         DX = (XR - XL) / MAX(1.,FLOAT(NC))
12         DY = 4./3. * DX * XYCOORD(1)/XYCOORD(3)
13         XNGLE = ANGLE
14         CALL WDDRAW (XL, YB, DX, 0.0, DX, 0.0, DY, CHARS)
15         RETURN
16     END
```

```
23         SUBROUTINE BOXPLT (X1, Y1, X2, Y2)
24 C
25 C*****
26 C     BOXPLT
27 C*****
28 C
29         REAL XA(4),XB(4),YA(4),YB(4)
30 C
31         ENTRY VBXPLT (X1, Y1, X2, Y2)
32         XA(1)=X1
33         YA(1)=Y1
34         XB(1)=X2
35         YB(1)=Y1
36         XA(2)=X2
37         YA(2)=Y1
38         XB(2)=X2
39         YB(2)=Y2
40         XA(3)=X2
41         YA(3)=Y2
42         XB(3)=X1
43         YB(3)=Y2
44         XA(4)=X1
45         YA(4)=Y2
46         XB(4)=X1
47         YB(4)=Y1
48         CALL LINES(XA, YA, XB, YB, 4)
49         RETURN
50         END
```

```

51     SUBROUTINE CHPLOT (XARRAY, YARRAY, CHARAC, I1, I2, I3)
52     C
53     C     XARRAY REAL      - AN ARRAY, DIMENSIONED N, OF X COORDINATES OF
54     C                       THE DATA (MATH SPACE)
55     C     YARRAY REAL      - AN ARRAY, DIMENSIONED N, OF Y COORDINATES OF
56     C                       THE DATA (MATH SPACE)
57     C     CHARAC LITERAL - CHARACTER TO BE PLOTTED
58     C     I1     INTEGER - INDEX OF FIRST POINT TO BE PLOTTED
59     C     I2     INTEGER - INCREMENT AT WHICH POINTS ARE TO BE SELECTED
60     C                       TO PLOT
61     C     I3     INTEGER - INDEX OF LAST POINT IN DATA ARRAYS
62     C*****
63     C     CHPLOT
64     C*****
65     C
66     COMMON /DEV TYP/ IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
67     COMMON/GRFTYP/ANGLE,IRVRSE,CHSIZE(9),ITKWIT,LUDLAG,LUHSET
68     INTEGER I1, I2, I3, CHCODE
69     REAL XARRAY(4), YARRAY(4)
70     CHARACTER*1 CHARAC
71     INTEGER IX,IY
72     NOPTS = (I3-I1+I2) / I2
73     I1M = I1 - I2
74     DO 1 I=1,NOPTS
75     II=I1M+I2*I
76     1 CALL SYMBOL (XARRAY(II), YARRAY(II), CHARAC)
77     RETURN
78     END

```

```
79          SUBROUTINE CIRCLE(X, Y, R)
80 C
81 C      DRAW A CIRLE OF WIDTH LINWDM AND FILL WITH LFLMAT.
82 C
83          COMMON/DEV TYP/IDEVIC,LSW,LTSW,XYCOOD(4),LUOUT,LPAGE
84          COMMON/GRF MOD/LFLMAT(5),LINWDM(5),LCLMAT(5)
85          REAL XO, YO, XN, YN, DX, DY, THETA, ARC
86 C
87          IF(IDEVIC.EQ.0) RETURN
88          GO TO (1,1,2,1,1), IDEVIC
89      1      ARC = 3.1415/50.
90          THETA = 0.0
91          XO = X + R
92          YO = Y
93          DO 10 I = 1, 101
94          THETA = THETA + ARC
95          XN = R * COS(THETA) + X
96          YN = R * SIN(THETA) + Y
97          CALL LINE(XO, YO, XN, YN)
98          XO = XN
99          YO = YN
100      10      CONTINUE
101          RETURN
102      2      XL = MAX(X * XYCOOD(1) + XYCOOD(2),0.)
103          YL = MAX(Y * XYCOOD(3) + XYCOOD(4),0.)
104          RL = R * MIN(XYCOOD(1), XYCOOD(3))
105          CALL GMOVEA(XL, YL)
106          CALL GCIRA(RL)
107          RETURN
108          END
```

```

109     SUBROUTINE CONTUR (F, TEST, NX, NY, FL)
110         PARAMETER (NPT=101)
111         REAL      F(NX,NY), TEST(NX,NY)
112         REAL      XT1(NPT), XT2(NPT), XT3(NPT), YT1(NPT), YT2(NPT),
113         1        YT3(NPT), FT1(NPT), FT2(NPT), FT3(NPT)
114         REAL      T00(NPT), T10(NPT), T01(NPT), T11(NPT)
115     C
116     C     SUBARRAYS OUT OF THE DATA CANNOT BE CONTOURED. THE
117     C     ENTIRE F(I,J) ARRAY IS CONTOURED SO A COPY OVER INTO SCRATCH SPACE
118     C     IS REQUIRED IF ONLY A PORTION OF F IS TO BE PLOTTED.
119     C
120     C     F      REAL ARRAY (NX,NY)  THE REAL VALUES OF THE FUNCTION TO
121     C                                CONTOURED. A TOPOLOGICALLY SQUARE
122     C                                GRID CELL IS ASSUMED AND DX = DY = 1.
123     C     TEST  REAL ARRAY (NX,NY)  A USER SUPPLIED SCRATCH ARRAY OF
124     C                                THE SAME DIMENSIONALITY AS F.
125     C     NX    INTEGER                RANGE AND DIMENSION OF I IN F(I,J).
126     C     NY    INTEGER                RANGE AND DIMENSION OF J IN F(I,J).
127     C     FL    REAL                  THE VALUE OF F(I,J) FOR CONTOURING.
128     C
129     C     CALCULATE THE AVERAGE VALUE OF F AT THE CENTERS OF THE CELLS.
130     C
131         NTMAX = NPT - 2
132         NTRIA = 0
133         NXM = NX - 1
134         NYM = NY - 1
135         DO 200 J = 1, NYM
136         DO 200 I = 1, NXM
137             TEST(I,J) = F(I,J) + F(I+1,J)
138             TEST(I,J) = TEST(I,J) + F(I,J+1)
139             TEST(I,J) = TEST(I,J) + F(I+1,J+1)
140     200     TEST(I,J) = 0.25*TEST(I,J)
141     C
142     C     NOW CALCULATE THE CROSSINGS WHICH PARELLEL THE "I" AXIS.
143     C
144         DO 310 J = 1, NY
145         DO 300 I = 1, NXM
146     300     T00(I) = (F(I+1,J) - FL)*(FL - F(I,J))
147     C
148     C     A CROSS OCCURS IF THE CONTOUR PASSES THRU F(I,J) IN THE BOX I->I+1
149     C
150         DO 309 I = 1, NXM
151             IF (T00(I) .LT. 0.0) GO TO 309
152     C
153     C     THE LINE SEGMENT IS A HIT. TREAT IN A SCALAR WAY FIRST THE UPPER
154     C     AND THEN THE LOWER TRIANGLE.
155     C
156         IF (J.EQ.NY) GO TO 305
157         IF (NTRIA .GE. NTMAX) CALL PROPOL (NTRIA, XT1,YT1,FT1,
158     1        XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
159         NTRIA = NTRIA + 1
160         XT1(NTRIA) = FLOAT(I)
161         YT1(NTRIA) = FLOAT(J)
162         FT1(NTRIA) = F(I,J)
163         XT2(NTRIA) = FLOAT(I + 1)

```

```

164      YT2(NTRIA) = FLOAT(J)
165      FT2(NTRIA) = F(I+1,J)
166      XT3(NTRIA) = FLOAT(I) + 0.5
167      YT3(NTRIA) = FLOAT(J) + 0.5
168      FT3(NTRIA) = TEST(I,J)
169      305  IF (J.EQ.1) GO TO 309
170      IF (NTRIA .GE. NTMAX) CALL PROPOL (NTRIA, XT1,YT1,FT1,
171      1      XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
172      NTRIA = NTRIA + 1
173      XT1(NTRIA) = FLOAT(I + 1)
174      YT1(NTRIA) = FLOAT(J)
175      FT1(NTRIA) = F(I+1,J)
176      XT2(NTRIA) = FLOAT(I)
177      YT2(NTRIA) = FLOAT(J)
178      FT2(NTRIA) = F(I, J)
179      XT3(NTRIA) = FLOAT(I) + 0.5
180      YT3(NTRIA) = FLOAT(J) - 0.5
181      FT3(NTRIA) = TEST(I,J-1)
182      309  CONTINUE
183      310  CONTINUE
184  C
185  C      NEXT CALCULATE THE CROSSINGS ALONG THE "J" AXIS.
186  C
187      DO 360 J = 1, NYM
188      DO 350 I = 1, NX
189      350  T11(I) = (F(I,J+1) - FL)*(FL - F(I,J))
190  C
191  C      CORSS OCCURS IF THE CONTOUR VALUE PASSES THRU THE BOX IN F(I,J)
192  C      IN THE RANGE J->J+1
193  C
194      DO 359 I = 1, NX
195      IF (T11(I) .LT. 0.0) GO TO 359
196  C
197  C      THE LINE SEGMENT IS A HIT. TREAT IN A SCALAR WAY FIRST THE RIGHT
198  C      AND THEN THE LEFT TRIANGLE.
199  C
200      IF (I.EQ.NX) GO TO 355
201      IF (NTRIA .GE. NTMAX) CALL PROPOL (NTRIA, XT1,YT1,FT1,
202      1      XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
203      NTRIA = NTRIA + 1
204      XT1(NTRIA) = FLOAT(I)
205      YT1(NTRIA) = FLOAT(J + 1)
206      FT1(NTRIA) = F(I,J+1)
207      XT2(NTRIA) = FLOAT(I)
208      YT2(NTRIA) = FLOAT(J)
209      FT2(NTRIA) = F(I,J)
210      XT3(NTRIA) = FLOAT(I) + 0.5
211      YT3(NTRIA) = FLOAT(J) + 0.5
212      FT3(NTRIA) = TEST(I,J)
213      355  IF (I.EQ.1) GO TO 359
214      IF (NTRIA .GE. NTMAX) CALL PROPOL (NTRIA, XT1,YT1,FT1,
215      1      XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
216      NTRIA = NTRIA + 1
217      XT1(NTRIA) = FLOAT(I)
218      YT1(NTRIA) = FLOAT(J)

```

```

219          FT1(NTRIA) = F(I,J)
220          XT2(NTRIA) = FLOAT(I)
221          YT2(NTRIA) = FLOAT(J + 1)
222          FT2(NTRIA) = F(I,J+1)
223          XT3(NTRIA) = FLOAT(I) - 0.5
224          YT3(NTRIA) = FLOAT(J) + 0.5
225          FT3(NTRIA) = TEST(I-1,J)
226 359      CONTINUE
227 360      CONTINUE
228 C
229 C      NOW SEEK ALL TRIANGLES WITH TWO DIAGONAL CROSSINGS.
230 C
231          DO 490 J = 1, NYM
232          DO 420 I = 1, NXM
233          T00(I) = (TEST(I,J) - FL)*(FL - F(I,J))
234          T10(I) = (FL - TEST(I,J))*(F(I+1,J) - FL)
235          T01(I) = (FL - TEST(I,J))*(F(I,J+1) - FL)
236 420      T11(I) = (TEST(I,J) - FL)*(FL - F(I+1,J+1))
237 C
238 C      CONSIDER THE LOWER TRIANGLE IN THE SQUARE.
239 C
240          DO 430 I = 1, NXM
241          IF (AMIN1(T00(I), T10(I)) .LT. 0.0) GO TO 430
242          IF (NTRIA .GE. NTMAX) CALL PROPOL (NTRIA, XT1,YT1,FT1,
243 1          XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
244          NTRIA = NTRIA + 1
245          XT1(NTRIA) = FLOAT(I) + 0.5
246          YT1(NTRIA) = FLOAT(J) + 0.5
247          FT1(NTRIA) = TEST(I,J)
248          XT2(NTRIA) = FLOAT(I)
249          YT2(NTRIA) = FLOAT(J)
250          FT2(NTRIA) = F(I,J)
251          XT3(NTRIA) = FLOAT(I + 1)
252          YT3(NTRIA) = FLOAT(J)
253          FT3(NTRIA) = F(I+1,J)
254 430      CONTINUE
255 C
256 C      CONSIDER THE RIGHT TRIANGLE IN THE SQUARE.
257 C
258          DO 440 I = 1, NXM
259          IF (AMIN1(T10(I), T11(I)) .LT. 0.0) GO TO 440
260          IF (NTRIA .GE. NTMAX) CALL PROPOL (NTRIA, XT1,YT1,FT1,
261 1          XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
262          NTRIA = NTRIA + 1
263          XT1(NTRIA) = FLOAT(I) + 0.5
264          YT1(NTRIA) = FLOAT(J) + 0.5
265          FT1(NTRIA) = TEST(I,J)
266          XT2(NTRIA) = FLOAT(I + 1)
267          YT2(NTRIA) = FLOAT(J)
268          FT2(NTRIA) = F(I+1,J)
269          XT3(NTRIA) = FLOAT(I + 1)
270          YT3(NTRIA) = FLOAT(J + 1)
271          FT3(NTRIA) = F(I+1,J+1)
272 440      CONTINUE
273 C

```



```

274 C      CONSIDER THE UPPER TRIANGLE IN THE SQUARE.
275 C
276         DO 450 I = 1, NXM
277         IF (AMIN1(T11(I), T01(I)) .LT. 0.0) GO TO 450
278         IF (NTRIA .GE. NTMAX) CALL PROPOL (NTRIA, XT1,YT1,FT1,
279         1      XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
280         NTRIA = NTRIA + 1
281         XT1(NTRIA) = FLOAT(I) + 0.5
282         YT1(NTRIA) = FLOAT(J) + 0.5
283         FT1(NTRIA) = TEST(I,J)
284         XT2(NTRIA) = FLOAT(I + 1)
285         YT2(NTRIA) = FLOAT(J + 1)
286         FT2(NTRIA) = F(I+1,J+1)
287         XT3(NTRIA) = FLOAT(I)
288         YT3(NTRIA) = FLOAT(J + 1)
289         FT3(NTRIA) = F(I,J+1)
290         450 CONTINUE
291 C
292 C      CONSIDER THE LEFT TRIANGLE IN THE SQUARE.
293 C
294         DO 460 I = 1, NXM
295         IF (AMIN1(T01(I), T00(I)) .LT. 0.0) GO TO 460
296         IF (NTRIA .GE. NTMAX) CALL PROPOL (NTRIA, XT1,YT1,FT1,
297         1      XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
298         NTRIA = NTRIA + 1
299         XT1(NTRIA) = FLOAT(I) + 0.5
300         YT1(NTRIA) = FLOAT(J) + 0.5
301         FT1(NTRIA) = TEST(I,J)
302         XT2(NTRIA) = FLOAT(I)
303         YT2(NTRIA) = FLOAT(J + 1)
304         FT2(NTRIA) = F(I,J+1)
305         XT3(NTRIA) = FLOAT(I)
306         YT3(NTRIA) = FLOAT(J)
307         FT3(NTRIA) = F(I,J)
308         460 CONTINUE
309         490 CONTINUE
310 C
311 C      IF THE BUFFER IS ONLY PARTIALLY FULL, FLUSH AT THE END.
312 C
313         IF (NTRIA .GT. 0) CALL PROPOL (NTRIA, XT1,YT1,FT1,
314         1      XT2,YT2,FT2, XT3,YT3,FT3, FL, NX, NY)
315 C
316         RETURN
317         END

```

```

318          SUBROUTINE DEVICE (N)
319 C
320 C      DEVICE          1 CALCOMP PLOTTER
321 C                      2 PRINTER PLOT
322 C                      3 LEXIDATA DISPLAY UNIT WITH MATRIX CAMERA
323 C                      4 ADM WITH RETROGRAPHICS OR TEK 40XX SERIES
324 C                      5 EMPTY SLOT
325 C
326 C      COMMANDS          DEVICE - SELECT DEVICE (1-5)
327 C                      SETDEV - SET THE LOGICAL UNITS FRO OUTPUT
328 C                      LINWID - NUMBER OF STROKES IN A LINE
329 C                      FILTYP - TYPE OF FILLING FOR POLYGONS AND CIRCL
330 C                      CHRSTZ - HEIGHT OF CHARACTERS (IN RASTER UNITS)
331 C
332 C                      1 NEWFRM - SET UP DEVICE FOR A NEW PLOT
333 C                      2 FRAME - FORCE A FLUSH OF ALL BUFFERS
334 C                      3 ERASE - ERASE THE SCREEEN
335 C                      4 HDCOPY - HARD COPY COMMAND - SHUTTER ON MATRIX
336 C                      5 COLOR - SELECT THE COLOR TO DRAW
337 C                      6 SCALE - SCALE FOR PHYSICAL DEVICE
338 C                      7 ENDFRM - TERMINATE THE DEVICE AND ADVANCE
339 C
340 C      DEFAULT DEVICE ASSIGNMENTS
341 C
342 C          TYPE          LU      RESULT
343 C
344 C          DIAGNOSTICS    0      NO OUTPUT
345 C          GRAPHICS      7      NORMAL - ASSIGNED
346 C          CAMERA         8      ASSIGNED WITH LEXIDATA
347 C          CHARACTER SET  0      UNIT 0 - ASSIGNED
348 C          CALCOMP        L7:    RS232 LINE
349 C          PRINTER        PR:    WHATEVER
350 C          LEXIDATA       LEX:    DMA (L34DVR IN SYSTEM)
351 C          MATRIX CAMERA  LE:    RS232
352 C          TEKTRONIX      C:     CONSOLE
353 C          OPEN SLOT      NULL:  BIT BUCKET
354 C          CONSOLE INPUT  5
355 C          CHARACTER SET  9 (CLOSED THEN OPENED)
356 C          TABLET INPUT  8 (CLOSED THEN OPENED -
357 C                          CAMERA AND TABLET CAN
358 C                          NOT BE ACTIVE SIMULTANEOUSLY)
359 C
360 C      INTEGER LB, COUNT(5), NFRAME(5), PEN, COMMND, DEVID, PBLK(6)
361 C      CHARACTER*8 IDFLT0(5), MATFIL, TABFIL
362 C      INTEGER ICODET(2), ICTRLX(2), ICHANX(2,2), SCRATCH(60)
363 C      INTEGER HBLANK, HMINUS, HAPOSA, HAPOSB, HASTRA, HASTRB, NHOLD
364 C      INTEGER VSNUM(2), TEKRAS(4), LEXRAS(4), HWSIZE(20), CALRAS(4)
365 C      INTEGER LPAGE(30,65), START, LINE, SLASH, ENDIT
366 C      INTEGER MATUNT, TABUNT, USPAT(12,4)
367 C      LOGICAL LSW, LTSW, OPEN, OPENANY, OPNCAL, OPNLEX, IRVRSE
368 C      LOGICAL OPNMAT, OPNTAB, OPENTB
369 C      COMMON/DEV TYP/IDEVIC, LSW, LTSW, XYCOORD(4), LUOUT, LPAGE
370 C      COMMON/GRFTYP/ANGLE, IRVRSE, CHSIZE(9), ITKWIT, LUDIAG, LUHSET
371 C      COMMON/DEV POS/X1R(5), Y1R(5), X2R(5), Y2R(5)
372 C      COMMON/GRF MOD/LFLMAT(5), LINWDM(5), LCLMAT(5)

```

```

373      EQUIVALENCE (HWSIZE(1),CALRAS),(HWSIZE(9),LEXRAS)
374      EQUIVALENCE (HWSIZE(13),TEKRAS)
375      DATA START,LINE,SLASH,ENDIT/4H(    , 4H108H, 4H    /, 4H    ) /
376      DATA HBLANK, HMINUS, HAPOSA, HAPOSB, HASTRA, HASTRB
377      1  /4H    , 4H---- , 4H¶    , 4H    ¶ , 4H*---- , 4H---* /
378      DATA OPENANY/.FALSE./, OPNLEX/.TRUE./
379      DATA ITIMET/20/, IDLETM/50/, LUDIAG/0/
380      DATA COUNT/5*0/, NFRAME/5*0/, ANGLE/0.0/
381      DATA TEKRAS/0,0,1022,768/, LEXRAS/50,50,32700,26160/
382      DATA CALRAS/0, 0, 10750, 7900/
383      DATA IDFLTO/'L7: ', 'PR: ', 'LEX:', 'C:', 'NULL: '/
384      DATA X1R/0.,0.,1.E+7,0.,0./, X2R/2*1279.,11279000.,2*1279./
385      DATA Y1R/0.,0.,1.E+7,0.,0./, Y2R/2*1023.,11023000.,2*1023./
386      DATA LUSET/7/, ICODET/27,23/, ICTRLX/31,24/
387      DATA ICHANX/27,97,27,127/, OPNCAL/.TRUE./
388      DATA NFRAME/5*0/,COUNT/5*0/,LUHSET/9/, OPNMAT/.TRUE./
389      DATA LFLMAT/0,0,1,0,0/, LINWDM/ 1,1,2,1,1/, LCLMAT/5*1/
390      DATA MATUNT/8/, MATFIL/'L14: '/, TABFIL/'L19: '/, TABUNT/8/
391      DATA XYCOORD/1.,0.,1.,0./, OPNTAB/.TRUE./
392      DATA CHSIZE/31.,31.,0.3,0.5,0.04,0.58,0.,0.,.03/
393      DATA USPAT/3*3640,3*455,3*3640,3*455,0,0,778,
394      . 992,504,240,240,504,992,778,0,0,2*0,220,270,ZF8,
395      . Z1FC,Z3FE,Z7FF,4*0,
396      . 0,96,240,504,1020,2046,4095,3999,3855,3591,3075,2049/
397      C
398      C   *** START OF THE SECTION WHICH HANDELS FUNCTIONS
399      C
400      C
401      C   DEVICE
402      C
403      IF(OPENANY) GO TO 1100
404      IDEVIC = N
405      LUOUT = LUSET
406      ITKWIT = 40
407      IF (IDEVIC.LT.1.OR.IDEVIC.GT.5) IDEVIC = 4
408      CLOSE(LUOUT)
409      OPEN(UNIT=LUOUT,IOSTAT=IOS,ERR=901,FILE=IDFLTO(IDEVIC),
410      1  TYPE='DEVICE',SHARE='ERW',RKEY=0,WKEY=0)
411      OPENANY = .TRUE.
412      IF (LUDIAG.NE.0) WRITE (LUDIAG,1003) IDEVIC
413      RETURN
414      C
415      C   ERROR ON CALL TO DEVICE
416      C
417      901  CLOSE(LUOUT)
418      IF (LUDIAG.NE.0) WRITE (LUDIAG,1014) IDEVIC, IOS
419      C
420      C   SET A VALUE OF ZERO IN THE DEVICE TYPE TO INDICATE THAT
421      C   A DEVICE CAN NOT BE ACCESSED. ADDED 1/24/85
422      C
423      IDEVIC = 0
424      RETURN
425      C
426      C   SETDEV - CHANGE THE LOGICAL UNIT
427      C

```

```
428     ENTRY SETDEV (N1, N2)
429     IF (N1.GT.0) LUSET = N1
430     IF (N2.GE.0) LUDIAG = N2
431     IF (LUDIAG.NE.0) WRITE (LUDIAG, 1005) LUOUT, LUSET
432     RETURN
433 C
434 C     NEWFRM
435 C
436     ENTRY NEWFRM
437     MODEP = 1
438     IF (.NOT.OPENANY) GO TO 1000
439     COUNT(IDEVIC) = COUNT(IDEVIC) + 1
440     NFRAME(IDEVIC) = 1
441     IRVRSE = .FALSE.
442     IF (LUDIAG.NE.0) WRITE (LUDIAG, 1101) COUNT(IDEVIC)
443     GO TO (1,2,3,4,5), IDEVIC
444 C
445 C     FRAME - FORCE A WRITE OF THE BUFFER
446 C
447     ENTRY FRAME
448     MODEP = 2
449     IF (.NOT.OPENANY) GO TO 1000
450     NFRAME(IDEVIC) = NFRAME(IDEVIC) + 1
451     IF (LUDIAG.NE.0) WRITE (LUDIAG, 1001) NFRAME(IDEVIC), IDEVIC
452     GO TO (1,2,3,4,5), IDEVIC
453 C
454 C     ERASE
455 C
456     ENTRY ERASE
457     MODEP = 3
458     IF (.NOT.OPENANY) GO TO 1000
459     IF (LUDIAG.NE.0) WRITE (LUDIAG, 1501) NFRAME(IDEVIC)
460     GO TO (1,2,3,4,5), IDEVIC
461 C
462 C     HARD COPY
463 C
464     ENTRY HDCOPY
465     MODEP = 4
466     IF (.NOT.OPENANY) GO TO 1000
467     IF (LUDIAG.NE.0) WRITE (LUDIAG, 1502) NFRAME(IDEVIC)
468     GO TO (1,2,3,4,5), IDEVIC
469 C
470 C     COLOR
471 C
472     ENTRY COLOR (N)
473     MODEP = 5
474     IF(.NOT.OPENANY) GO TO 1000
475     ICOLOR = N
476     IF (LUDIAG.NE.0) WRITE (LUDIAG, 1214) ICOLOR
477     GO TO (1,2,3,4,5), IDEVIC
478 C
479 C     SET THE TYPE FOR POLYGON AND CIRCLE FILLING 0, 1, 2, OR 3
480 C
481     ENTRY FILTYP(IFIL)
482     IF(.NOT.OPENANY) GO TO 1000
```

```

483         LFLMAT(IDEVIC) = IFIL
484         IF(LUDIAG.GT.0) WRITE(LUDIAG,1503) LFLMAT
485         MODEP = 5
486         GO TO (1,2,3,4,5), IDEVIC
487 C
488 C     SET THE SIZE OF THE HARDWARE CHARACTERS
489 C
490         ENTRY CHRISZ (CHFRZ, GCHFRZ)
491         IF(.NOT.OPENANY) GO TO 1000
492         IF(CHFRZ.GT.0) CHSIZE(9) = CHFRZ
493         CHSIZE(1) = CHSIZE(9) * (Y2R(IDEVIC)-Y1R(IDEVIC))
494         CHSIZE(2) = CHSIZE(1)
495         IF(GCHFRZ.GT.0) CHSIZE(5) = GCHFRZ
496         IF(LUDIAG.GT.0) WRITE(LUDIAG,1505) CHFRZ, GCHFRZ
497         MODEP = 5
498         RETURN
499 C
500 C     SET THE LINE WIDTH
501 C
502         ENTRY LINWID(LINWD)
503         IF(.NOT.OPENANY) GO TO 1000
504         LINWDM(IDEVIC) = MAX0(LINWD,1)
505         IF(LUDIAG.GT.0) WRITE(LUDIAG,1504) LINWDM
506         MODEP = 5
507         GO TO (1,2,3,4,5), IDEVIC
508 C
509 C     SCALE
510 C
511         ENTRY SCALNG(X1,Y1,X2,Y2,X1H,Y1H,X2H,Y2H,X1S,Y1S,X2S,Y2S)
512         IF(.NOT.OPENANY) GO TO 1000
513         IND = (IDEVIC-1)*4
514         IF(IDEVIC.EQ.1) THEN
515             SCLRAS = 1000.
516         ELSE
517             SCLRAS = 1.
518         ENDIF
519         IF (X1H.GE.0.0) HWSIZE(IND+1) = X1H * SCLRAS
520         IF (Y1H.GT.0.0) HWSIZE(IND+2) = Y1H * SCLRAS
521         IF (X2H.GE.0.0) HWSIZE(IND+3) = X2H * SCLRAS
522         IF (Y2H.GT.0.0) HWSIZE(IND+4) = Y2H * SCLRAS
523         IF (X1S.NE.0.0) X1R(IDEVIC) = X1S
524         IF (X2S.NE.0.0) X2R(IDEVIC) = X2S
525         IF (Y1S.NE.0.0) Y1R(IDEVIC) = Y1S
526         IF (Y2S.NE.0.0) Y2R(IDEVIC) = Y2S
527         GO TO 11
528         ENTRY DEFINE (X1, Y1, X2, Y2)
529 11     MODEP = 6
530         IF(.NOT.OPENANY) GO TO 1000
531         DX = X2 - X1
532         IF (DX.EQ.0.0) DX = 1.0
533         XYCOORD(1) = (X2R(IDEVIC) - X1R(IDEVIC))/DX
534         XYCOORD(2) = X1R(IDEVIC) - XYCOORD(1)*X1
535         DY = Y2 - Y1
536         IF (DY.EQ.0.0) DY = 1.0
537         XYCOORD(3) = (Y2R(IDEVIC) - Y1R(IDEVIC))/DY

```

```

538         XYCOORD(4) = Y1R(IDEVIC) - XYCOORD(3)*Y1
539         IF (LUDIAG.NE.0) WRITE (LUDIAG, 1400) XYCOORD
540         CHSIZE(1) = CHSIZE(9) * (Y2R(IDEVIC)-Y1R(IDEVIC))
541         CHSIZE(2) = CHSIZE(1)
542         GO TO (1,2,3,4,5), IDEVIC
543 C
544 C     ENDFRM - TO CLOSE THE LOGICAL UNIT
545 C
546         ENTRY ENDFRM
547         IF(.NOT.OPENANY) GO TO 1000
548         MODEP = 7
549         IF (LUDIAG.NE.0) WRITE (LUDIAG,1201) COUNT(IDEVIC),
550         . NFRAME(IDEVIC)
551         NFRAME(IDEVIC) = 0
552         GO TO (1,2,3,4,5), IDEVIC
553     10    CLOSE(7)
554         OPENANY = .FALSE.
555         IDEVIC = 0
556         RETURN
557 C
558 C     *** START OF THE SECTION WHICH HANDELS THE HARDWARE ***
559 C
560 C
561 C     CALCOMP SECTION
562 C
563     1     GO TO (101,102,103,103,105,106,107), MODEP
564 C
565 C     CALCOMP INITIALIZATION
566 C
567     101    CALL CALBUF (LUOUT,0,1,IDX)
568         IF(OPNCAL) CALL CALPLT (0.0, +1.0, 1007)
569         OPNCAL = .FALSE.
570         CALL CALPLT (FLOAT(CALRAS(3))/((X2R(1)-X1R(1))*1000.),
571         . FLOAT(CALRAS(4))/((Y2R(1)-Y1R(1))*1000.), 1001)
572         CALL CALPEN (1)
573         LSW = .TRUE.
574         RETURN
575 C
576 C     CALCOMP FRAME (FLUSH BUFFER AND RESET, IGNORE HARDCOPY)
577 C
578     102    RETURN
579 C
580 C     ERASE THE SCREEN AND PRINT HARDCOPY ARE IGNORED FOR THE CALCOMP
581 C
582     103    RETURN
583 C
584 C     COLOR SELECTION PRINTOUT.
585 C
586     105    PEN = MOD(ICOLOR-1,4) + 1
587         PEN = MINO(MAXO(PEN,1),4)
588         LCLMAT(IDEVIC) = PEN
589         CALL CALPEN(PEN)
590         RETURN
591 C
592 C     CALCOMP AXIS SET.

```

```

593 C
594 106 CONTINUE
595 CALL CALPLT (FLOAT(CALRAS(3))/((X2R(1)-X1R(1))*1000.),
596 .    FLOAT(CALRAS(4))/((Y2R(1)-Y1R(1))*1000.), 1001)
597 RETURN
598 C
599 C ENDFRM FOR CALCOMP
600 C
601 107 CONTINUE
602 CALL CALPLT(0., 0., 999)
603 GO TO 10
604 C
605 C PAPER PLOT SECTION
606 C
607 2 GO TO (201,202,203,204,205,206,207), MODEP
608 C
609 C PAPER PLOT INITIALIZATION.
610 C THE PAGE PLOT IS OUTPUT AS ONE LARGE VARIABLE FORMAT.
611 C SET UP THE FORMAT STATEMENT IN LPAGE.
612 C
613 201 CONTINUE
614 203 DO 211 J=1,65
615 LPAGE(2,J)=LINE
616 211 LPAGE(30,J)=SLASH
617 LPAGE(1,1)=START
618 LPAGE(30,65)=ENDIT
619 C
620 C SET UP THE PAGE AND CLEAN THE WORK AREA.
621 C
622 DO 221 J = 1, 65
623 DO 221 I = 4, 29
624 221 LPAGE(I,J) = HBLANK
625 DO 222 I = 5, 28
626 LPAGE(I,1) = HMINUS
627 222 LPAGE(I,63) = HMINUS
628 DO 223 J = 2, 62
629 LPAGE(4,J) = HAPOSA
630 223 LPAGE(29,J) = HAPOSB
631 LPAGE(29,1) = HASTRB
632 LPAGE(29,63) = HASTRB
633 LPAGE(4,63) = HASTRA
634 LPAGE(4, 1) = HASTRA
635 RETURN
636 C
637 C PAPER PLOT- WRITE OUT THE PAGE AS IT IS.
638 C
639 202 WRITE(LUOUT, LPAGE)
640 RETURN
641 C
642 C HARDCOPY COMMAND
643 C
644 204 RETURN
645 C
646 C COLOR SELECTION PRINTOUT.
647 C

```

```

648     205     RETURN
649 C
650 C     SET SCALING FOR PAGE PLOT
651 C
652     206     RETURN
653 C
654 C     ENDFRM FOR THE PAPER PLOTTER
655 C
656     207     WRITE (LUOUT,1212)
657             GO TO 10
658 C
659 C     LEXIDATA/MATRIX SECTION
660 C
661     3       GO TO (301,302,303,304,305,306,307), MODEP
662 C
663 C     INITIALIZE THE LEXIDATA/MATRIX - OPEN AND SET THE LUT ONLY
664 C     IF THE DEVICE IS/WAS CLOSED
665 C
666     301     IF(OPNLEX) THEN
667             CALL GSOPN(LUOUT, 1, LIERR)
668             IF(LIERR.NE.0) GO TO 1002
669 C
670 C     CURSOR OFFSET - THIS IS INSTALLATION SPECIFIC
671 C
672             CALL DSCSL(10, 153, 75)
673             CALL DSCER
674             CALL SETLUT ; SET THE DEFAULT LOOK UP TABLE
675             CALL GSTYPE(3)
676 C
677 C     SET VIEW CORRESPONDENCE
678 C
679             CALL GDEFVS (1, 15, 0, 1, 4, 0)
680             CALL GINTVS (1)
681             CALL GACTVS (1)
682             CALL GDFWIN (1, X1R(IDEVIC), Y1R(IDEVIC), X2R(IDEVIC),
683             Y2R(IDEVIC))
684             CALL GDFVP (1, LEXRAS(1), LEXRAS(2), LEXRAS(3), LEXRAS(4))
685             VSNUM(1) = 2
686             VSNUM(2) = 0
687             CALL GDFVU (1, 1, 1, VSNUM)
688             CALL GACTVU(1)
689             CALL GDASEG
690             CALL GCRSEG(100)
691             DO 3001 I = 1, 4
692     3001    CALL GDPAT(100+I, USPAT(1,I))
693             CALL GCLSEG
694             CALL GSVUAS(100, VSNUM)
695             CALL GDLSEG(100)
696             CALL GSIVIS(1)
697             CALL GSVUAS (0, VSNUM)
698 C
699 C     DEFINE LOCATOR PORT EXTENTS AND SET TRACKING TYPE
700 C
701             CALL GDVINI(DEVID, 2, 1)
702             CALL GDFLPT(DEVID, LEXRAS(1),LEXRAS(2),LEXRAS(3),LEXRAS(4))

```



```

703          CALL GSTTRK(DEVID, 2, 1)
704 C
705 C SET ECHO TYPE
706 C
707          CALL GSTDVE(DEVID, 1, 1)
708          CALL GDVENB(DEVID)
709          CALL GRSLG (DEVID, 255)
710          OPNLEX = .FALSE.
711          ENDIF
712 C
713 C SET THE PARAMETERS FOR THE LEXIDATA DISPLAY.
714 C
715          CALL GCLSEG
716          CALL DSCLR(-1)
717          CALL GCTSEG
718          CALL GSFTYP(LFLMAT(IDEVIC))
719          CALL GSLWID(LINWDM(IDEVIC))
720          CALL GSCHSZ(CHSIZE(2))
721          CALL GSCNDX (1)
722          RETURN
723 C
724 C FRAME - FORCE OUT THE CONTENTS OF THE BUFFER
725 C
726 302      CALL GCLSEG
727          CALL GMPCUR
728          CALL GCTSEG
729          RETURN
730 C
731 C ERASE THE SCREEN AND THE CURSOR
732 C
733 303      CALL DSCLR(-1)
734          CALL DSCSL (10, 153, 75)
735          RETURN
736 C
737 C HARD COPY - CAMERA
738 C
739 304      IF(OPNMAT) THEN
740          CLOSE(MATUNT)
741          OPNTAB = .TRUE.
742          OPEN(UNIT=MATUNT, IOSTAT=IOS, ERR=314, FILE=MATFIL,
743          .   TYPE='DEVICE', SHARE='ERW', RKEY=0, WKEY=0)
744          OPNMAT = .FALSE.
745          ENDIF
746          IF(LUDIAG.NE.0) WRITE(LUDIAG,1003) MATUNT
747          CALL SYSIO(PBLK,41,MATUNT, '.CE.',4,0)
748          CALL WAIT(ITIMET,2,IOS)
749          RETURN
750 C
751 C ERROR ON CALL TO DEVICE
752 C
753 314      CLOSE(MATUNT)
754          IF (LUDIAG.NE.0) WRITE (LUDIAG,1014) MATUNT, IOS
755          STOP 'camera can not be accessed'
756 C
757 C SET COLOR - THIS APPLIES TO WHAT IS IN THE LUT

```

```

758 C
759 305   LEXCOL = MOD(ICOLOR-1,15) + 1
760       LEXCOL = MAX0(LEXCOL,1)
761       LCLMAT(IDEVIC) = LEXCOL
762       CALL GSCNDX(LCLMAT(IDEVIC))
763       CALL GSLWID(LINWDM(IDEVIC))
764       LFLLEX = LFLMAT(IDEVIC)
765       IF(LFLLEX.LE.4) THEN
766         CALL GSFTYP(LFLLEX)
767       ELSE
768         CALL GSFTYP(96+LFLLEX)
769       ENDIF
770       RETURN
771 C
772 C     SET THE WINDOW AND AREA OF NORMALIZED DEVICE COORDINATES
773 C
774 306   IF(OPNLEX) GO TO 301
775       CALL GCLSEG
776       CALL GDAVU(1)
777       CALL GDFWIN (1, X1R(IDEVIC), Y1R(IDEVIC), X2R(IDEVIC),
778         .      Y2R(IDEVIC))
779       CALL GDFVP (1, LEXRAS(1), LEXRAS(2), LEXRAS(3), LEXRAS(4))
780       CALL GDFVU (1, 1, 1, VSNUM)
781       CALL GACTVU (1)
782 C
783 C     OPEN A NEW SEGMENT
784 C
785       CALL GCTSEG
786       RETURN
787 C
788 C
789 307   OPNLEX = .TRUE.
790       IF (.NOT.OPNMAT) THEN
791         CLOSE(MATUNT)
792         OPNMAT = .TRUE.
793       ENDIF
794       CALL GCLSEG
795       CALL GDASEG
796       CALL GFRAME
797       CALL DSCLR(-1)
798       CALL GRSLG(DEVID,255)
799       CALL GDVDSB(DEVID)
800       GO TO 10
801 C
802 C     TEKTRONIX SECTION
803 C
804 4     GO TO (401,402,403,404,405,406,407), MODEP
805 C
806 C     TEKTRONIX INITIALIZATION
807 C
808 401   CALL TEKINT (120)
809       CALL SWINDO (TEKRAS(1),TEKRAS(3),TEKRAS(2),TEKRAS(4))
810       CALL WINDO (X1R(IDEVIC), X2R(IDEVIC)-X1R(IDEVIC),
811         .      Y1R(IDEVIC), Y2R(IDEVIC)-Y1R(IDEVIC))
812       LTSW = .TRUE.

```

```
813          CALL TEKHOM
814          CALL TTKSND
815          CALL WAIT (IDLETM, 1, IS)
816          RETURN
817 C
818 C          FRAME (NO MODE SWITCHING)
819 C
820 402      CONTINUE
821          CALL TTKSND
822          CALL WAIT (IDLETM, 1, IS)
823          RETURN
824 C
825 C          ERASE THE SCREEN
826 C
827 403      CONTINUE
828          CALL TEKHOM
829          CALL TEKERA
830          CALL TOUTST (2, ICTRLX)
831          CALL TTKSND
832          CALL WAIT (IDLETM, 1, IS)
833          RETURN
834 C
835 C          ISSUE HARD COPY COMMAND
836 C
837 404      CONTINUE
838          CALL TOUTST (2, ICODET)
839          CALL TEKHOM
840          CALL TTKSND
841          CALL WAIT (ITIMET, 2, IS)
842          RETURN
843 C
844 C          COLOR - DOES NOT APPLY TO ADM OR TEKTRONIX 4054
845 C
846 405      CONTINUE
847          RETURN
848 C
849 C          SET SCREEN CORRESPONDENCE
850 C
851 406      CONTINUE
852          CALL SWINDO (TEKRAS(1),TEKRAS(3),TEKRAS(2),TEKRAS(4))
853          CALL VWINDO (X1R(IDEVIC), X2R(IDEVIC)-X1R(IDEVIC),
854                      Y1R(IDEVIC), Y2R(IDEVIC)-Y1R(IDEVIC))
855          RETURN
856 C
857 C          END THE FRAME ON THE ADM OR TEK 4054
858 C
859 407      CALL TEKHOM
860          CALL TOUTST (2, ICTRLX)
861          CALL TTKSND
862          CALL WAIT (IDLETM, 1, IS)
863          CALL SVSTAT (SCRATCH)
864          GO TO 10
865 C
866 C          SLOT FOR AN EXTRA DEVICE
867 C
```

```

868      5      STOP 74
869      C
870      C      DELAY IN SECONDS FOR HARD COPY UNITS
871      C
872          ENTRY DELAY(ITIME)
873          ITIMET = ITIME
874          RETURN
875      C
876      C      IOWAIT
877      C
878          ENTRY IOWAIT(IT)
879          CALL WAIT (IT, 1, IS)
880          RETURN
881      C
882      C      GET THE DEVICE ID FOR THE LEXIDATA TRACKING TYPE
883      C
884          ENTRY LEXDID(IDVL)
885          IDVL = DEVID
886          RETURN
887      C
888      C      OPEN ACCESS TO THE TABLET - IF POSSIBLE
889      C
890          ENTRY TABLET (OPENTB)
891          OPENTB = .FALSE.
892          IF(OPNTAB) THEN
893              IF(.NOT.OPNMAT) CLOSE(MATUNT)
894              OPEN(UNIT=TABUNT, IOSTAT=IOS, ERR=1304, FILE=TABFIL,
895                  .           TYPE='DEVICE', SHARE='ERW', RKEY=0,
896                  .           WKEY=0)
897              OPNTAB = .FALSE.
898              OPENTB = .TRUE.
899              OPNMAT = .TRUE.
900          ELSE
901              IF(LUDIAG.GT.0) WRITE(LUDIAG,1302)
902              OPENTB = .TRUE.
903          ENDIF
904          RETURN
905      C
906      C      THE TABLET HAS GENERATED AN ERROR
907      C
908      1304      CLOSE(TABUNT)
909              IF(LUDIAG.GT.0) WRITE(LUDIAG,1303) TABUNT,IOS
910              STOP 'can not access tablet'
911      C
912      C      PUNISH IF NOT INITIALIZED.
913      C
914      1000      IF (LUDIAG.NE.0) WRITE (LUDIAG, 1004) MODEP, IDEVIC
915              IDEVIC = 0
916              RETURN
917      1002      IF (LUDIAG.NE.0) WRITE(LUDIAG, 1013)LIERR
918              STOP 'Lexidata can not be accessed'
919      1100      IF (LUDIAG.NE.0) WRITE(LUDIAG, 1006) IDEVIC
920              RETURN
921      C
922      C      FORMATS

```

```
923 C
924 1001 FORMAT (' FRAMES PLOTTED = ',I5,' ON DEVICE ',I3)
925 1003 FORMAT (' DEVICE = ',I2,' INITIALIZED')
926 1004 FORMAT (' DEVICE NOT INITIALIZED. ', 2I5)
927 1005 FORMAT (' LUOUT = ',I3,' AND WILL BE SET TO ',I3,
928      ' AT THE NEXT CALL TO "DEVICE"')
929 1006 FORMAT (' DEVICE ',I3,' IS ALREADY OPEN')
930 1013 FORMAT (' CAN NOT OPEN LEXIDATA, ERROR =',I3)
931 1014 FORMAT (' DEVICE ',I3,' CAN NOT BE ACESSED, ERROR =',I3)
932 1101 FORMAT (' FRAME NUMBER ',I5,' INITIALIZED')
933 1201 FORMAT (' PLOTTER CLOSED WITH',2I5,' FRAMES')
934 1212 FORMAT ('1')
935 1214 FORMAT (' COLOR SELECT = ',I3)
936 1302 FORMAT(' CAMERA IS ALREADY AVAILABLE')
937 1303 FORMAT (' CAMERA ',I3,' CAN NOT BE ACESSED, ERROR =',I3)
938 1400 FORMAT (' FACTOR ',4F8.2)
939 1501 FORMAT (' ERASE COMMAND ISSUED AT FRAME ',I5)
940 1502 FORMAT (' HARDCOPY COMMAND ISSUED AT FRAME ',I5)
941 1503 FORMAT (' SET THE FILL TYPE FOR SURFACES ',5I3)
942 1504 FORMAT (' SET THE LINE WIDTH ',5I3)
943 1505 FORMAT (' SET THE CHARACTER SIZE (%) ',2F6.3)
944      END
```

```

945     SUBROUTINE DEVINP(INPUT,BTMASK,IDV,X,Y,Z)
946     LOGICAL JFIRST/.TRUE./,TFIRST/.TRUE./,KFIRST/.TRUE./
947     INTEGER BTMASK,DEVID,PBLK(6),TLOOK(4)/1,2,4,8/
948     CHARACTER*1 INST(2)/Z1B,'A'/
949     CHARACTER*20 DATAIN
950     COMMON/DEVTYP/IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
951     COMMON/TXXDAT/TABIN,TABWIN,TABXYC
952     REAL TABIN(2),TABWIN(4),TABXYC(4)
953     COMMON/DEVPOS/X1R(5),Y1R(5),X2R(5),Y2R(5)
954 C
955 C     SELECT THE INPUT FROM THE DEVICE CODE
956 C
957     X = 0.
958     Y = 0.
959     Z = 0.
960     IDV = 0
961     IF(INPUT.LT.1.OR.INPUT.GT.3) RETURN
962     IF(INPUT.NE.1.AND.(.NOT.JFIRST)) THEN
963         CALL DSCSL (10, 153, 75)
964         JFIRST = .TRUE.
965     ENDIF
966     GO TO (10, 20, 30), INPUT
967 C
968 C     GET THE DEVICE ID FOR THE JOYSTICK
969 C
970     10 IF(JFIRST) THEN
971         CALL LEXDID(DEVID)
972         CALL GDFBGF(DEVID, 15, 0, IDUM)
973         CALL DSCSL (8, 153, 75)
974         JFIRST = .FALSE.
975     ENDIF
976 C
977 C     SET APPROPRIATE LIGHTS
978 C
979     CALL GSLTG(DEVID,BTMASK)
980 C
981 C     READ THE LOCATOR
982 C
983     XWC = 0.0
984     YWC = 0.0
985     3 CALL GRBTG(DEVID, IDV)
986     IDV = IAND(IAND(IDV,BTMASK),15)
987     IF(IDV.EQ.0) GO TO 3 ; NO SWITCH HAS BEEN TOGGLED
988     CALL GSLC(DEVID, IDVV, LCX, LCY, XWC, YWC)
989     IF(LCX.LT.0) GO TO 3
990     CALL GRSLG (DEVID, 255)
991     2 CALL GRBTG (DEVID, IDVV)
992     IF(IDVV.NE.0) GO TO 2
993 C
994 C     NORMALIZE TO <RE>WINDOWED COORDINATES
995 C
996     X = (XWC-XYCOORD(2)) / XYCOORD(1)
997     Y = (YWC-XYCOORD(4)) / XYCOORD(3)
998     Z = 0.0
999     RETURN

```

```

1000 C
1001 C   THE TABLET HAS BEEN SELECTED
1002 C
1003   20  IF(TFIRST) THEN
1004       TFIRST = .FALSE.
1005       CALL TABLET(OPENTB)
1006       IF(IDEVIC.EQ.0) THEN
1007           DX = 1280.
1008           DY = 1024.
1009       ELSE
1010           DX = (X2R(IDEVIC)-X1R(IDEVIC)) / XYCOORD(1)
1011           DY = (Y2R(IDEVIC)-Y1R(IDEVIC)) / XYCOORD(3)
1012       ENDIF
1013       TABXYC(1) = 8.2034E-5 * DX
1014       TABXYC(2) = 0.0
1015       TABXYC(3) = 1.0938E-4 * DY
1016       TABXYC(4) = 0.0
1017   ENDIF
1018   21  CALL SYSIO(PBLK,72,8,DATAIN,20,0)
1019       READ(DATAIN,22,ERR=21) IX,IY,IBT
1020   22  FORMAT(1X,I5,2X,I5,1X,I2)
1021       IF(IBT.EQ.0) GO TO 21
1022 C
1023 C   CONVERT THE CURSOR BUTTON TO A STANDARD FORMAT
1024 C
1025       IDV = TLOOK(IBT)
1026       IDV = IAND(BTMASK, IDV)
1027       IF(IDV.EQ.0) GO TO 21
1028       TABIN(1) = IX
1029       TABIN(2) = IY
1030       X = FLOAT(IX)*TABXYC(1) + TABXYC(2)
1031       Y = FLOAT(IY)*TABXYC(3) + TABXYC(4)
1032       Z = 0.0
1033       RETURN
1034 C
1035 C   INPUT FROM THE KEYBOARD
1036 C
1037   30  CALL SYSIO (PBLK,41,5,'X,Y,Z,IDV = ',12,0)
1038       READ(5,32,END=33,ERR=30) XCOR,YCOR,ZCOR,IDV
1039   32  FORMAT(3F,I)
1040       X = XCOR
1041       Y = YCOR
1042       Z = ZCOR
1043       IDV = IAND(BTMASK, IDV)
1044       RETURN
1045   33  IDV = BTMASK
1046       RETURN
1047   END

```

```

1048     SUBROUTINE ENUMBR (REALNO, XR1, YR1, XR2, YR2)
1049     C
1050     C*****
1051     C     ENUMBR
1052     C*****
1053     C
1054         DX = (XR2 - XR1)/10.0
1055         THENO = REALNO
1056         EXPON = 0.001
1057         IF (REALNO .EQ. 0.0) GO TO 10
1058     12     IF (ABS(THENO) .LT. 10.0) GO TO 11
1059         EXPON = EXPON + 1.0
1060         THENO = THENO*0.1000001
1061         GO TO 12
1062     11     IF (ABS(THENO) .GE. 1.0) GO TO 10
1063         EXPON = EXPON - 1.0
1064         THENO = THENO*9.99999
1065         GO TO 11
1066     10     CALL FNUMBR (XR1, YR1, DX, 0.0, 0.7*DX, 0.0, YR2-YR1,
1067     1         THENO, 5, 2)
1068         CALL WDDRAW (XR1+5.0*DX, YR1, DX, 0.0, 0.7*DX, 0.0,
1069     1         YR2-YR1, '*10q.')
1070         CALL FNUMBR (XR1+8.0*DX, (YR1+YR2)/2.0, 0.66*DX, 0.0,
1071     1         0.75*DX, 0.0, 0.64*(YR2-YR1), EXPON, 3, 0)
1072     C     1         0.50*DX, 0.0, 0.6*(YR2-YR1), EXPON, 3, 0)
1073         RETURN
1074     END

```



```

1075 SUBROUTINE FNUMBR (XSTART, YSTART, DX, DY, SX, SKY, SY,
1076 1 NUMBR, WIDTH, DIGITS)
1077 C
1078 C X REAL - X COORDINATE (MATHEMATICAL SPACE) OF THE FIRST
1079 C CHARACTER TO BE DRAWN
1080 C Y REAL - Y COORDINATE (MATHEMATICAL SPACE) OF THE FIRST
1081 C CHARACTER TO BE DRAWN
1082 C DX REAL - INCREMENT TO BE ADDED TO THE X COORDINATE FOR
1083 C EACH CHARACTER DRAWN
1084 C DY REAL - INCREMENT TO BE ADDED TO THE Y COORDINATE FOR
1085 C EACH CHARACTER DRAWN
1086 C SX REAL - X MATH SPACE SIZE FOR CHARACTERS
1087 C SKY REAL - SLANT MODIFIER FOR CHARACTERS
1088 C SY REAL - Y MATH SPACE SIZE FOR CHARACTERS
1089 C RNUMB REAL - NUMBER TO BE PLOTTED IN F FORMAT
1090 C WIDTH INTEGER - TOTAL WIDTH OF FIELD INCLUDING DECIMAL POINT
1091 C DIGITS INTEGER - NUMBER TO FRACTIONAL PLACES TO BE DRAWN
1092 C
1093 REAL NUMBER, NUMBR, TEST(6)
1094 INTEGER WIDTH, DIGITS, NUM, NN(22)
1095 LOGICAL SPACE
1096 INTEGER WORD(7), HASTR, HTEMP, SFIL(3)
1097 INTEGER TEXT(13)
1098 CHARACTER*1 WWORD(28), BLANK, HMINUS
1099 DATA TEXT /Z00000030, Z00000031, Z00000032, Z00000033,
1100 1 Z00000034, Z00000035, Z00000036, Z00000037,
1101 2 Z00000038, Z00000039, Z00000020, Z0000002E,
1102 3 Z0000007C /
1103 DATA SFIL / '*q..', '**q.', '***q' /
1104 DATA HMINUS, HASTR / '- ', '*****' /, BLANK/' '/
1105 EQUIVALENCE (WORD(1),TEST(1),WWORD(1))
1106 C
1107 I1 = 1
1108 I2 = 1
1109 NUMBER = ABS(NUMBR)
1110 IF (DIGITS.EQ.0) NUM = NUMBER + 0.5
1111 IF (DIGITS.EQ.0) GO TO 5
1112 C
1113 WIDTH = WIDTH - 1
1114 NUM = NUMBER*10**DIGITS +0.5
1115 5 DO 10 I = 1,WIDTH
1116 J = WIDTH - I + 1
1117 NN(J) = NUM - 10*(NUM/10)
1118 C
1119 C BREAK OUT THE DIGITS.
1120 C
1121 NUM = NUM/10
1122 10 CONTINUE
1123 J = WIDTH - DIGITS
1124 IF(J .NE. 0) GO TO 18
1125 WORD(1) = TEXT(11)
1126 I1 = I1 + 1
1127 GO TO 25
1128 18 SPACE = .TRUE.
1129 DO 20 K = 1,J

```

```

1130         I = NN(K)
1131 C
1132 C     LEADING ZERO = SPACE
1133 C
1134         IF (I.NE.0) SPACE = .FALSE.
1135         IF(K .EQ. J .AND. SPACE) SPACE = .FALSE.
1136         IF (I.EQ.0.AND.SPACE) I = 10
1137         I1 = I1 + 1
1138         HTEMP = ISHFT(WORD(I2), 8)
1139         WORD(I2) = IOR(HTEMP,TEXT(I+1))
1140         IF(I1 .LT. 5) GO TO 20
1141         I2 = I2 + 1
1142         I1 = 1
1143     20    CONTINUE
1144         IF (DIGITS.EQ.0) GO TO 40
1145 C
1146 C     PUT IN DECIMAL POINT
1147 C
1148     25    I1 = I1 + 1
1149         HTEMP = ISHFT(WORD(I2), 8)
1150         WORD(I2) = IOR(HTEMP,TEXT(12))
1151         IF(I1 .LT. 5) GO TO 26
1152         I2 = I2 + 1
1153         I1 = 1
1154     26    DO 30 K = 1,DIGITS
1155         I = NN(J+K)
1156         I1 = I1 + 1
1157         HTEMP = ISHFT(WORD(I2), 8)
1158         WORD(I2) = IOR(HTEMP,TEXT(I+1))
1159         IF(I1 .LT. 5) GO TO 30
1160         I2 = I2 + 1
1161         I1 = 1
1162     30    CONTINUE
1163         WIDTH = WIDTH + 1
1164 C
1165 C     PUT IN 1. (END OF TEXT)
1166 C
1167     40    I1 = I1 + 1
1168         HTEMP = ISHFT(WORD(I2), 8)
1169         WORD(I2) = IOR(HTEMP,TEXT(13))
1170         IF(I1 .LT. 5) GO TO 42
1171         I2 = I2 + 1
1172         I1 = 1
1173     42    HTEMP = ISHFT(WORD(I2), 8)
1174         WORD(I2) = IOR(HTEMP,TEXT(12))
1175 C
1176 C     LEFT JUSTIFY LAST WORD.
1177 C
1178     44    I1 = I1 + 1
1179         IF(I1 .GE. 5) GO TO 46
1180         WORD(I2) = ISHFT(WORD(I2), 8)
1181         GO TO 44
1182 C
1183 C     PUT IN THE MINUS SIGN
1184 C

```

```
1185 46 IF (NUMBR .GE. 0.0) GO TO 50
1186 IF(WWORD(1).NE.BLANK) GO TO 60
1187 NUMMX = 4 * I2
1188 DO 47 I = 2, NUMMX
1189 IP = I
1190 IF(WWORD(I).NE.BLANK) GO TO 48
1191 47 CONTINUE
1192 48 WWORD(IP-1) = HMINUS
1193 GO TO 50
1194 60 I2 = WIDTH/4
1195 IF(I2 .EQ. 0)GO TO 64
1196 DO 62 K = 1, I2
1197 62 WORD(K) = HASTR
1198 64 I1 = WIDTH - 4*I2
1199 IF(I1 .GT. 0) WORD(I2+1) = SFIL(I1)
1200 C
1201 50 CALL WDDRAW(XSTART, YSTART, DX, DY, SX, SXY, SY, WORD)
1202 RETURN
1203 END
```

```

1204 SUBROUTINE GRAFIT (NPL,X1,X2,X1R,X2R,IX1,IX2,Y1,Y2,Y1R,Y2R,IY1,
1205 .   IY2,XTIT,NDVX,YTIT,NDVY)
1206 C
1207 C   NPL   INTEGER - THE NUMBER OF THE PLOT REFERENCED.  UP TO FOUR
1208 C                   PLOTS MAY BE PLACED ON ONE GRAPH
1209 C   X1    REAL   - A FLOATING POINT NUMBER, THE LABEL OF THE X
1210 C                   AXIS MINIMUM.  THE LITERAL STRING 'NONE'
1211 C                   PRODUCES NO LABEL
1212 C   X2    REAL   - A FLOATING POINT NUMBER, THE LABEL OF THE X AXIS
1213 C                   MAXIMUM.  THE LITERAL STRING 'NONE' PRO-
1214 C                   DUCES NO LABEL
1215 C   X1R   REAL   - THE USER SPACE VALUE OF THE X AXIS MINIMUM
1216 C   X2R   REAL   - THE USER SPACE VALUE OF THE X AXIS MAXIMUM
1217 C   IX1   REAL   - THE MINIMUM VALUE OF X IN THE DATA TO BE PLOTTED
1218 C   IX2   REAL   - THE MAXIMUM VALUE OF X IN THE DATA TO BE PLOTTED
1219 C   Y1    REAL   - A FLOATING POINT NUMBER, THE LABEL OF THE Y
1220 C                   AXIS MINIMUM.  THE LITERAL STRING 'NONE'
1221 C                   PRODUCES NO LABEL.
1222 C   Y2    REAL   - A FLOATING POINT NUMBER, THE LABEL OF THE Y
1223 C                   AXIS MAXIMUM.  THE LITERAL STRING 'NONE'
1224 C                   PRODUCES NO LABEL
1225 C   Y1R   REAL   - THE USER SPACE VALUE OF THE Y AXIS MINIMUM
1226 C   Y2R   REAL   - THE USER SPACE VALUE OF THE Y AXIS MAXIMUM
1227 C   IY1   REAL   - THE MINIMUM VALUE OF Y IN THE DATA TO BE PLOTTED
1228 C   IY2   REAL   - THE MAXIMUM VALUE OF Y IN THE DATA TO BE PLOTTED
1229 C   XTIT  CHAR   - A STRING OF LITERAL CHARACTERS TERMINATED BY
1230 C                   A ¶.  THIS IS THE TITLE FOR THE X AXIS AND
1231 C                   SHOULD BE LESS THAN 30 CHARACTERS
1232 C   NDVX  INTEGER - THE NUMBER OF INTERVALS TO BE DRAWN ON THE
1233 C                   X AXIS
1234 C   YTIT  CHAR   - A STRING OF LITERAL CHARACTERS TERMINATED BY
1235 C                   A ¶.  THIS IS THE TITLE FOR THE Y AXIS AND
1236 C                   SHOULD BE LESS THAN 30 CHARACTERS
1237 C   NDVY  INTEGER - THE NUMBER OF INTERVALS TO BE DRAWN ON THE
1238 C                   Y AXIS
1239 C
1240 C   ENTRY POINTS (WITH ARGUMENTS):
1241 C
1242 C   PLOTCH (NPL,X,Y,N,CHARAC) - PLOTS ALPHANUMERIC CHARACTERS AT THE
1243 C                   COORDINATES PROVIDED
1244 C   X     REAL   - A ONE DIMENSIONAL ARRAY CONTAINING THE
1245 C                   COORDINATES FOR X
1246 C   Y     REAL   - A ONE DIMENSIONAL ARRAY CONTAINING THE
1247 C                   COORDINATES FOR Y
1248 C   N     INTEGER - THE NUMBER OF ENTRIES IN 'X' AND 'Y'.  IF 'N' =
1249 C                   NO COORDINATES ARE PLOTTED.
1250 C   PCHR  CHAR*1  A SINGLE CHARACTER TO BE USED FOR PLOTTING
1251 C
1252 C   PLOTLN (NPL,X,Y,N) - PLOTS STRAIGHT LINES THROUGH THE COORDINATES
1253 C                   PROVIDED.  ARGUMENTS ARE THE SAME AS FOR
1254 C                   PLOTCH.
1255 C
1256 C   PARAMETER (NLINES=200,NPLTMX=4)
1257 C   REAL IX1, IX2, IY1, IY2, NONE, HOLD(12,NPLTMX), WX(NLINES)
1258 C   REAL CC(16), C(16), WY(NLINES)

```

```

1259 CHARACTER*1 XTIT(NDVX), YTIT(NDVY), PCHR, TITLE(132)
1260 INTEGER IX, IY, IT, NTX, NTY, INDX(16)
1261 DIMENSION X(1), Y(1), XL(1), YB(1), XR(1), YT(1)
1262 LOGICAL PLOTON(NPLTMX)/NPLTMX*.FALSE./
1263 LOGICAL LLABPL(NPLTMX)/NPLTMX*.FALSE./
1264 DIMENSION NTICX(NPLTMX), NTICY(NPLTMX), XMINO(NPLTMX), XMAXO(NPLTMX)
1265 DIMENSION YMINO(NPLTMX), YMAXO(NPLTMX)
1266 COMMON /DEV TYP/ IDEVIC, LSW, LTSW, XYCOORD(4), LUOUT, LPAGE
1267 COMMON /GRFTYP/ ANGLE, IRVRSE, CHSIZE(9), ITKWIT, LUDIAG
1268 DATA NONE/'NONE'/
1269 DATA CC/.0500,.0600,.0310,.0600,.11,.15,
1270 .0800,.0781,.1780,.2500,.0300,.0315,.0600,.040,
1271 .020,.025/
1272 DATA INDX/3,3,3,9,9,9,3,3,3,3,9,9,9,3,3,9/
1273 C
1274 C THIS ROUTINE INITIALIZES A GRAPHICS GRID OR CONTOUR DIAGRAM
1275 C WITH A FULL SET OF LABELS AND SWITCHES THE REST OF THE CONTOUR
1276 C GRAPHICS MATERIAL TO THE GRAPHICS MODE.
1277 C
1278 C DEFINE THE PLOTTING REGIONS.
1279 C
1280 CHSIZE(4) = .44
1281 IF(NPL.GT.NPLTMX.OR.NPL.LT.1) STOP 72
1282 PLOTON(NPL) = .TRUE.
1283 HOLD(1,NPL) = X1
1284 HOLD(2,NPL) = X2
1285 HOLD(3,NPL) = X1R
1286 HOLD(4,NPL) = X2R
1287 HOLD(5,NPL) = IX1
1288 HOLD(6,NPL) = IX2
1289 HOLD(7,NPL) = Y1
1290 HOLD(8,NPL) = Y2
1291 HOLD(9,NPL) = Y1R
1292 HOLD(10,NPL) = Y2R
1293 HOLD(11,NPL) = IY1
1294 HOLD(12,NPL) = IY2
1295 IF(LLABPL(NPL)) THEN
1296 HOLD(1,NPL) = XMINO(NPL)
1297 HOLD(2,NPL) = XMAXO(NPL)
1298 HOLD(5,NPL) = XMINO(NPL)
1299 HOLD(6,NPL) = XMAXO(NPL)
1300 HOLD(7,NPL) = YMINO(NPL)
1301 HOLD(8,NPL) = YMAXO(NPL)
1302 HOLD(11,NPL) = YMINO(NPL)
1303 HOLD(12,NPL) = YMAXO(NPL)
1304 ENDIF
1305 CALL BOXPLT (X1R, Y1R, X2R, Y2R)
1306 DO 11 I = 1, 16
1307 11 C(I) = CC(I) * (HOLD(INDX(I)+1,NPL)-HOLD(INDX(I),NPL))
1308 C
1309 C DETERMINE NUMBER OF CHARACTERS IN AXIS TITLES.
1310 C
1311 TITLE(1) = 'q'
1312 TITLE(2) = 'M'
1313 MAXT = 130

```

```

1314     CALL WDCOUNT(XTIT, NTX)
1315     IF (NTX.LT.1) THEN
1316         IF (LUDIAG.GT.0) WRITE (LUDIAG,12)
1317     ELSE
1318         DO 15 I = 1, MAXT
1319     15  TITLE(I+2) = XTIT(I)
1320         C(8) = (X2R+X1R)/2.
1321         XRR = C(8) + NTX*C(1)
1322         CALL LABEL(TITLE, C(8), Y1R-C(6), XRR, Y1R-C(6)+C(13), 0.0)
1323     END IF
1324     12  FORMAT(' NO ESCAPE SEQUENCE IN TITLE - GRAFIT')
1325     CALL WDCOUNT(YTIT, NTY)
1326     IF(NTY.LT.1) THEN
1327         IF (LUDIAG.GT.0) WRITE (LUDIAG,12)
1328     ELSE
1329         C(11) = (Y1R+Y2R) / 2.
1330         DO 16 I = 1, MAXT
1331     16  TITLE(I+2) = YTIT(I)
1332         CALL LABEL(TITLE, X1R-C(7), C(11), X1R-C(7)+NTY*C(1),
1333         . C(11)+C(13), 1.5707)
1334     ENDIF
1335 C
1336 C     DECIDE WHICH LABELS TO USE
1337 C
1338     IF(LLABPL(NPL)) THEN
1339         CALL FNUMBR (X1R-C(10), Y1R-C(16), C(3), 0.0,
1340         . C(2),0.0,C(4), YMINO(NPL), 6, 1)
1341         CALL FNUMBR (X1R-C(10), Y2R-C(16), C(3), 0.0,
1342         . C(2),0.0,C(4), YMAXO(NPL), 6, 1)
1343         CALL FNUMBR (X1R-C(9), Y1R-C(5), C(3), 0.0,
1344         . C(2),0.0,C(4), XMINO(NPL), 6, 1)
1345         CALL FNUMBR (X2R-C(9), Y1R-C(5), C(3), 0.0,
1346         . C(2),0.0,C(4), XMAXO(NPL), 6, 1)
1347     ELSE
1348         IF(Y1.NE.NONE)
1349         . CALL FNUMBR (X1R-C(10), Y1R-C(16), C(3), 0.0,
1350         . C(2),0.0,C(4), Y1, 6, 1)
1351         IF(Y2.NE.NONE)
1352         . CALL FNUMBR (X1R-C(10), Y2R-C(16), C(3), 0.0,
1353         . C(2),0.0,C(4), Y2, 6, 1)
1354         IF(X1.NE.NONE)
1355         . CALL FNUMBR (X1R-C(9), Y1R-C(5), C(3), 0.0,
1356         . C(2),0.0,C(4), X1, 6, 1)
1357         IF(X2.NE.NONE)
1358         . CALL FNUMBR (X2R-C(9), Y1R-C(5), C(3), 0.0,
1359         . C(2),0.0,C(4), X2, 6, 1)
1360     ENDIF
1361 C
1362 C     PUT TIKS ON THE GRAPH IF THE NUMBER IS GREATER THAN ZERO.
1363 C
1364     IF(LLABPL(NPL)) THEN
1365         NDVXLL = NTICX(NPL)
1366         NDVYLL = NTICY(NPL)
1367     ELSE
1368         NDVXLL = NDVX

```

```

1369     NDVYLL = NDVY
1370     ENDIF
1371     IF (NDVXLL.GT.1) THEN
1372         DELX = (X2R - X1R)/FLOAT(NDVXLL)
1373         DO 3 I = 1, NDVXLL-1
1374             XX = X1R + FLOAT(I)*DELX
1375     3   CALL LINE (XX, Y1R, XX, Y1R+C(16))
1376         DO 7 I = 1, NDVXLL-1
1377             XX = X1R + FLOAT(I)*DELX
1378     7   CALL LINE(XX, Y2R, XX, Y2R-C(16))
1379     END IF
1380     IF (NDVYLL.GT.1) THEN
1381         DELY = (Y2R - Y1R)/FLOAT(NDVYLL)
1382         DO 4 I = 1, NDVYLL-1
1383             YY = Y1R + FLOAT(I)*DELY
1384     4   CALL LINE (X1R, YY, X1R+C(15), YY)
1385         DO 8 I = 1, NDVYLL-1
1386             YY = Y1R + FLOAT(I)*DELY
1387     8   CALL LINE(X2R, YY, X2R-C(15), YY)
1388     END IF
1389     RETURN
1390 C
1391     ENTRY PLOTCH (NPL, X, Y, N, PCHR)
1392 C
1393     IF (N.LE.0) RETURN
1394     IF(NPL.GT.NPLTMX.OR.NPL.LT.1) STOP 72
1395     IF(.NOT.PLTON(NPL)) RETURN
1396     DX = (HOLD(4,NPL)-HOLD(3,NPL)) / (HOLD(6,NPL)-HOLD(5,NPL))
1397     DY=(HOLD(10,NPL)-HOLD(9,NPL))/(HOLD(12,NPL)-HOLD(11,NPL))
1398     CHSIZE(2) = (HOLD(10,NPL)-HOLD(9,NPL)) * XYCOORD(3) * CHSIZE(5)
1399     NLM = 1
1400     13 IST = NLM
1401     NLM = MIN(NLM-1+N LINES, N)
1402     DO 5 I = IST, NLM
1403         WX(I-IST+1) = HOLD(3,NPL) + DX * (X(I)-HOLD(5,NPL))
1404     5   WY(I-IST+1) = HOLD(9,NPL) + DY * (Y(I)-HOLD(11,NPL))
1405         CALL CHPLOT (WX, WY, PCHR, 1, 1, NLM-IST+1)
1406         IF(NLM.LT.N) GO TO 13
1407         CHSIZE(2) = CHSIZE(1)
1408     RETURN
1409 C
1410     ENTRY PLOTLN (NPL, X, Y, N)
1411 C
1412     IF (N.LE.0) RETURN
1413     IF(NPL.GT.NPLTMX.OR.NPL.LT.1) STOP 72
1414     IF(.NOT.PLTON(NPL)) RETURN
1415     DX = (HOLD(4,NPL)-HOLD(3,NPL)) / (HOLD(6,NPL)-HOLD(5,NPL))
1416     DY=(HOLD(10,NPL)-HOLD(9,NPL))/(HOLD(12,NPL)-HOLD(11,NPL))
1417     NLM = 1
1418     14 IST = NLM
1419     NLM = MIN(NLM-1+N LINES,N)
1420     DO 6 I = IST, NLM
1421         WX(I-IST+1) = HOLD(3,NPL) + DX * (X(I)-HOLD(5,NPL))
1422     6   WY(I-IST+1) = HOLD(9,NPL) + DY * (Y(I)-HOLD(11,NPL))
1423         CALL LNPLLOT (WX, WY, 1, 1, NLM-IST+1)

```

```

1424     IF(NLM.GE.N) RETURN
1425     GO TO 14
1426 C
1427     ENTRY PLOTBAR (NPL, XL, YB, XR, YT, N, VALUE)
1428 C
1429     IF (N.LE.0) RETURN
1430     IF(NPL.GT.NPLTMX.OR.NPL.LT.1) STOP 72
1431     IF(.NOT.PLOTON(NPL)) RETURN
1432     DX = (HOLD(4,NPL)-HOLD(3,NPL)) / (HOLD(6,NPL)-HOLD(5,NPL))
1433     DY = (HOLD(10,NPL)-HOLD(9,NPL)) / (HOLD(12,NPL)-HOLD(11,NPL))
1434     NLM = 1
1435 20 IST = NLM
1436     NLM = MIN(NLM-1+NLINES,N)
1437     CALL LINWID(1)
1438     DO 21 I = IST, NLM
1439     X1 =      HOLD(3,NPL) + DX * (XR(I)-HOLD(5,NPL))
1440     X0 =      HOLD(3,NPL) + DX * (XL(I)-HOLD(5,NPL))
1441     Y1 =      HOLD(9,NPL) + DY * (YT(I)-HOLD(11,NPL))
1442     Y0 =      HOLD(9,NPL) + DY * (YB(I)-HOLD(11,NPL))
1443     WX(1) = X0
1444     WX(2) = X0
1445     WX(3) = X1
1446     WX(4) = X1
1447     WY(1) = Y0
1448     WY(2) = Y1
1449     WY(3) = Y1
1450     WY(4) = Y0
1451     CALL PLYGON(WX, WY, 4)
1452     TX = X1 - X0
1453     TY = HOLD(10,NPL) - HOLD(9,NPL)
1454     IDIGIT = 0
1455     IF (ABS(VALUE).LT.10.) IDIGIT = 1
1456     NDIG = 3
1457     IF(ABS(VALUE).GE.1000.) NDIG = 4
1458     YOFF = Y1 - .02 * TY
1459     IF (VALUE.LT.0.0) YOFF = Y0 - 0.15 * TY
1460     FAC = 1. / FLOAT(NDIG)
1461     XST = FAC * .95
1462     IF(VALUE.NE.0.0) CALL FNUMBR(X0+0.01*TX,YOFF, ; -.025->+.01
1463     . FAC*TX,0.0,XST*TX,0.0,0.12*TY,ABS(VALUE),NDIG,IDIGIT)
1464 21 CONTINUE
1465     CALL LINWID(3)
1466     IF(NLM.GE.N) RETURN
1467     GO TO 20
1468 C
1469     ENTRY GRISET (XL, YB, XR, YT)
1470 C
1471     CALL DEFINE (XL, YB, XR, YT)
1472     RETURN
1473 C
1474     ENTRY GRILAB (NPL, XLL, YBT, XRT, YTT)
1475     DX = (XRT-XLL) / 3.
1476     DY = (YTT-YBT) / 3.
1477     XINT1 = 10.**(INT(ALOG10(DX)))
1478     YINT1 = 10.**(INT(ALOG10(DY)))

```



```
1479     XINT2 = DX / XINT1
1480     YINT2 = DY / YINT1
1481     IF(XINT2.LT.2.0) THEN
1482         DX = XINT1
1483     ELSE IF (XINT2.LT.5.0) THEN
1484         DX = 2.*XINT1
1485     ELSE IF (XINT2.LT.10.) THEN
1486         DX = 5.*XINT1
1487     ELSE
1488         DX = 10.*XINT1
1489     ENDIF
1490     IF(YINT2.LT.2.0) THEN
1491         DY = YINT1
1492     ELSE IF (YINT2.LT.5.0) THEN
1493         DY = 2.*YINT1
1494     ELSE IF (YINT2.LT.10.) THEN
1495         DY = 5.*YINT1
1496     ELSE
1497         DY = 10.*YINT1
1498     ENDIF
1499     XINT2 = XLL / DX
1500     YINT2 = YBT / DY
1501     IF(XINT2.LT.0.0) XINT2 = XINT2 - 0.99999
1502     IF(YINT2.LT.0.0) YINT2 = YINT2 - 0.99999
1503     XMINO(NPL) = DX * (INT(ABS(XINT2))*SIGN(1.,XINT2))
1504     YMINO(NPL) = DY * (INT(ABS(YINT2))*SIGN(1.,YINT2))
1505     XINT2 = XRT / DX
1506     YINT2 = YTT / DY
1507     IF(XINT2.GT.0.0) XINT2 = XINT2 + 0.99999
1508     IF(YINT2.GT.0.0) YINT2 = YINT2 + 0.99999
1509     XMAXO(NPL) = DX * (INT(ABS(XINT2))*SIGN(1.,XINT2))
1510     YMAXO(NPL) = DY * (INT(ABS(YINT2))*SIGN(1.,YINT2))
1511     NTICX(NPL) = (XMAXO(NPL)-XMINO(NPL)+0.5) / DX
1512     NTICY(NPL) = (YMAXO(NPL)-YMINO(NPL)+0.5) / DY
1513     LLABPL(NPL) = .TRUE.
1514     RETURN
1515     END
```

```

1516 SUBROUTINE HHDRAW (XCHAR, YCHAR, SX,SXY,SY, CHNUMB, SET, IERR)
1517 C
1518 C XCHAR REAL - X STARTING COORDINATE OF THE CHARACTER TO BE
1519 C DRAWN
1520 C YCHAR REAL - Y STARTING COORDINATE OF THE CHARACTER TO BE
1521 C DRAWN
1522 C SX REAL - X MATH SPACE SIZE FOR CHARACTERS
1523 C SXY REAL - SLANT MODIFIER FOR CHARACTERS
1524 C SY REAL - Y MATH SPACE SIZE FOR CHARACTERS
1525 C CHNUMB INTEGER - INDEX TO IDENTIFY CHARACTERS WITHIN THE SPECI-
1526 C FIED SET
1527 C SET INTEGER - NUMBER SPECIFYING A PARTICULAR CHARACTER SET
1528 C
1529 C
1530 REAL XCHAR,YCHAR,SX,SXY,SY,X(128),Y(128)
1531 INTEGER M, CHNUMB,SET,PEN(128),ERROR,IERR
1532 CHARACTER*1 CHRIRV
1533 COMMON /DEVTYP/ IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
1534 COMMON/GRFTYP/ANGLE,IRVRSE,CHSIZE(9),ITKWIT,LUDIAG,LUHSET
1535 C
1536 C
1537 IERR = 1
1538 IF (SET.GT.1) GO TO 3
1539 IF (SET.LT.1) RETURN
1540 C
1541 IRV = IREVCH(CHNUMB, CHRIRV)
1542 CALL SYMBOL(XCHAR+CHSIZE(7),YCHAR+CHSIZE(8),CHRIRV)
1543 RETURN
1544 C
1545 3 CALL HSETS (CHNUMB,X,Y,PEN,N,SET-1,ERROR)
1546 IF(ERROR.NE.0) RETURN
1547 IERR = 0
1548 C
1549 M = N
1550 IF (M.EQ.0) RETURN
1551 C
1552 C ROTATE COORDINATES IF ANGLE>0.0
1553 C
1554 SXP = ABS(XYCOORD(1))
1555 SYP = ABS(XYCOORD(3))
1556 OSXP = 1. / SXP
1557 OSYP = 1. / SYP
1558 SXP = SXP * SX
1559 SYP = SYP * SY
1560 DO 9 I = 1, M
1561 X(I) = X(I) * 0.216 * SXP
1562 9 Y(I) = Y(I) * 0.0800 * SYP + X(I) * SXY
1563 IF (ANGLE.EQ.0.0) GO TO 4
1564 SINX = SIN(ANGLE)
1565 COSX = COS(ANGLE)
1566 DO 6 I = 1, M
1567 XP = X(I)*COSX - Y(I)*SINX
1568 YP = X(I)*SINX + Y(I)*COSX
1569 X(I) = XP
1570 6 Y(I) = YP

```

```
1571      4      DO 1 I = 1, M
1572          X(I) = X(I) * OSXP
1573      1      Y(I) = Y(I) * OSYP
1574      C
1575      C      ADD THE ABSOLUTE POSITION
1576      C
1577          DO 7 I = 1, M
1578          X(I) = X(I) + XCHAR
1579      7      Y(I) = Y(I) + YCHAR
1580      C
1581      C      WRITE THE CHARACTER AS A SET OF CONNECTED STROKES
1582      C
1583          IS = 1
1584          IL = IS
1585      10     IF(PEN(IL+1).EQ.0.OR.IL.GE.M) GO TO 2
1586          IL = IL + 1
1587          GO TO 10
1588      2     IF(IL.GT.IS) CALL LINES(X(IS),Y(IS),X(IS+1),Y(IS+1),IL-IS)
1589          IF(IL.GE.M) RETURN
1590          IL = IL + 1
1591          IS = IL
1592          GO TO 10
1593          RETURN
1594      END
```

```

1595     SUBROUTINE HSETS (CHNUMB,XX,YY,PEN,N,SET,ERROR)
1596 C
1597 C*****
1598 C   HSETS
1599 C*****
1600 C
1601 C THIS SUBROUTINE RECEIVES A SET NUMBER AND A CHARACTER NUMBER
1602 C AND SEARCHES THROUGH 'HSETS.DAT' (FILE CONTAINING THE PACKED
1603 C HERSHEY CHARACTER SUBSETS) TO FIND THE CHARACTER'S COORDINATES
1604 C AND CORRESPONDING 'PEN' VALUES.
1605 C
1606     PARAMETER(MAXSET=23,MAX1=24)
1607     LOGICAL FOPEN/.FALSE./
1608     INTEGER IN(125,96),IOS,CHAR,SET,COORD,ERROR
1609     INTEGER CHNUMB,PEN(125),MASK1,MASK2,TABLE(MAX1)
1610     INTEGER FIRST,LAST,NCHAR(MAXSET),TT(64),IOB(64),IOBC
1611     EQUIVALENCE (TT(1),TABLE(1)),(TT(33),NCHAR(1))
1612     DATA MASK2/ZFFFF/,MASK1/Z7FFFFFFF/,NSET/-1/
1613     REAL XX(125),YY(125)
1614     COMMON/GRFTYP/ANGLE,IRVRSE,CHSIZE(9),ITKWIT,LUDIAG,LUHSET
1615 C
1616 C OPEN FILE CONTAINING PACKED HERSHEY CHARACTER SETS IF NOT
1617 C PREVIOUSLY OPENED
1618 C
1619     IF(SET.EQ.NSET.AND.FOPEN) GO TO 41
1620     CLOSE(LUHSET)
1621 C
1622 C OPEN A NEW FILE FOR THE PACKED HERSHEY CHARACTER SUBSETS
1623 C
1624     OPEN(UNIT=LUHSET,IOSTAT=IOS,ERR=50,FILE='SYS:H1.CAT/S',
1625 * ACCESS='DIRECT',FORM='BINARY',RECL=256,SIZE=730,
1626 * BLOCKSIZE=256,TYPE='CONTIG')
1627     REWIND LUHSET
1628     FOPEN=.TRUE.
1629     IF(LUDIAG.NE.0) WRITE(LUDIAG,20) LUHSET
1630 20     FORMAT(' HSET CHARACTER FILE OPENED TO UNIT =',I3)
1631 C
1632 C READ STARTING RECORD POSITIONS OF THE SETS INTO 'TABLE'
1633 C READ THE NUMBER OF CHARACTERS IN EACH SET INTO 'NCHAR'
1634 C
1635     READ(LUHSET) TT
1636 C
1637 C CHECK THAT THE SET AND CHARACTER NUMBERS ARE VALID
1638 C
1639     ERROR=0
1640     IF(SET.GT.MAXSET) THEN ;SET NOT IN FILE
1641     ERROR=1
1642     GO TO 52
1643     ENDIF
1644     IF(CHNUMB.GT.NCHAR(SET)) THEN ,CHARACTER NOT IN SET
1645     ERROR=2
1646     GO TO 52
1647     ENDIF
1648 C
1649 C READ THE CHARACTERS' COORDINATES INTO THE 'IN' ARRAY

```

```

1650 C
1651     NSET = SET
1652     FIRST=TABLE(SET) ;FIRST RECORD OF SET
1653     LAST=TABLE(SET+1) - 1 ;LAST RECORD OF SET
1654     CHAR=1
1655     COORD=0
1656     DO 30 I=FIRST, LAST
1657     READ(LUHSET, REC=I) IOB
1658 C
1659 C STORE DATA IN ARRAY ELEMENTS FOR 1 CHARACTER AT A TIME
1660 C
1661     J=0
1662 32     J=J+1
1663     COORD=COORD+1
1664     IN(COORD, CHAR)=IOB(J)
1665     IF((IOB(J).NE.MASK1).AND.(J.LT.64)) GO TO 32
1666 C
1667 C END OF CHARACTER FOUND; READ COORDINATES FOR NEXT CHARACTER
1668 C
1669     IF(IOB(J).EQ.MASK1) THEN
1670         CHAR=CHAR+1
1671         COORD=0
1672         IF(CHAR.GT.NCHAR(SET)) GO TO 31
1673     ENDIF
1674     IF(J.LT.64) GO TO 32
1675 30     CONTINUE ;READ NEXT RECORD
1676 31     CLOSE(LUHSET)
1677 C
1678 C UNPACK THE DATA FOR THE SPECIFIED CHARACTER & DETERMINE THE REAL
1679 C VALUES OF THE COORDINATES AND THE INTEGER VALUE OF THE PEN.
1680 C
1681 41     N=0
1682 40     IF(IN(N+1, CHNUMB).EQ.MASK1) RETURN ;END OF CHARACTER
1683     N=N+1
1684 C
1685 C FIND THE 'PEN' VALUE & ELIMINATE 'PEN' BIT FROM PACKED WORD
1686 C
1687     PEN(N)=ISHFT(IN(N, CHNUMB), -31)
1688     IOBC=IAND(IN(N, CHNUMB), MASK1)
1689 C
1690 C FIND THE X & Y COORDINATES
1691 C
1692     XX(N)=(ISHFT(IOBC, -16))/100.00
1693     YY(N)=(IAND(IN(N, CHNUMB), MASK2))/100.0
1694     GO TO 40 ;UNPACK NEXT COORDINATE AND PEN VALUE
1695 C
1696 C ERROR
1697 C
1698 50     ERROR = IOS
1699     IF(LUDIAG.NE.0) WRITE(LUDIAG, 51) IOS
1700 51     FORMAT(' ERROR IN HSET - UNABLE TO OPEN CHARACTER FILE, ERROR = ',
1701     . I4)
1702 52     CLOSE(LUHSET)
1703     RETURN
1704     END

```

```

1705          INTEGER FUNCTION IDCHAR (CHARX)
1706 C
1707 C*****
1708 C      IDHCAR
1709 C*****
1710 C
1711          LOGICAL ZEROP/.TRUE./
1712          INTEGER  IDN(128), CHAR, KA, MASK/127/
1713          CHARACTER*1 CHARX, CHARXX(4), JA(4), CHRIRV
1714          CHARACTER*8  ASCII(16)
1715          CHARACTER*1  ASCIII(128)
1716          EQUIVALENCE (CHAR,CHARXX)
1717          EQUIVALENCE (JA,KA), (ASCII,ASCIII)
1718          DATA ASCII/'01234567','89ABCDEF','GHIJKLMN','OPQRSTUVWXYZ',
1719          . 'WXYZabcd','efghijkl','mnopqrst','uvwxyz %',
1720          . '&@ $#. , ; , ' ! ? - + = * / ( , ' ) [ ] $ % ^ &lt; ' , ' > ° ' ' _ . - ' ,
1721          . 4 * ' ' /
1722 C
1723 C      THIS ROUTINE INITIALIZES THE IDENTITY ARRAY ID TO THE IDENTITY
1724 C      NUMBER OF THE ACCEPTED ASCII TERMINAL CHARACTERS.
1725 C
1726          IF(ZEROP) THEN
1727              DO 2 I = 1, 128
1728          2      IDN(I) = 127
1729              DO 1 I = 1, 128
1730          1      JA(4) = ASCIII(I)
1731              KA = IAND(KA, MASK)
1732          1      IF(IDN(KA+1).EQ.127) IDN(KA+1) = I
1733              ZEROP = .FALSE.
1734          ENDIF
1735 C
1736 C      THIS INTEGER FUNCTION RETURNS THE IDENTITY NUMBER OF THE ONE
1737 C      BYTE CHARACTER ENTERED THROUGH THE ARGUMENT LIST AS A CHARACTER*1
1738 C      VARIABLE.
1739 C
1740          CHARXX(4) = CHARX
1741          CHAR = IAND(CHAR, MASK)
1742          IDCHAR = IDN(CHAR+1)
1743          RETURN
1744 C
1745 C      THIS ENTRY REVERSES THE CHARACTER PROCESS - USED BY HHDRAW WHEN
1746 C      CALLING SYMBOL. CHANGES THE NUMBER BACK INTO A CHARACTER
1747 C
1748          ENTRY IREVCH (ICHR, CHRIRV)
1749          IREVCH = 0
1750          IF(ICHR.GE.1.AND.ICHR.LE.128) CHRIRV = ASCIII(ICHR)
1751          RETURN
1752          END

```

```
1753     SUBROUTINE LABEL (CHARS, XL, YB, XR, YT, ANGLE)
1754     COMMON /DEVTYP/ IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
1755     COMMON/GRFTYP/XNGLE,IRVRSE,CHSIZE(9),ITKWIT,LUDIAG,LUHSET
1756     CHARACTER*1 CHARS(*)
1757     CALL WDCOUNT(CHARS, NC)
1758     IF(NC.LE.0) THEN
1759         IF(LUDIAG.GT.0) WRITE(LUDIAG,3) (CHARS(I),I=1,132)
1760     3     FORMAT(' NO ESCAPE SEQUENCE IN LABEL -',/,1X,132A1)
1761         RETURN
1762     ENDIF
1763     DX = (XR - XL) / MAX(1.,FLOAT(NC))
1764     DY = (YT - YB)
1765     XNGLE = ANGLE
1766     CALL WDDRAW (XL, YB, DX, 0.0, DX, 0.0, DY, CHARS)
1767     RETURN
1768     END
```

```
1769          SUBROUTINE LINE (X1, Y1, X2, Y2)
1770 C
1771 C*****
1772 C      LINE
1773 C*****
1774 C
1775          COMMON /DEV TYP/ IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
1776 C
1777 C      THIS ROUTINE PLOTS A LINE FROM (X1,Y1) TO (X2,Y2).
1778 C      THE PRESENT ALGORITHM ALWAYS PLOTS FROM THE CURRENT LOCATION TO
1779 C      THE TARGET POINT THUS INSURING PERFECT LINE CONTINUITY.
1780 C
1781 C
1782          REAL IX1, IX2, IY1, IY2, IDX, IDY
1783 C
1784          IX1 = XYCOORD(1)*X1 + XYCOORD(2)
1785          IX2 = XYCOORD(1)*X2 + XYCOORD(2)
1786          IY1 = XYCOORD(3)*Y1 + XYCOORD(4)
1787          IY2 = XYCOORD(3)*Y2 + XYCOORD(4)
1788          IDX =IX2 - IX1
1789          IDY = IY2 - IY1
1790          CALL PUTLVN (IX1,IY1,IDX,IDY,1)
1791          RETURN
1792          END
```



```

1793          SUBROUTINE LINES(X1,Y1,X2,Y2,NL)
1794 C
1795 C          THIS SUBROUTINE PLOTS LINES FROM (X1(J),Y1(J)) TO (X2(J),Y2(J))
1796 C          WHILE CONVERTING TO RASTER NUMBERS THROUGH SCALE.
1797 C
1798 C*****
1799 C          LINES
1800 C*****
1801 C
1802          PARAMETER (NLINES=200)
1803          COMMON /DEVTYP/ IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
1804          COMMON/GRFTYP/ANGLE,IRVRSE,CHSIZE(9),ITKWIT,LUDIAG,LUHSET
1805          REAL X1(NL),X2(NL),Y1(NL),Y2(NL)
1806          REAL IX1(NLINES), IY1(NLINES), IDX(NLINES), IDY(NLINES)
1807 C
1808 C
1809          NLM = NL
1810          I2 = 1
1811          5      I1 = I2
1812          I2 = MINO(I2-1+NLINES,NL)
1813          NLMIN = I2 - I1 + 1
1814          DO 10 I = I1, I2
1815          IX1(I-I1+1)=XYCOORD(1)*X1(I)+XYCOORD(2)
1816          IDX(I-I1+1)=XYCOORD(1)*X2(I)+XYCOORD(2)-IX1(I-I1+1)
1817          IY1(I-I1+1)=XYCOORD(3)*Y1(I)+XYCOORD(4)
1818          10      IDY(I-I1+1)=XYCOORD(3)*Y2(I)+XYCOORD(4)-IY1(I-I1+1)
1819 C
1820          CALL PUTLNV(IX1,IY1,IDX,IDY,NLMIN)
1821          IF (I2.GE.NL) RETURN
1822          GO TO 5
1823          RETURN
1824          END

```

```
1825      SUBROUTINE LNPLLOT (XARRAY, YARRAY, I1, I2, I3)
1826 C
1827      PARAMETER (NLINES=200)
1828      REAL      XARRAY(1), YARRAY(1)
1829      REAL XA(NLINES),YA(NLINES),XB(NLINES),YB(NLINES)
1830      INTEGER  I1, I2, I3, IST, I, NOPTS, NOPT
1831 C
1832      NOPTS=(I3-I1+I2)/I2-1
1833      IST=I1
1834      NOPT=NOPTS
1835      NOPT = MINO (NLINES,NOPT)
1836      DO 1 I=1,NOPT
1837      XA(I)=XARRAY(IST+I2*I-I2)
1838      YA(I)=YARRAY(IST+I2*I-I2)
1839      XB(I)=XARRAY(IST+I2*I)
1840      YB(I)=YARRAY(IST+I2*I)
1841      1 CONTINUE
1842      CALL LINES(XA,YA,XB,YB,NOPT)
1843      NOPTS=NOPTS-NOPT
1844      IST=IST+I2*NOPT
1845      IF(NOPTS.GT.0) GO TO 2
1846      RETURN
1847      END
```

```
1852     SUBROUTINE MAPIN (MAPFIL,  
1853     . WIN, XYZ, NV, NE, NES, NEX, NP, NPS, NPX)  
1854 C  
1855     PARAMETER (NVM=4000,NEM=400,NPM=400)  
1856     CHARACTER*16 MAPFIL  
1857     REAL XYZ(3,NV), WIN(4)  
1858     INTEGER NP(8,NPM), NPS(4,NPM), NE(2,NEM), NES(4,NEM)  
1859     LOGICAL FIRSTC/.TRUE./  
1860 C  
1861 C     INITIALIZE THE COUNTERS THE FIRST TIME THRU THIS ROUTINE  
1862 C  
1863     IF(FIRSTC) THEN  
1864         NV = 0  
1865         NEX = 0  
1866         NPX = 0  
1867         FIRSTC = .FALSE.  
1868     ENDIF  
1869 C  
1870 C     OPEN AND INPUT A STRUCTURES FILE  
1871 C  
1872     NV1 = NVM  
1873     NE1 = NEM  
1874     NP1 = NPM  
1875     CALL PLYPLT (MAPFIL,0)  
1876     CALL PLYGNS(WIN, XYZ(1,NV+1), NV1, NE(1,NEX+1), NES(1,NEX+1),  
1877     . NE1, NP(1,NPX+1), NPS(1,NPX+1), NP1)  
1878     IF(NV1.LE.0) RETURN  
1879 C  
1880 C     UPDATE THE EDGE AND POLYGON POINTERS  
1881 C  
1882     DO 132 I = NEX+1, NEX+1+NE1  
1883     DO 132 J = 1, 2  
1884 132 NE(J,I) = NE(J,I) + NV  
1885     DO 133 I = NPX+1, NPX+1+NP1  
1886     DO 133 J = 1, 8  
1887     IF(NP(J,I).GT.0) NP(J,I) = NP(J,I) + NV  
1888 133 CONTINUE  
1889     NV = NV + NV1  
1890     NEX = NEX + NE1  
1891     NPX = NPX + NP1  
1892     RETURN  
1893     END
```

```
1894     SUBROUTINE MAPOUT(WIN, XYZ, NV, NE, NES, NEX, NP, NPS, NPX)
1895 C
1896 $INCLUDE BUILD.COM (NLIST)
1897     REAL XYZ(3,NV), WIN(4)
1898     INTEGER NP(NPVERT,NPX), NPS(NSPEC,NPX), NE(2,NEX), NES(NSPEC,NEX)
1899 C
1900     CALL DEFINE(WIN(1),WIN(2),WIN(3),WIN(4))
1901     CALL PLYPL4(NE,NEX,XYZ,WIN,NES)
1902     CALL PLYPL5(NP,NPX,XYZ,WIN,NPS,NPVERT)
1903     CALL FRAME
1904     RETURN
1905     END
```

```
1906          SUBROUTINE PLYGON(X, Y, N)
1907 C
1908 C      DRAW A COMPLETE POLYGON AND FILL IT WITH 'LFLMAT'
1909 C
1910          PARAMETER (NPMAX=32)
1911          COMMON/DEVTYP/IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
1912          COMMON/GRFMOD/LFLMAT(5),LINWDM(5),LCLMAT(5)
1913          DIMENSION X(N), Y(N), XL(2*NPMAX)
1914 C
1915          IF(IDEVIC.EQ.0) RETURN
1916          NL = MINO(NPMAX, N)
1917          GO TO (1,1,3,1,1), IDEVIC
1918          1  DO 11 I = 1, N-1
1919          11 CALL LINE(X(I), Y(I), X(I+1), Y(I+1))
1920          CALL LINE(X(N), Y(N), X(1), Y(1))
1921          RETURN
1922          3  DO 31 I = 1, NL
1923          XL(2*I-1) = MAX(X(I) * XYCOORD(1) + XYCOORD(2), 0.0)
1924          31 XL(2*I) = MAX(Y(I) * XYCOORD(3) + XYCOORD(4), 0.0)
1925          CALL GMOVE(XL(1), XL(2))
1926          CALL GPOLA (XL(3), NL-1)
1927          RETURN
1928          END
```

```

1929     SUBROUTINE PLYPL1(CHOOSE,ENDSTR,SSTART,SFIRST,SLAST,SVALID,
1930     IIBLANK)
1931 C
1932 C     THIS ROUTINE FINDS POSITIONS OF SUBSTRINGS WITHIN A CHARACTER
1933 C     STRING.  COMMAS AND SPACES ARE SUBSTRING DELIMITERS.  WHEN
1934 C     A VALID SUBSTRING IS FOUND, ITS FIRST AND LAST CHARACTER
1935 C     POSITIONS ARE RETURNED TO THE CALLING ROUTINE.
1936 C     THIS ROUTINE ALSO INCLUDES THE OPTION OF CONSIDERING A GROUP
1937 C     OF BLANKS SET OF BY COMMAS TO BE A VALID SUBSTRING.  IF SUCH
1938 C     A SUBSTRING IS FOUND, THE POSITIONS OF THE FIRST AND LAST
1939 C     BLANKS ARE NOT RETURNED, BUT RATHER THE POSITIONS OF THE
1940 C     DELIMITERS (COMMAS).
1941 C
1942 C     VARIABLES USED:
1943 C     CFIND = POSITION OF 1ST COMMA FOUND (IF 'BLANK' OPTION SET)
1944 C     CHOOSE = INTEGER CONTAINING CHARACTER STRING
1945 C     ENDSTR = NUMBER OF CHARACTERS IN THE ENTIRE STRING
1946 C     IBLANK = 'BLANK' FLAG
1947 C         (input) = 0 - BLANKS NOT VALID SUBSTRING
1948 C         (input) = 1 - BLANKS FOLLOWED BY 1 COMMA - VALID SUBSTRING
1949 C         (input) = 2 - BLANKS FOLLOWED BY 2 COMMAS - VALID SUBSTRING
1950 C         (output)= -1,-2 - SPECIFIED BLANKS (1 OR 2) FOUND
1951 C     SFIRST = POSITION OF FIRST CHARACTER OF SUBSTRING
1952 C     SLAST  = POSITION OF LAST CHARACTER OF SUBSTRING
1953 C     SSTART = STARTING POSITION FOR SUBSTRING SEARCH
1954 C     STRING = CHARACTER ARRAY CONTAINING THE CHARACTER STRING
1955 C     SVALID = 'VALID SUBSTRING' FLAG
1956 C             = TRUE - VALID SUBSTRING FOUND
1957 C             = FALSE - NO VALID SUBSTRING FOUND
1958 C
1959 C     LOGICAL SVALID
1960 C     INTEGER SFIRST,SLAST,SSTART,ENDSTR
1961 C     INTEGER IBLANK,CFIND,CHOOSE(20),STR(20)
1962 C     CHARACTER*1 STRING(80),SPACE/' '/,CR/ZD/,COMMA/Z2C/
1963 C     EQUIVALENCE (STR,STRING)
1964 C
1965 C     ASSUME VALID SUBSTRING
1966 C
1967 C     SVALID=.TRUE.
1968 C
1969 C     FORM AN EQUIVALENT CHARACTER STRING
1970 C
1971 C     DO 10 I=1,20
1972 10    STR(I)=CHOOSE(I)
1973 C
1974 C     INVALID STARTING POSITION - PAST END OF STRING
1975 C
1976 C     IF(ENDSTR.LT.SSTART) GO TO 40
1977 C
1978 C     FIND POSITION OF FIRST ELEMENT OF SUBSTRING
1979 C
1980 C     CFIND=0
1981 C     DO 20 I=SSTART,ENDSTR
1982 C     SFIRST=I
1983 C

```

```

1984 C   A COMMA FOUND
1985 C
1986 C   IF (IBLANK.NE.0.AND.STRING(I).EQ.COMMA) THEN
1987 C
1988 C       FOUND FIRST COMMA
1989 C
1990 C       IF (CFIND.EQ.0) THEN
1991 C           CFIND=I
1992 C           IF (IBLANK.EQ.2) GO TO 20
1993 C       ENDIF
1994 C
1995 C       FOUND BLANK SUBSTRING; NOTE POSITIONS OF DELIMITERS (COMMAS)
1996 C
1997 C       SLAST=SFIRST
1998 C       SFIRST=CFIND
1999 C       IBLANK=-IBLANK
2000 C       GO TO 100
2001 C   ENDIF
2002 C
2003 C   FOUND FIRST CHARACTER OF SUBSTRING - NOW FIND LAST
2004 C
2005 C   IF((STRING(I).NE.SPACE).AND.(STRING(I).NE.CR).AND.
2006 C   I (STRING(I).NE.COMMA)) GO TO 60
2007 20  CONTINUE
2008 C
2009 C   NO SUBSTRING FOUND - ONLY DELIMITER
2010 C
2011 C   WRITE(6,998)
2012 998  FORMAT(' REACHED THE END WITHOUT FINDING A NON-BLANK CHARACTER')
2013 C   GO TO 40
2014 C
2015 C   FIND POSITION OF LAST CHARACTER OF SUBSTRING
2016 C
2017 60  IF (SFIRST.EQ.ENDSTR) GO TO 45
2018 C   DO 50 J=SFIRST+1,ENDSTR
2019 C   SLAST=J-1
2020 C
2021 C   FOUND SUBSTRING DELIMITER - CAN RETURN NOW
2022 C
2023 C   IF((STRING(J).EQ.SPACE).OR.(STRING(J).EQ.COMMA)) GO TO 100
2024 50  CONTINUE
2025 C
2026 C   NO SUBSTRING DELIMITER => LAST CHARACTER OF SUBSTRING IS THE
2027 C   LAST CHARACTER OF THE STRING
2028 C
2029 45  SLAST=ENDSTR
2030 C   GO TO 100
2031 C
2032 C   NO SUBSTRING FOUND
2033 C
2034 40  SVALID=.FALSE.
2035 100  RETURN
2036 C   END

```

```

2037     SUBROUTINE PLYPL2(WINDOW,OPENN,WORLD)
2038 C
2039 C READ WINDOW COORDINATES AND STORE THEM IN 'WINDOW' ARRAY. ALSO
2040 C CHECK VALIDITY OF COORDINATES. IF VALID, 'OPENN' IS TRUE;
2041 C OTHERWISE, 'OPENN' IS FALSE. WINDOW COORDINATES WILL BE STORED
2042 C AS FOLLOWS:
2043 C
2044 C
2045 C
2046 C
2047 C
2048 C
2049 C
2050     LOGICAL OPENN
2051     REAL WINDOW(4),WORLD(6)
2052 C
2053 C INITIALIZE
2054 C
2055     OPENN=.FALSE. ;ASSUME INVALID COORDINATES
2056 C
2057 C CHECK VALIDITY OF WINDOW COORDINATES
2058 C
2059     DO 30 I=1,2
2060     IF(WINDOW(I).GE.WINDOW(I+2)) RETURN ;INVALID COORDINATES
2061 30 CONTINUE
2062 C
2063 C WINDOW COORDINATES VALID
2064 C
2065     OPENN=.TRUE.
2066     RETURN •
2067     END

```



```

2068      SUBROUTINE PLYPL3(ARRAY,NARRAY,SP,NPNT,NVERT)
2069 C
2070 C THIS SUBROUTINE PLOTS VERTICES FOUND WITHIN THE WINDOW AND THEIR
2071 C CORRESPONDING LINE NUMBERS.
2072 C
2073      INTEGER CHARAC/'.'/ ,NARRAY,CHR,NPNT(NVERT),DIGITS
2074      REAL ARRAY(3,NARRAY),SP(4),NUM,NUMBR,SSP(4)
2075      REAL NX,NY,XINC,XSTART
2076      COMMON/PLSCMR/DOZ,WCX,WCY,FANGLE,TANAL
2077      COMMON/DEVTYP/IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
2078 C
2079 C DEFINE THE WINDOW SPACE
2080 C
2081      DO 1 I = 1, 4
2082 1      SSP(I) = XYCOORD(I)
2083      CALL DEFINE(SP(1),SP(2),SP(3),SP(4))
2084 C
2085 C PLOT 1 VERTEX AND NUMBER AT A TIME
2086 C
2087      DO 20 I=1,NARRAY
2088 C
2089 C DO NOT PLOT A 'DELETED' VERTEX
2090 C
2091      IF(NPNT(I).EQ.0) GO TO 20
2092 C
2093 C ADD PERSPECTIVE
2094 C
2095      D = DOZ/((DOZ+ARRAY(3,I))*TANAL)
2096      XX = (ARRAY(1,I)-WCX)*D + WCX
2097      YY = (ARRAY(2,I)-WCY)*D + WCY
2098 C
2099 C DO NOT PLOT A VERTEX WHICH IS OUTSIDE OF WINDOW
2100 C
2101      IF((XX.LT.SP(1)).OR.(XX.GT.SP(3))) GO TO 20
2102      IF((YY.LT.SP(2)).OR.(YY.GT.SP(4))) GO TO 20
2103 C
2104 C PLOT THE VERTEX USING THE CHARACTER MODE OF THE HARDWARE
2105 C
2106      CALL SYMBOL(XX,YY,CHARAC)
2107 C
2108 C SCALE MULTIPLIERS FOR NUMBERS
2109 C
2110      XINC=0.025*(SP(3)-SP(1))
2111      NX=0.035*(SP(3)-SP(1))
2112      NY=0.035*(SP(4)-SP(2))
2113      NUMBR=I
2114 C
2115 C DETERMINE NUMBER OF DIGITS 'NUM' (THE VERTEX LINE NUMBER)
2116 C
2117      NUM=I
2118      DIGITS=0
2119 10      NUM=NUM/10.0
2120      DIGITS=DIGITS+1
2121      IF (NUM.GE.1) GO TO 10
2122 C

```

```
2123 C CALCULATE X COORDINATE OF FIRST DIGIT OF THE NUMBER
2124 C
2125     IF (DIGITS.EQ.1) THEN
2126         XSTART=XX
2127     ELSE
2128         IF (MOD(DIGITS,2).EQ.0) THEN
2129             XSTART=XX-(DIGITS*XINC)/4
2130         ELSE
2131             XSTART=XX-(DIGITS*XINC)/2
2132         ENDIF
2133     ENDIF
2134 C
2135 C PLOT THE NUMBER
2136 C
2137     CALL FNUMBR(XSTART,YY-NY,XINC,0.0,NX,0.0,
2138 * NY,NUMBR,DIGITS,0) ;PRINT VERTEX NUMBER
2139 20 CONTINUE
2140 C
2141 C RESTORE THE COORDINATE SYSTEM
2142 C
2143     DO 2 I = 1, 4
2144 2 XYCOORD(I) = SSP(I)
2145     RETURN
2146     END
```

```

2147      SUBROUTINE PLYPL4(ARRAY,NARRAY,VERTX,SP,SPEC)
2148 C
2149 C THIS SUBROUTINE PLOTS EDGES WHICH ARE WITHIN THE WINDOW OR
2150 C EDGE SEGMENTS WHICH CROSS A PORTION OF THE WINDOW.
2151 C
2152      LOGICAL VALID
2153      INTEGER ARRAY(2,NARRAY), SPEC(4,NARRAY)
2154      REAL VERTX(3,NARRAY),SP(4),SSP(4),U1,V1,U2,V2
2155      COMMON/PLSCMR/DOZ,WCX,WCY,FANGLE,TANAL
2156      COMMON/DEVTYP/IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
2157 C
2158 C DEFINE THE WINDOW SPACE
2159 C
2160      DO 1 I = 1, 4
2161 1      SSP(I) = XYCOORD(I)
2162      CALL DEFINE(SP(1),SP(2),SP(3),SP(4))
2163 C
2164 C PLOT ONE EDGE AT A TIME
2165 C
2166      LWD = -1
2167      LCL = -1
2168      DO 150 J=1,NARRAY
2169 C
2170 C ADD PERSPECTIVE
2171 C
2172      D = DOZ/((DOZ+VERTX(3,ARRAY(1,J)))*TANAL)
2173      U1=(VERTX(1,ARRAY(1,J))-WCX)*D + WCX
2174      V1=(VERTX(2,ARRAY(1,J))-WCY)*D + WCY
2175      D = DOZ/((DOZ+VERTX(3,ARRAY(2,J)))*TANAL)
2176      U2=(VERTX(1,ARRAY(2,J))-WCX)*D + WCX
2177      V2=(VERTX(2,ARRAY(2,J))-WCY)*D + WCY
2178 C
2179 C CLIP LEFT EDGE
2180 C
2181      IF (U1.GE.SP(1).AND.U2.GE.SP(1)) GO TO 50
2182      IF (U1.LT.SP(1).AND.U2.LT.SP(1)) GO TO 150
2183      IF (U1.GT.SP(1)) GO TO 40
2184      V1 = (V1-V2)*U1/(U2-U1)+V1
2185      U1 = SP(1)
2186      GO TO 50
2187 40      V2 = (V2-V1)*U2/(U1-U2)+V2
2188      U2 = SP(1)
2189 C
2190 C CLIP RIGHT EDGE
2191 C
2192 50      IF (U1.LE.SP(3).AND.U2.LE.SP(3)) GO TO 70
2193      IF (U1.GT.SP(3).AND.U2.GT.SP(3)) GO TO 150
2194      IF (U1.GT.SP(3)) GO TO 60
2195      V2 = (V2-V1)*(SP(3)-U1)/(U2-U1)+V1
2196      U2 = SP(3)
2197      GO TO 70
2198 60      V1 = (V1-V2)*(SP(3)-U2)/(U1-U2)+V2
2199      U1 = SP(3)
2200 C
2201 C CLIP BOTTOM EDGE

```

```

2202 C
2203 70 IF (V1.GE.SP(2).AND.V2.GE.SP(2)) GO TO 90
2204 IF (V1.LT.SP(2).AND.V2.LT.SP(2)) GO TO 150
2205 IF (V1.GT.SP(2)) GO TO 80
2206 U1 = (U1-U2)*V1/(V2-V1)+U1
2207 V1 = SP(2)
2208 GO TO 90
2209 80 U2 = (U2-U1)*V2/(V1-V2)+U2
2210 V2 = SP(2)
2211 C
2212 C CLIP TOP EDGE
2213 C
2214 90 IF (V1.LE.SP(4).AND.V2.LE.SP(4)) GO TO 110
2215 IF (V1.GT.SP(4).AND.V2.GT.SP(4)) GO TO 150
2216 IF (V1.GT.SP(4)) GO TO 100
2217 U2 = (U2-U1)*(SP(4)-V1)/(V2-V1)+U1
2218 V2 = SP(4)
2219 GO TO 110
2220 100 U1 = (U1-U2)*(SP(4)-V2)/(V1-V2)+U2
2221 V1 = SP(4)
2222 C
2223 C PLOT THE EDGE
2224 C
2225 110 IF(LWD.NE.SPEC(1,J)) THEN
2226     LWD = SPEC(1,J)
2227     CALL LINWID(LWD)
2228     ENDIF
2229     IF(LCL.NE.SPEC(2,J)) THEN
2230         LCL = SPEC(2,J)
2231         CALL COLOR(LCL)
2232     ENDIF
2233     CALL LINE(U1, V1, U2, V2)
2234 150 CONTINUE
2235 C
2236 C RESTORE THE ORIGINAL COORDINATE SYSTEM
2237 C
2238 DO 2 I = 1, 4
2239 2 XYCOORD(I) = SSP(I)
2240 RETURN
2241 END

```

```

2242     SUBROUTINE PLYPL5(ARRAY,NARRAY,VERTX,SP,SPEC,NPVERT)
2243 C
2244 C THIS SUBROUTINE GRAPHS POLYGONS WHICH ARE WITHIN THE WINDOW OR
2245 C PARTS OF POLYGONS WHICH CROSS A PORTION OF THE WINDOW
2246 C
2247     PARAMETER(MAXLN=10)
2248     LOGICAL PASS1
2249     INTEGER ARRAY(NPVERT,NARRAY),NARRAY,FIRST,SECOND,NUM,S1
2250     INTEGER INVALID,VINDX,ENDPT,SPEC(4,NARRAY)
2251     REAL VERTX(3,NARRAY),SP(4),D,SSP(4)
2252     REAL X1,X2,Y1,Y2,X(MAXLN),Y(MAXLN)
2253     COMMON/PLSCMR/DOZ,WCX,WCY,FANGLE,TANAL
2254     COMMON/DEVTP/IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
2255 C
2256 C DEFINE WINDOW SPACE
2257 C
2258     DO 1 I = 1, 4
2259 1     SSP(I) = XYCOORD(I)
2260     CALL DEFINE(SP(1),SP(2),SP(3),SP(4))
2261 C
2262 C PLOT POLYGONS ONE AT A TIME
2263 C
2264     LWD = -1
2265     NDX = -1
2266     LFL = -1
2267     DO 10 I = 1, NARRAY
2268     NUM=0
2269 C
2270 C DETERMINE THE NUMBER OF VERTICES (NON-ZERO ENTRIES) IN THE
2271 C POLYGON
2272 C
2273     DO 15 J=1,NPVERT
2274     IF(ARRAY(J,I).NE.0) NUM=NUM+1 ;NUMBER OF VERTICES IN POLYGON
2275 15 CONTINUE
2276 C
2277 C FIND THE PERSPECTIVE
2278 C
2279     DO 60 IV=1,NUM
2280     D = DOZ/((DOZ+VERTX(3,ARRAY(IV,I)))*TANAL)
2281     X(IV) = (VERTX(1,ARRAY(IV,I)) - WCX)*D + WCX
2282     Y(IV) = (VERTX(2,ARRAY(IV,I)) - WCY)*D + WCY
2283 60 CONTINUE
2284 C
2285 C PLOT THE POLYGON
2286 C
2287     IF(SPEC(1,I).NE.LWD) THEN
2288     LWD = SPEC(1,I)
2289     CALL LINWID(LWD)
2290     ENDIF
2291     IF(SPEC(2,I).NE.NDX) THEN
2292     NDX = SPEC(2,I)
2293     CALL COLOR(NDX)
2294     ENDIF
2295     IF(SPEC(3,I).NE.LFL) THEN
2296     LFL = SPEC(3,I)

```

```
2302     CALL LINES(X(1), Y(1), X(2), Y(2), NUM-1)
2303     ENDIF
2304 10   CONTINUE           ;GET NEXT POLYGON
2305 C
2306 C   RESTORE THE COORDINATE SYSTEM
2307 C
2308     DO 2 I = 1, 4
2309 2   XYCOORD(I) = SSP(I)
2310     RETURN
2311     END
```

```

2312     SUBROUTINE PLYPLT(FILE,OPTION)
2313     $INCLUDE BUILD.COM
2314     PARAMETER      (NPART=20)
2315     LOGICAL        EXIST, VALID
2316     INTEGER        NVERTO, NELMTO, NEDGE0, NPOLYO, FIRST, LAST, IOS
2317     INTEGER        EDGS(NSPEC,NEDGE), POLS(NSPEC,NPOLY), SELECT
2318     INTEGER        FNM(5), OPTION, EDG(2,NEDGE), POL(NPVERT,NPOLY)
2319     INTEGER        NPL(2,NPART), ISPEC(NSPEC), JP(NPVERT)
2320     REAL           WIN(4), VER(3,NVERT), SPACE(4)
2321     CHARACTER*5    SOURCE
2322     CHARACTER*17   FNAME, FILENM, OLDFIL, FILE
2323     CHARACTER*20   NOREC, BLANK
2324     DATA          VALID/.FALSE./, BLANK/' '/, LUNIT/9/, OLDFIL/' '/
2325     DATA          IUNIT/5/, ZERO/0/
2326     EQUIVALENCE   (FILENM,FNM)
2327     COMMON/PLSCMR/DOZ,WCX,WCY,FANGLE,TANAL
2328     COMMON/GRFTYP/ANGLE,IRVRSE,CHSIZE(9),ITKWIT,LUDIAG,LUHSET
2329     C*****
2330     C     DATA FILE
2331     C*****
2332     C
2333     C     CREATE EQUIVALENT CHARACTER STRING OF FILENAME
2334     C
2335         FILENM = FILE
2336         IF(FILENM.EQ.OLDFIL) GO TO 1100
2337     C
2338     C     READ FILENAME
2339     C
2340         CLOSE (LUNIT)
2341         CALL PLYPL1(FNM,17,1,FIRST,LAST,EXIST,0)
2342         IF(.NOT.EXIST) GO TO 90
2343         FNAME = BLANK
2344         DO 10 I = FIRST, LAST
2345             J = I - FIRST + 1
2346             10 FNAME(J:J) = FILENM(I:I)
2347     C
2348     C     FILE EXIST?
2349     C
2350         INQUIRE(FILE=FNAME,IOSTAT=IOS,ERR=91,EXIST=EXIST)
2351         IF(.NOT.EXIST) GO TO 93
2352     C
2353     C     OPEN FILE
2354     C
2355         OPEN(UNIT=LUNIT,IOSTAT=IOS,ERR=94,FILE=FNAME,STATUS='OLD')
2356         REWIND LUNIT
2357     C
2358     C     GET(1)     READ DATA INTO VARIABLES AND ARRAYS
2359     C
2360         READ(LUNIT,FMT=1500,ERR=98) SOURCE,(WINDOW(I),I=1,4)
2361         IF(SOURCE.NE.'BUILD'.AND.SOURCE.NE.'MOVIE'.AND.SOURCE.NE.'FILNG')
2362             GO TO 98
2363         READ(LUNIT,FMT=1510,ERR=98) (WORLD(I),I=1,6)
2364         READ(LUNIT,FMT=1520,ERR=98) NPARTO,NVERTO,NELMTO,NEDGE0
2365         IF (NPARTO.LT.0) GO TO 98
2366         NPOLYO=NELMTO-NEDGE0

```

```

2367      READ(LUNIT,FMT=1520,ERR=98) ((NPL(I,J),I=1,2),J=1,NPARTO)
2368      IF(NVERTO.LE.0) GO TO 1090
2369      DO 1020 J=1,NVERTO
2370 1020  READ(LUNIT,FMT=1530,ERR=98) NPOINT(J),(VERTEX(I,J),I=1,3)
2371      IF (NELMTO.LE.0) GO TO 1090
2372  C
2373  C      INITIALIZE EDGE AND POLYGON COUNTERS
2374  C
2375      NE=0
2376      NP=0
2377      DO 1080 J=1,NELMTO
2378      READ(LUNIT,FMT=1525,ERR=98) (JP(I),I=1,NPVERT),(ISPEC(I),I=1,
2379 1  NSPEC)
2380  C
2381  C      COUNT NUMBER OF VERTICES IN ELEMENT
2382  C
2383      NCON=0
2384      DO 1030 I=1,NPVERT
2385      IF (JP(I).EQ.0) GO TO 1035
2386 1030  NCON=NCON+1
2387  C
2388  C      ELEMENT IS EDGE
2389  C
2390 1035  IF (NCON.EQ.2) THEN
2391          NE=NE+1
2392          DO 1040 I=1,2
2393 1040  EDGE(I,NE)=JP(I)
2394          DO 1050 I=1,NSPEC
2395 1050  ESPEC(I,NE)=ISPEC(I)
2396  C
2397  C      ELEMENT IS POLYGON
2398  C
2399      ELSE
2400          NP=NP+1
2401          DO 1060 I=1,NCON
2402 1060  POLY(I,NP)=JP(I)
2403          IF (NCON.LT.NPVERT) THEN
2404              DO 1065 I=NCON+1,NPVERT
2405 1065  POLY(I,NP)=0
2406          ENDIF
2407          DO 1070 I=1,NSPEC
2408 1070  PSPEC(I,NP)=ISPEC(I)
2409      ENDIF
2410 1080  CONTINUE
2411 1090  CLOSE(LUNIT)
2412  C*****
2413  C  DISPLAY
2414  C*****
2415  C
2416  C **SPECIFY WINDOW**
2417  C
2418 1100  IF (NVERTO.NE.0) THEN
2419          CALL PLYPL2(WINDOW,VALID,WORLD)
2420      ELSE
2421          NOREC = 'NO RECORDS'

```



```

2422         IF(LUDIAG.GT.0) WRITE(LUDIAG,108) NOREC
2423         RETURN
2424     ENDIF
2425 C
2426 C IF VALID WINDOW , SET 'SPACE' COORDINATES TO 'WINDOW' ,OTHERWISE
2427 C SET THE 'SPACE' COORDINATES TO 'WORLD'
2428 C
2429         IF(VALID) THEN                ;VALID WINDOW
2430             DO 40 I=1,4
2431 40         SPACE(I)=WINDOW(I)
2432         ELSE
2433             IF(LUDIAG.GT.0) WRITE(LUDIAG,107)
2434             DO 41 I=1,2
2435 41         SPACE(I)=WORLD(I)
2436             DO 42 I=3,4
2437 42         SPACE(I)=WORLD(I+1)
2438         ENDIF
2439 C
2440 C **FIND CENTER OF WINDOW**
2441 C
2442         WCX=(SPACE(3)+SPACE(1))/2
2443         WCY=(SPACE(4)+SPACE(2))/2
2444 C
2445 C **INITIALIZE FIELD OF VIEW**
2446 C
2447         DOZ=10000.
2448         FANGLE=90.
2449         TANAL=1.0
2450 C
2451 C INITIALIZE AND SET UP FOR A SINGLE GRAPH - 'NOREC', IF SET, WILL
2452 C INDICATE THAT THERE ARE NO RECORDS TO BE DISPLAYED
2453 C
2454         NOREC=BLANK
2455 C
2456         IF (OPTION.EQ.0) THEN          ;NO PLOTS - READ IN DATA FILE ONLY
2457             RETURN
2458         ELSE IF (OPTION.EQ.1) THEN
2459             IF (NEDGEQ.EQ.0) NOREC='NO EDGES'
2460         ELSE IF(OPTION.EQ.2) THEN
2461             IF (NVERTQ.EQ.0) NOREC='NO VERTICES'
2462         ELSE IF (OPTION.EQ.3) THEN
2463             IF (NPOLYQ.EQ.0) NOREC='NO POLYGONS'
2464         ELSE IF (OPTION.EQ.4) THEN
2465             IF (NEDGEQ.EQ.0.AND.NPOLYQ.EQ.0) NOREC='NO EDGES OR POLYGONS'
2466         ELSE
2467             NOREC='INVALID OPTION'
2468             IF(LUDIAG.GT.0) WRITE(LUDIAG,108) NOREC
2469             RETURN
2470         ENDIF
2471 C
2472 C **PLOT**
2473 C
2474         IF (NOREC.EQ.BLANK) THEN
2475             IF (OPTION.EQ.1) THEN
2476                 CALL PLYPL4(EDGE,NEDGEQ,VERTEX,SPACE,ESPEC)

```

```

2477     ELSE IF (OPTION.EQ.2) THEN
2478         CALL PLYPL3(VERTEX,NVERTO,SPACE,NPOINT,NVERT)
2479     ELSE IF (OPTION.EQ.3) THEN
2480         CALL PLYPL5(POLY,NPOLYO,VERTEX,SPACE,PSPEC,NPVERT)
2481     ELSE IF (OPTION.EQ.4) THEN
2482         IF(NEDGE0.GT.0) CALL PLYPL4(EDGE,NEDGE0,VERTEX,SPACE,ESPEC)
2483         IF(NPOLYO.GT.0) CALL PLYPL5(POLY,NPOLYO,VERTEX,SPACE,PSPEC,
2484             NPVERT)
2485     ENDIF
2486     ELSE
2487         IF(LUDIAG.GT.0) WRITE(LUDIAG,108) NOREC
2488     ENDIF
2489     RETURN
2490 C*****
2491 C   ENTRY PLYGNS
2492 C*****
2493 C
2494 C   A ROUTINE TO RETURN THE COORDINATE ARRAY VALUES FROM A DATA FILE
2495 C
2496     ENTRY PLYGNS(WIN,VER,NVM,EDG,EDGS,NEM,POL,POLS,NPM)
2497     NVM = MIN(NVERT,NVM,NVERTO)
2498     DO 80 I = 1, 4
2499 80   WIN(I) = WINDOW(I)
2500     DO 81 I = 1, NVM
2501     DO 82 J = 1, 3
2502 82   VER(J,I) = VERTEX(J,I)
2503 81   CONTINUE
2504     NEM = MIN(NEDGE,NEM,NE)
2505     DO 88 I = 1, NEM
2506     DO 83 J = 1, 2
2507 83   EDG(J,I) = EDGE(J,I)
2508     DO 84 J = 1, NSPEC
2509 84   EDGS(J,I) = ESPEC(J,I)
2510 88   CONTINUE
2511     NPM = MIN(NPOLY,NPM,NP)
2512     DO 85 I = 1, NPM
2513     DO 86 J = 1, NPVERT
2514 86   POL(J,I) = POLY(J,I)
2515     DO 87 J = 1, NSPEC
2516 87   POLS(J,I) = PSPEC(J,I)
2517 85   CONTINUE
2518     RETURN
2519 C
2520 C   ERRORS
2521 C
2522 90   IF(LUDIAG.GT.0) WRITE(LUDIAG,100) FILE
2523     RETURN
2524 91   CLOSE(LUNIT)
2525     IF (IOS.EQ.349) THEN
2526         IF(LUDIAG.GT.0) WRITE(LUDIAG,100) FNAME
2527     ELSE IF (IOS.EQ.324) THEN
2528         IF(LUDIAG.GT.0) WRITE(LUDIAG,101)
2529     ELSE
2530         IF(LUDIAG.GT.0) WRITE(LUDIAG,102) IOS
2531     ENDIF

```

```
2532      RETURN
2533 93     IF(LUDIAG.GT.0) WRITE(LUDIAG,103)
2534      RETURN
2535 94     CLOSE(UNIT=LUNIT,IOSTAT=IOS,ERR=95)
2536 95     IF(LUDIAG.GT.0)WRITE(LUDAIG,104) IOS
2537      RETURN
2538 98     IF(LUDIAG.GT.0) WRITE(LUDIAG,110)
2539      RETURN
2540 C
2541 C  FORMATS
2542 C
2543 100    FORMAT(' INVALID FILE DESCRIPTOR',2X,A17)
2544 101    FORMAT(' NO RECORDS IN FILE')
2545 102    FORMAT(' INQUIRE ERROR = ',I4)
2546 103    FORMAT(' FILE DOES NOT EXIST')
2547 104    FORMAT(' FILE ERROR',I4)
2548 107    FORMAT(' WINDOW DEFAULTS TO "WORLD"')
2549 108    FORMAT(1X,A20,/,/)
2550 110    FORMAT(' ERROR - DATA  FORMAT PROBLEMS -- ENTER NEW COMMAND')
2551 1500   FORMAT(A5,4E12.5)
2552 1510   FORMAT(6E12.5)
2553 1520   FORMAT(16I5)
2554 1525   FORMAT(8I5,5X,8I5)
2555 1530   FORMAT(I5,3E12.5)
2556      END
```

```

2557 SUBROUTINE PROPOL (NTRIA, XT1, YT1, FT1, XT2, YT2, FT2, XT3, YT3, FT3,
2558 1 FL, NX, NY)
2559 C
2560 PARAMETER (NPT=101)
2561 REAL XT1(NTRIA), XT2(NTRIA), XT3(NTRIA)
2562 REAL YT1(NTRIA), YT2(NTRIA), YT3(NTRIA)
2563 REAL FT1(NTRIA), FT2(NTRIA), FT3(NTRIA)
2564 INTEGER MASK(NPT)
2565 REAL DIFF1(NPT), DIFF2(NPT), DIFF3(NPT)
2566 REAL SGN1(NPT), SGN2(NPT), SGN3(NPT)
2567 REAL DFL1(NPT), DFL2(NPT), DFL3(NPT)
2568 REAL ALF1(NPT), ALF2(NPT), ALF3(NPT), X1(NPT), X2(NPT)
2569 REAL X3(NPT), Y1(NPT), Y2(NPT), Y3(NPT), TEST(NPT)
2570 REAL DELTA, MASK1, MASKS, RMASK(NPT)
2571 EQUIVALENCE (MASK(1), RMASK(1))
2572 DATA MASK1, DELTA, MASKS / 1.0, 1.0E-20, Z80000000/
2573 DATA XMN1, YMN1, XMN2, YMN2, XMX1, YMX1, XMX2, YMX2/4*0., 4*100./
2574 C
2575 C * * * * *
2576 C
2577 C PROPOL (NTRIA, XT1, YT1, FT1, XT2, YT2, FT2, XT3, YT3, FT3,
2578 C FL, NX, NY)
2579 C
2580 C NTRIA INTEGER NUMBER OF TRIANGLES IN LIST I
2581 C XT1 REAL ARRAY(NTRIA) X POSITIONS OF FIRST VERTICES I
2582 C YT1 REAL ARRAY(NTRIA) Y POSITIONS OF FIRST VERTICES I
2583 C FT1 REAL ARRAY(NTRIA) FUNCTION VALUES AT FIRST VERTICES I
2584 C XT2 REAL ARRAY(NTRIA) X POSITIONS OF SECOND VERTICES I
2585 C YT2 REAL ARRAY(NTRIA) Y POSITIONS OF SECOND VERTICES I
2586 C FT2 REAL ARRAY(NTRIA) FUNCTION VALUES AT SECOND VERTICES I
2587 C XT3 REAL ARRAY(NTRIA) X POSITIONS OF THIRD VERTICES I
2588 C YT3 REAL ARRAY(NTRIA) Y POSITIONS OF THIRD VERTICES I
2589 C FT3 REAL ARRAY(NTRIA) FUNCTION VALUES AT THIRD VERTICES I
2590 C FL REAL VALUE OF F AT WHICH CONTOUR IS DRAWN I
2591 C NX INTEGER MAXIMUM VALUE OF X POSITIONS I
2592 C NY INTEGER MAXIMUM VALUE OF Y POSITIONS I
2593 C
2594 C PROPOL ASSUMES A CONTOUR CROSSING LIES
2595 C BETWEEN (XT1, YT1) AND (XT2, YT2). THE VERTICES MUST BE NUMBERED
2596 C COUNTER-CLOCKWISE AROUND THE TRIANGLE. THE CODE IN PROPOL IS SPEC-
2597 C IALIZED TO THE RECTANGULAR X-Y CASE BY THE TRANSFORMATION OF DO
2598 C LOOP 145. PLOAR PLOTS CAN BE FORMED BY CHANGING THIS TRANSFORMA-
2599 C TION.
2600 C
2601 C ENTRY POINTS: CNTSET, CNTFRM (SEE DOCUMENTATION OR LISTING BELOW)
2602 C
2603 C * * * * *
2604 C
2605 C IF (NTRIA .GT. NPT) NTRIA = NPT
2606 C XD = (XMN2-XMN1) / FLOAT(NX-1)
2607 C XB = ((XMN2-XMN1)-(XMN3-XMN4)) / FLOAT((NY-1)*(NX-1))
2608 C XC = (XMN4 - XMN1) / FLOAT(NY-1)
2609 C YD = (YMN4 - YMN1) / FLOAT(NX-1)
2610 C YB = ((YMN4-YMN1)-(YMN3-YMN2)) / FLOAT((NY-1)*(NX-1))
2611 C YC = (YMN2 - YMN1) / FLOAT(NX-1)

```

```

2612 C
2613 DO 100 I = 1, NTRIA
2614 DIFF3(I) = FT2(I) - FT1(I)
2615 DIFF1(I) = FT3(I) - FT1(I)
2616 DIFF2(I) = FT3(I) - FT2(I)
2617 SGN3(I) = SIGN (1.0, DIFF3(I))
2618 SGN1(I) = SIGN (1.0, DIFF1(I))
2619 100 SGN2(I) = SIGN (1.0, DIFF2(I))
2620 DO 110 I = 1, NTRIA
2621 DIFF3(I) = ABS(DIFF3(I)) + DELTA
2622 DIFF1(I) = ABS(DIFF1(I)) + DELTA
2623 DIFF2(I) = ABS(DIFF2(I)) + DELTA
2624 DFL3(I) = SGN3(I)*(FL - FT1(I))
2625 DFL1(I) = SGN1(I)*(FL - FT1(I))
2626 110 DFL2(I) = SGN2(I)*(FL - FT2(I))
2627 DO 120 I = 1, NTRIA
2628 ALF3(I) = DFL3(I)/DIFF3(I)
2629 ALF1(I) = DFL1(I)/DIFF1(I)
2630 ALF2(I) = DFL2(I)/DIFF2(I)
2631 X3(I) = XT1(I) + ALF3(I)*(XT2(I) - XT1(I))
2632 Y3(I) = YT1(I) + ALF3(I)*(YT2(I) - YT1(I))
2633 X1(I) = XT1(I) + ALF1(I)*(XT3(I) - XT1(I))
2634 Y1(I) = YT1(I) + ALF1(I)*(YT3(I) - YT1(I))
2635 X2(I) = XT2(I) + ALF2(I)*(XT3(I) - XT2(I))
2636 Y2(I) = YT2(I) + ALF2(I)*(YT3(I) - YT2(I))
2637 120 TEST(I) = (1.0 - ALF2(I))*ALF2(I)
2638 C
2639 C IF TEST IS LESS THAN ZERO, THEN MISS, OTHERWISE HIT.
2640 C
2641 DO 130 I = 1, NTRIA
2642 MASK(I) = LSHF (TEST(I), -31)
2643 MASK(I) = MASK(I) - 1
2644 X2(I) = AND (RMASK(I), X2(I))
2645 130 Y2(I) = AND (RMASK(I), Y2(I))
2646 DO 140 I = 1, NTRIA
2647 MASK(I) = -1 - MASK(I)
2648 X1(I) = AND (RMASK(I), X1(I))
2649 Y1(I) = AND (RMASK(I), Y1(I))
2650 X1(I) = X1(I) + X2(I)
2651 140 Y1(I) = Y1(I) + Y2(I)
2652 C
2653 C NOW PLOT THE LINE SEGMENTS.
2654 C
2655 DO 145 I = 1, NTRIA
2656 X1(I) = (X1(I)-1.)*(XD-XB*(Y1(I)-1.))+XC*(Y1(I)-1.)+XMIN1
2657 X3(I) = (X3(I)-1.)*(XD-XB*(Y3(I)-1.))+XC*(Y3(I)-1.)+XMIN1
2658 Y1(I) = (Y1(I)-1.)*(YD-YB*(X1(I)-1.))+YC*(X1(I)-1.)+YMIN1
2659 145 Y3(I) = (Y3(I)-1.)*(YD-YB*(X3(I)-1.))+YC*(X3(I)-1.)+YMIN1
2660 DO 150 I = 1, NTRIA
2661 X3(I) = AMIN1(XMAX, AMAX1(XMIN, X3(I)))
2662 X1(I) = AMIN1(XMAX, AMAX1(XMIN, X1(I)))
2663 Y3(I) = AMIN1(YMAX, AMAX1(YMIN, Y3(I)))
2664 150 Y1(I) = AMIN1(YMAX, AMAX1(YMIN, Y1(I)))
2665 CALL LINES (X3, Y3, X1, Y1, NTRIA)
2666 NTRIA = 0

```

```

2667         RETURN
2668 C
2669 C
2670 C -----
2671 C
2672 ENTRY CNTSET (XM1, YM1, XM2, YM2, XM3, YM3, XM4, YM4)
2673 C
2674 C * * * * *
2675 C
2676 C DESCRIPTION: CNTSET MAY BE CALLED BY THE USER TO MOVE THE CONTOUR
2677 C PLOT GENERATED AROUND ON THE PLOTTING REGION OR TO STRETCH OR
2678 C COMPRESS THE PLOT. THE DATA STATEMENT GIVES THE DEFAULT FOR A
2679 C LARGE SQUARE PLOT. THE PLOT WILL EXTEND FROM XXMIN TO XXMAX IN THE
2680 C HORIZONTAL AND FROM YYMIN TO YYMAX IN THE VERTICAL. ALL FOUR OF
2681 C THESE VALUES SHOULD BE IN THE RANGE 1 TO 1023.
2682 C
2683 C * * * * *
2684 C
2685         XMN1 = XM1
2686         YMN1 = YM1
2687         XMN2 = XM2
2688         YMN2 = YM2
2689         XMN3 = XM3
2690         YMN3 = YM3
2691         XMN4 = XM4
2692         YMN4 = YM4
2693         XMAX = MAX(XMN1, XMN2, XMN3, XMN4)
2694         XMIN = MIN(XMN1, XMN2, XMN3, XMN4)
2695         YMAX = MAX(YMN1, YMN2, YMN3, YMN4)
2696         YMIN = MIN(YMN1, YMN2, YMN3, YMN4)
2697         RETURN
2698 C
2699 C
2700 C -----
2701 C
2702 ENTRY CNTFRM
2703 C
2704 C * * * * *
2705 C
2706 C DESCRIPTION: CNTFRM IS A USER-CALLED ROUTINE TO PLOT THE RECTAN-
2707 C GULAR BOUNDARY OF THE CONTOURED REGION. CNTFRM HAS NO ARGUMENTS
2708 C SINCE THE REGION IS SPECIFIED BY DEFAULT OR VIA A PREVIOUS CALL TO
2709 C CNTSET.
2710 C
2711 C * * * * *
2712 C
2713 C
2714         CALL LINE(XMN1, YMN1, XMN2, YMN2)
2715         CALL LINE(XMN2, YMN2, XMN3, YMN3)
2716         CALL LINE(XMN3, YMN3, XMN4, YMN4)
2717         CALL LINE(XMN4, YMN4, XMN1, YMN1)
2718         RETURN
2719 END

```

```

2720      SUBROUTINE PUTCH (X, Y, CHARAC)
2721 C
2722      INTEGER LPAGE(30, 65)
2723      COMMON/DEVTYP/IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
2724      COMMON/GRFTYP/ANGLE,IRVRSE,CHSIZE(9),ITKWIT,LUDIAG,LUHSET
2725      CHARACTER*1 CHARAC, CH(4)
2726      EQUIVALENCE (CH,ICH)
2727      INTEGER K, K1, M, IK, IM, MASK(5)
2728      INTEGER ISHFT, AND, OR
2729      LOGICAL LSW, LTSW
2730      DATA MASK /ZFFFF00FF, ZFF00FFFF, Z00FFFFFF, ZFFFFFF00,
2731      . ZFF000000/, CHSZ/0.0/
2732 C
2733      ICH = 0
2734      CH(1) = CHARAC
2735      CHS = CHSIZE(2)
2736      GO TO (1,2,3,5), IDEVIC
2737 C
2738 C      CALCOMP SECTION.
2739 C
2740      1      CALL CALSYM (X, Y, CHS, CH, 0, 0.0, 1)
2741      LSW = .TRUE.
2742      RETURN
2743 C
2744 C      PAGE PLOT SECTION.
2745 C
2746      2      JX1 = MAX(0, MIN(IFIX(X),1023))
2747      JY1 = MAX(0, MIN(IFIX(Y),1023))
2748      I = (JX1 + 5)/10 + 1
2749      J = 65 - (JY1+8)/16
2750      K = I/4
2751      M = I - 4*K
2752      K1 = 4 - M
2753      IF(M .EQ. 0) K = K - 1
2754      IM = IAND(LPAGE(K+4, J), MASK(K1))
2755      IK = IAND(ICH, MASK(5))
2756      IF(M .EQ. 0) IK = ISHFT(IK, -24)
2757      IF(M .EQ. 3) IK = ISHFT(IK, -16)
2758      IF(M .EQ. 2) IK = ISHFT(IK, -8)
2759      LPAGE(K+4, J) = IOR(IK, IM)
2760      RETURN
2761 C
2762 C      LEXIDATA/MATRIX SECTION
2763 C
2764      3      CALL GSCHSZ (CHS)
2765      XL = MAX(X,0.)
2766      YL = MAX(Y,0.)
2767      CALL GMOVEA (XL, YL)
2768      CALL GTXTWC (CH, 1)
2769      RETURN
2770 C
2771 C      TEKTRONIX SECTION
2772 C
2773      5      CONTINUE
2774      CALL MOVEA(X,Y)

```

```
2775     CALL A1OUT(1, ICH)
2776     LTSW=.TRUE.
2777     RETURN
2778 END
```



```

2779      SUBROUTINE PUTLNV (X1, Y1, DX, DY,  NL)
2780  C
2781      PARAMETER (NLines=200)
2782      INTEGER LENGTH, COUNT
2783      REAL X1(NL), Y1(NL), DX(NL), DY(NL)
2784      INTEGER MASK1/15/,OP11/12/,IVERT/1023/
2785      INTEGER LPAGE(30,65), HSYMBL, HDOT, HAPOS, HMINUS
2786      REAL JX1(NLINES),JY1(NLINES),JX2(NLINES),JY2(NLINES)
2787      REAL XY(2,NLINES), JXS, JYS, IDDX, IDDY
2788      COMMON/DEV TYP/IDEVIC,LSW,LTSW,XYCOOD(4),LUOUT,LPAGE
2789      COMMON/GRFTYP/ANGLE,IRVRSE,CHSIZE(9),ITKWIT,LU DIAG,LUHSET
2790      COMMON/GRF MOD/LFLMAT(5),LINWDM(5),LCLMAT(5)
2791      INTEGER K, K1, M, M1, IK, IM, MASK(4)
2792      LOGICAL LSW, LTSW, IRVRSE, LEXLSW
2793      DATA HDOT, HAPOS, HMINUS /Z000C002E, Z00000027, Z0000002D /
2794      DATA MASK /ZFFFF00FF, ZFF00FFFF, Z00FFFFFF, ZFFFFFFF00 /
2795      DATA JXS, JYS / 2*0.0 /
2796  C
2797      IF(IDEVIC.EQ.0) RETURN
2798      IF(NL.GT.NLINES.AND.LUDIAG.GT.0) WRITE(LUDIAG,1002)
2799 1002  FORMAT(' *** BUFFER SPACE EXCEEDED IN PUTLNV ***')
2800      NLP = MINO(NLINES,NL)
2801  C
2802  C      MASK THE LINE SEGMENT FOR CALCOMP AND PAPER, AND TEKTRONIK.
2803  C
2804      DO 10 I=1,NLP
2805      JX1(I) = X1(I)
2806      JY1(I) = Y1(I)
2807      JX2(I) = JX1(I) + DX(I)
2808 10     JY2(I) = JY1(I) + DY(I)
2809      DO 11 I = 1, NLP
2810      JX1(I) = MAX(JX1(I),0.)
2811      JY1(I) = MAX(JY1(I),0.)
2812      JX2(I) = MAX(JX2(I),0.)
2813 11     JY2(I) = MAX(JY2(I),0.)
2814      GO TO (1,2,3,5), IDEVIC
2815  C
2816  C      CALCOMP SECTION.
2817  C
2818 1     DO 16 I = 1, NLP
2819      IF(LSW) GO TO 12
2820      IF(JXS.EQ.JX1(I) .AND. JYS.EQ.JY1(I))GO TO 15
2821 12     CALL CALPLT(JX1(I), JY1(I), 3)
2822      LSW =.FALSE.
2823 15     CALL CALPLT(JX2(I), JY2(I), 2)
2824      JXS=JX2(I)
2825      JYS=JY2(I)
2826 16     CONTINUE
2827      RETURN
2828  C
2829  C      PAGE PLOT SECTION.
2830  C
2831 2     DO 22 I=1,NLP
2832      HSYMBL = HDOT
2833      IF(DX(I).GT.2*DY(I))HSYMBL=HMINUS

```

```

2834      IF(DY(I).GT.2*DX(I))HSYMBL=HAPOS
2835      IDDX=JX2(I)-JX1(I)
2836      IDDY=JY2(I)-JY1(I)
2837      DO 20 L = 1, 9
2838      JX=JX1(I)+((L-1)*IDDX)/8
2839      JY=JY1(I)+((L-1)*IDDY)/8
2840      IX= (JX+5)/10 + 1
2841      IY= 65 - (JY+8)/16
2842      K=IX/4
2843      M = IX- 4*K
2844      IK = HSYMBL
2845      K1 = 4 - M
2846      M1 = 8*K1
2847      IF(M .EQ. 0) K = K - 1
2848      IM = IAND(LPAGE(K+4,IY), MASK(K1))
2849      IF( M .NE. 0) IK = ISHFT(HSYMBL, M1)
2850      20  LPAGE(K+4,IY) = IOR(IM, IK)
2851      22  CONTINUE
2852      RETURN
2853  C
2854  C      LEXIDATA/MATRIX SECTION
2855  C
2856      3      IL = 1
2857      32     IS = 1
2858      CALL GMOVEA(JX1(IL),JY1(IL))
2859      XY(1,IS) = JX2(IL)
2860      XY(2,IS) = JY2(IL)
2861      31     IL = IL + 1
2862      IF(JX1(IL).NE.XY(1,IS).OR.JY1(IL).NE.XY(2,IS).OR.IL.GT.NLP)
2863      GO TO 30
2864      IS = IS + 1
2865      XY(1,IS) = JX2(IL)
2866      XY(2,IS) = JY2(IL)
2867      GO TO 31
2868      30     CALL GPLNA(XY,IS)
2869      IF(IL.GT.NLP) RETURN
2870      GO TO 32
2871      RETURN
2872  C
2873  C      TEKTRONIX SECTION
2874  C
2875      5      CONTINUE
2876      DO 40 I=1,NLP
2877      IF(LTSW) GO TO 42
2878      IF(JXS.EQ.JX1(I).AND.JYS.EQ.JY1(I)) GO TO 45
2879      42     CALL MOVEA(JX1(I),JY1(I))
2880      LTSW=.FALSE.
2881      45     CALL DRAWA(JX2(I),JY2(I))
2882      JXS=JX2(I)
2883      JYS=JY2(I)
2884      40     CONTINUE
2885      RETURN
2886      END

```

```
2887     SUBROUTINE SETLUT
2888 C
2889     DIMENSION LUT(48)
2890     DATA LUT/0,15,15,0,0,15,15,0,15,1,3,5,7,9,11,13,
2891     . 0,15,0,15,0,15,0,15,8,1,3,5,7,9,11,13,
2892     . 0,15,0,0,15,0,15,15,0,1,3,5,7,9,11,13/
2893 C
2894     CALL DSLWT (16, 48, LUT)
2895     RETURN
2896     END
```

```

2897 SUBROUTINE SRFSET(XX, YY, ZZMIN, ZZMAX, NX, NY)
2898 C
2899 C THIS SUBROUTINE AND ITS ASSOCIATED ENTRIES CONSTRUCT AND
2900 C DELIVER PLOTS OF A SURFACE Z(I,J) WITH HIDDEN LINES REMOVED
2901 C FOR I = 1, ..., NX AND J = 1, ..., NY. THE TYPE, ORIENTATION,
2902 C AND DETAILS OF THE PLOTS ARE QUITE FLEXIBLE AS SEEN BY CAREFUL
2903 C STUDY OF THE TEST PROGRAM AND THE RESULTING OUTPUT
2904 C THE PARTICULAR VERSION IS AN INTERFACE TO THE GENERAL PURPOSE
2905 C PLOTTING PACKAGE WHICH CAN DRAW COLOR PLOTS ON THE CALCOMP,
2906 C LEXIDATA, PRINTER OR TEKTRONICS (4000'S SERIES).
2907 C
2908 C SURFACE INITIALIZES THE SURFACE PLOTTING PACKAGE AND MUST
2909 C BE CALLED EACH TIME A NEW PLOT IS DESIRED TO RESET THE
2910 C HIDDEN LINE ARRAYS.
2911 C NOTE THAT MAX(NX,NY) MUST BE LESS THAN NPT
2912 C
2913 C THE LOGICAL PLOTTING REGION IS A 3D RECTANGULAR PARALLELE-
2914 C PIPED WITH 8 CORNER VERTICES.
2915 C XX REAL - ARRAY DIMENSIONED 8 CONTAINING THE 8 CORNER
2916 C VERTEX X LOCATIONS. THE VERTICES ARE NUMBERED
2917 C AS SHOWN ON THE SURFACE GEOMETRY SHEET (AVAILABL
2918 C FROM THE SPL LIBRARIAN OR THE AUTHOR).
2919 C YY REAL - ARRAY DIMENSIONED 8 CONTAINING THE 8 CORNER
2920 C VERTEX Y LOCATIONS. THE VERTICES ARE NUMBERED
2921 C AS SHOWN ON THE SURFACE GEOMETRY SHEET WHICH IS
2922 C AVAILABLE FROM THE SPL LIBRARIAN OR THE AUTHOR.
2923 C ZZMIN REAL - THE VALUE OF Z(I,J) TO BE PLOTTED AT THE
2924 C BOTTOM SURFACE OF THE PARALLELEPIPED.
2925 C SMALLER Z(I,J) ARE SET (I.E. LIMITED) TO
2926 C ZZMIN.
2927 C ZZMAX REAL - THE VALUE OF Z(I,J) TO BE PLOTTED AT THE
2928 C UPPER SURFACE OF THE PARALLELEPIPED.
2929 C LARGER Z(I,J) ARE SET (I.E. LIMITED) TO
2930 C ZZMAX.
2931 C NX INTEGER - DIMENSION AND RANGE OF I IN Z(I,J)
2932 C NY INTEGER - DIMENSION AND RANGE OF J IN Z(I,J)
2933 C
2934 C SFRAME (MODE1) - PLOTS THE FRAME OF THE RECTANGULAR PARALLELE-
2935 C PIPED PLOTTING REGION.
2936 C MODE1 INTEGER - MODE1 = 1 PLOTS + AT THE VERTICES
2937 C = 2 ALSO PLOTS LINE SEGMENTS
2938 C CONNECTING THE VERTICES.
2939 C
2940 C SSKIRT (ZZ, NX, NY, MODE2) - PLOTS ANY OF THE SKIRTS WHICH MAY
2941 C BE DESIRED. ONLY ONE SKIRT IS PLOTTED PER CALL AND THE
2942 C HIDDEN LINE ALGORITHM IS INVOKED. THEREFORE, THE SIDE
2943 C SKIRTS (+2, -2, +4, -4) AND THE DATA SURFACE BACK SKIRTS
2944 C (+3, -3) SHOULD ONLY BE PLOTTED AFTER THE DATA SURFACE
2945 C HAS BEEN CONSTRUCTED USING SURFACE. THIS ENTRY DOES NOT
2946 C PLOT THE Z SURFACE (HERE THE DATA ARE CALLED ZZ TO AVOID
2947 C DECLARATION CONFLICTS). HOWEVER Z(I,J) ARE NEEDED TO
2948 C DEFINE THE SKIRT POSITIONS.
2949 C ZZ REAL - ARRAY CONTAINING THE DATA TO BE PLOTTED AS A
2950 C SURFACE
2951 C NX INTEGER - DIMENSION AND RANGE OF I IN Z(I,J)

```

```

2952 C      NY      INTEGER - DIMENSION AND RANGE OF J IN Z(I,J)
2953 C      MODE2   INTEGER - MODE2 = 1(-1) LOWER (UPPER) FRONT SKIRT
2954 C                        = 2(-2) LOWER (UPPER) RIGHT SKIRT
2955 C                        = 3(-3) LOWER (UPPER) BACK SKIRT
2956 C                        = 4(-4) LOWER (UPPER) LEFT SKIRT
2957 C      SURFAC (Z, NX, NY, MODE3) - CONSTRUCTS AND PLOTS THE DATA SUR-
2958 C      FACE AND SETS THE HIDDEN LINE ARRAYS WHICH ARE USED IN
2959 C      SURFSK.
2960 C      Z        REAL    - ARRAY DIMENSIONED (NX,NY) CONTAINING THE DATA
2961 C                        TO BE PLOTTED AS A SURFACE
2962 C      NX      INTEGER - DIMENSION AND RANGE OF I IN Z(I,J)
2963 C      NY      INTEGER - DIMENSION AND RANGE OF J IN Z(I,J)
2964 C      MODE3   INTEGER - MODE3 = 1 PLOTS THE UPPER SURFACE
2965 C                        =-1 PLOTS THE LOWER SURFACE
2966 C
2967 C      PARAMETER (NPT=101)
2968 C      LOGICAL  SW(1024)
2969 C      REAL     XX(8), YY(8)
2970 C      REAL     X(8), Y(8), H(1280), G(1280)
2971 C      REAL     AJ1(NPT), AJN(NPT), AI1(NPT), AIN(NPT), ZVAL(NPT)
2972 C      REAL     RDZVB(NPT), RDZTV(NPT)
2973 C      LOGICAL  SWITCH
2974 C      REAL     Z(NX,NY), X1(NPT), X2(NPT), Y1(NPT), Y2(NPT)
2975 C      REAL     ZZ(NX, NY)
2976 C      COMMON /SURCMN/ X, Y, ZMIN, ZMAX, RDZ, RNXM1, RNYM1, NNX, NNY
2977 C
2978 C      INITIALIZE HIDDEN LINE ARRAYS.
2979 C
2980 C      NNX = NX
2981 C      NNY = NY
2982 C      IF(MAX0(NX,NY).GT.NPT) STOP 76
2983 C      YMIN = 1.E+25
2984 C      YMAX = -1.E+25
2985 C      DO 3 I = 1, 8
2986 C      YMIN = AMIN1(YMIN, YY(I))
2987 C      YMAX = AMAX1(YMAX, YY(I))
2988 C      DO 1 I = 1, 1280
2989 C      G(I) = YMAX
2990 C      H(I) = YMIN
2991 C
2992 C      FOR SPECIAL EFFECTS ZBOT AND ZTOP MAY DIFFER FROM ZMIN AND ZMAX.
2993 C
2994 C      ZBOT = ZZMIN
2995 C      ZTOP = ZZMAX
2996 C      ZMIN = ZZMIN
2997 C      ZMAX = ZZMAX
2998 C      RDZ = 1.0/(ZTOP - ZBOT)
2999 C      DO 2 I = 1, 8
3000 C      X(I) = XX(I)
3001 C      Y(I) = YY(I)
3002 C      RNXM1 = 1.0/FLOAT(NX - 1)
3003 C      RNYM1 = 1.0/FLOAT(NY - 1)
3004 C      RETURN
3005 C
3006 C

```

```

3007 C -----
3008 C
3009 C     ENTRY SFRAME (MODE1)
3010 C
3011 C     * * * * *
3012 C
3013 C     DESCRIPTION: SURFRM PLOTS THE FRAME OF THE RECTANGULAR PARALLELO-
3014 C     PIPED PLOTTING REGION.
3015 C
3016 C     ARGUMENTS:
3017 C     MODE1  INTEGER          MODE1 = 1  PLOTS + AT THE VERTICES.
3018 C                               MODE1 = 2  ALSO PLOTS LINE SEGMENTS
3019 C                               CONNECTING THE VERTICES.
3020 C
3021 C     * * * * *
3022 C
3023 C     CALL CHPLOT (X, Y, '+', 1, 1, 8)
3024 C
3025 C     NOW CHECK THE VARIOUS LINES.
3026 C
3027 C     IF (MODE1 .LE. 1) GO TO 12
3028 C     CALL SURF5 (X(1), Y(1), X(2), Y(2))
3029 C     CALL SURF5 (X(6), Y(6), X(2), Y(2))
3030 C     CALL SURF5 (X(6), Y(6), X(5), Y(5))
3031 C     CALL SURF5 (X(1), Y(1), X(5), Y(5))
3032 C     IF (Y(7).GT.Y(5) .OR. X(7).LT.X(5))
3033 C 1       CALL SURF5 (X(5), Y(5), X(7), Y(7))
3034 C     IF (Y(8).GT.Y(6) .OR. X(8).GT.X(6))
3035 C 1       CALL SURF5 (X(8), Y(8), X(6), Y(6))
3036 C     IF (Y(3).LT.Y(1) .OR. X(3).LT.X(1))
3037 C 1       CALL SURF5 (X(1), Y(1), X(3), Y(3))
3038 C     IF (Y(4).LT.Y(2) .OR. X(4).GT.X(2))
3039 C 1       CALL SURF5 (X(2), Y(2), X(4), Y(4))
3040 C     IF (Y(7).GE.Y(5) .AND. Y(8).GE.Y(6))
3041 C 1       CALL SURF5 (X(8), Y(8), X(7), Y(7))
3042 C     IF (X(8).GE.X(6) .AND. X(4).GE.X(2))
3043 C 1       CALL SURF5 (X(8), Y(8), X(4), Y(4))
3044 C     IF (Y(4).LE.Y(2) .AND. Y(3).LE.Y(1))
3045 C 1       CALL SURF5 (X(4), Y(4), X(3), Y(3))
3046 C     IF (X(3).LE.X(1) .AND. X(7).LE.X(5))
3047 C 1       CALL SURF5 (X(7), Y(7), X(3), Y(3))
3048 C 12      CONTINUE
3049 C     RETURN
3050 C
3051 C
3052 C -----
3053 C
3054 C     ENTRY SSKIRT (ZZ, NX, NY, MODE2)
3055 C
3056 C
3057 C     * * * * *
3058 C
3059 C     DESCRIPTION: SURFSK PLOTS ANY OF THE SKIRTS WHICH MAY BE DESIRED.
3060 C     ONLY ONE SKIRT IS PLOTTED PER CALL AND THE HIDDEN LINE ALGORITHM
3061 C     IS INVOKED. THEREFORE, THE SIDE SKIRTS (+2, -2, +4, -4) AND THE

```

```

3062 C   BACK SKIRTS (+3, -3) SHOULD ONLY BE PLOTTED AFTER THE DATA SURFACE
3063 C   HAS BEEN CONSTRUCTED USING SURFACE. THIS ENTRY DOES NOT PLOT THE Z
3064 C   SURFACE (HERE THE DATA ARE CALLED ZZ TO AVOID DECLARATION CON-
3065 C   FFLICTS). HOWEVER Z(I,J) ARE NEEDED TO DEFINE THE SKIRT POSITIONS.
3066 C
3067 C   ARGUMENTS:
3068 C   ZZ      REAL ARRAY (NX,NY)  THE DATA TO BE PLOTTED AS A SURFACE. I
3069 C   NX      INTEGER              DIMENSION AND RANGE OF I IN Z(I,J). I
3070 C   NY      INTEGER              DIMENSION AND RANGE OF J IN Z(I,J). I
3071 C   MODE2   INTEGER              MODE2 = 1(-1) LOWER (UPPER) FRONT SKIRT
3072 C                                   MODE2 = 2(-2) LOWER (UPPER) RIGHT SKIRT
3073 C                                   MODE2 = 3(-3) LOWER (UPPER) BACK SKIRT
3074 C                                   MODE2 = 4(-4) LOWER (UPPER) LEFT SKIRT
3075 C
3076 C   * * * * *
3077 C
3078 C       NSKIRT = IABS(MODE2)
3079 C       GO TO (21, 22, 23, 22), NSKIRT
3080 C
3081 C   THE FRONT SKIRT IS ADDED.
3082 C
3083 C   21   CONTINUE
3084 C       XL = X(1)
3085 C       XR = X(2)
3086 C       YL = Y(1)
3087 C       YR = Y(2)
3088 C       IF (MODE2 .GT. 0) GO TO 211
3089 C       XL = X(5)
3090 C       XR = X(6)
3091 C       YL = Y(5)
3092 C       YR = Y(6)
3093 C   211  DO 212 I = 1, NX
3094 C       F11 = FLOAT(I-1)/FLOAT(NX-1)
3095 C       F1N = FLOAT(NX-I)/FLOAT(NX-1)
3096 C       XA = XR*F11 + XL*F1N
3097 C       YA = YR*F11 + YL*F1N
3098 C       CALL SURF4 (XB,YB, I,1, ZZ(I,1))
3099 C   212  CALL SURF5 (XA, YA, XB, YB)
3100 C       RETURN
3101 C
3102 C   THE RIGHT OR LEFT SIDE SKIRT IS ADDED. IF IT IS OBSCURED THE
3103 C   HIDDEN LINE ALGORITHMS ARE USED SO CALL ONLY AFTER SURFACE IS USED
3104 C   22   NSK = (4 - NSKIRT)/2
3105 C       I = 1 + (NX-1)*NSK
3106 C       XST = X(NSK+1)
3107 C       YST = Y(NSK+1)
3108 C       XND = X(NSK+3)
3109 C       YND = Y(NSK+3)
3110 C       IF (MODE2 .GT. 0) GO TO 221
3111 C       XST = X(NSK+5)
3112 C       YST = Y(NSK+5)
3113 C       XND = X(NSK+7)
3114 C       YND = Y(NSK+7)
3115 C   221  DO 222 J = 1, NY
3116 C       FJ1 = FLOAT(J-1)/FLOAT(NY-1)

```

```

3117      FJN = FLOAT(NY-J)/FLOAT(NY-1)
3118      XA = XND*FJ1 + XST*FJN
3119      YA = YND*FJ1 + YST*FJN
3120      CALL SURF4 (XB,YB, I,J, ZZ(I,J))
3121 C
3122 C      THE SKIRT MAY BE OBSCURED. CHECK ON TOP OR BOTTOM.
3123 C
3124      K = XB + 0.5
3125      IF (MODE2 .GT. 0) YB = AMIN1(YB, G(K))
3126      IF (MODE2 .LT. 0) YB = AMAX1(YB, H(K))
3127      IF (MODE2.GT.0 .AND. YA.LT.G(K)) CALL SURF5 (XA, YA, XB, YB)
3128      IF (MODE2.LT.0 .AND. YA.GT.H(K)) CALL SURF5 (XA, YA, XB, YB)
3129      222 CONTINUE
3130      RETURN
3131 C
3132 C      THE BACK SKIRT IS ADDED (ASSUMED CALLED AFTER SURFACE).
3133 C
3134      23 CONTINUE
3135      XL = X(3)
3136      YL = Y(3)
3137      XR = X(4)
3138      YR = Y(4)
3139      IF (MODE2.GT.0) GO TO 231
3140      XL = X(7)
3141      YL = Y(7)
3142      XR = X(8)
3143      YR = Y(8)
3144      231 DO 232 I = 1, NX
3145          FII = FLOAT(I-1)/FLOAT(NX-1)
3146          FIN = FLOAT(NX-I)/FLOAT(NX-1)
3147          XA = XR*FII + XL*FIN
3148          YA = YR*FII + YL*FIN
3149          CALL SURF4 (XB,YB, I,NY, ZZ(I,NY))
3150          K = XB + 0.5
3151          IF (MODE2 .GT. 0) YB = AMIN1 (YB, G(K))
3152          IF (MODE2 .LT. 0) YB = AMAX1 (YB, H(K))
3153          IF (FLOAT(MODE2)*(YB-YA) .GT. 0.0) CALL SURF5(XA, YA, XB, YB)
3154      232 CONTINUE
3155      RETURN
3156 C
3157 C
3158 C -----
3159 C
3160      ENTRY SURFAC (Z, NX, NY, MODE3)
3161 C
3162 C      * * * * *
3163 C
3164 C      DESCRIPTION: SURFACE CONSTRUCTS AND PLOTS THE DATA SURFACE AND SET
3165 C      THE HIDDEN LINE ARRAYS WHICH ARE USED IN SURFSK.
3166 C
3167 C      ARGUMENTS:
3168 C      Z      REAL ARRAY (NX,NY) THE DATA TO BE PLOTTED AS A SURFACE. I
3169 C      NX      INTEGER           DIMENSION AND RANGE OF I IN Z(I,J). I
3170 C      NY      INTEGER           DIMENSION AND RANGE OF J IN Z(I,J). I
3171 C      MODE3  INTEGER           MODE3 = 1 PLOTS THE UPPER SURFACE I

```



```

3227      YA = AMAX1(Y1(I), H(IXH))
3228      YB = AMAX1(Y2(I), H(IXH))
3229      IF (YB.GT.YA) CALL SURF5 (X1(I),YA, X2(I),YB)
3230      GO TO 35
3231 39      YA = AMIN1(Y1(I), G(IXH))
3232      YB = AMIN1(Y2(I), G(IXH))
3
  233      IF (YB.LT.YA) CALL SURF5 (X1(I),YA, X2(I),YB)
3234      GO TO 35
3235 C
3236 C      THIS IS A SLANTED LINE SEGMENT.
3237 C
3238 38      DIX = 1.0/FLOAT(IXB - IXA)
3239      SWITCH = .TRUE.
3240      NSWT = IXB - IXA + 1
3241      DO 36 K = IXA, IXB
3242      SW(K) = .TRUE.
3243      IF (X2(I).GT.X1(I)) YK = Y1(I)*(IXB-K)*DIX + Y2(I)*(K-IXA)*DIX
3244      IF (X2(I).LT.X1(I)) YK = Y2(I)*(IXB-K)*DIX + Y1(I)*(K-IXA)*DIX
3245      IF ((MODE3.LT.0 .AND. G(K).GE.YK-0.5)
3246 1      .OR. (MODE3.GT.0 .AND. H(K).LE.YK+0.5)) GO TO 36
3247      SWITCH = .FALSE.
3248      SW(K) = .FALSE.
3249      NSWT = NSWT - 1
3250 36      CONTINUE
3251      IF (SWITCH) CALL SURF5 (X1(I), Y1(I), X2(I), Y2(I))
3252      IF (SWITCH .OR. NSWT.EQ.0) GO TO 35
3253 C
3254 C      PART OF THE LINE IS OBSCURED SO WE NEED TO PLOT SEGMENTS.
3255 C
3256      SW(IXA-1) = .FALSE.
3257      SW(IXB+1) = .FALSE.
3258      DO 41 K = IXA, IXB
3259      IF (X2(I).GT.X1(I)) YK = Y1(I)*(IXB-K)*DIX + Y2(I)*(K-IXA)*DIX
3260      IF (X2(I).LT.X1(I)) YK = Y2(I)*(IXB-K)*DIX + Y1(I)*(K-IXA)*DIX
3261      IF (.NOT.SW(K) .OR. SW(K-1)) GO TO 51
3262      XA = K
3263      YA = YK
3264      IF (K.EQ.IXA) GO TO 51
3265      IF (MODE3.GT.0) YA = AMIN1 (H(K), YK)
3266      IF (MODE3.LT.0) YA = AMAX1 (G(K), YK)
3267 51      IF (.NOT.SW(K) .OR. SW(K+1)) GO TO 41
3268      IF (K.EQ.IXB) GO TO 52
3269      IF (MODE3.GT.0) YK = AMIN1 (H(K), YK)
3270      IF (MODE3.LT.0) YK = AMAX1 (G(K), YK)
3271 52      CALL SURF5 (XA,YA,FLOAT(K),YK)
3272 41      CONTINUE
3273 35      CONTINUE
3274 37      CONTINUE
3275 C
3276 C      PLOT THE I TO I+1 LINE SEGMENTS IF NOT HIDDEN.
3277 C
3278      DO 33 I = 2, NX
3279      IXB = X2(I) + 0.5
3280      IXA = X2(I-1)
3281      DIX = 1.0/FLOAT(IXB - IXA)

```

```

3282 SWITCH = .TRUE.
3283 NSWT = IXB - IXA + 1
3284 DO 34 K = IXA, IXB
3285 SW(K) = .TRUE.
3286 YK = Y2(I-1)*(IXB-K)*DIX + Y2(I)*(K-IXA)*DIX
3287 IF (MODE3.GT.0) H(K) = AMAX1(H(K), YK)
3288 IF (MODE3.LT.0) G(K) = AMIN1(G(K), YK)
3289 IF ((MODE3.LT.0 .AND. G(K).GE.YK-0.5)
3290     1 .OR. (MODE3.GT.0 .AND. H(K).LE.YK+0.5)) GO TO 34
3291 SWITCH = .FALSE.
3292 SW(K) = .FALSE.
3293 NSWT = NSWT - 1
3294 34 CONTINUE
3295 IF (SWITCH) CALL SURF5 (X2(I-1), Y2(I-1), X2(I), Y2(I))
3296 IF (SWITCH .OR. NSWT.EQ.0) GO TO 33
3297 C
3298 C PART OF THE LINE IS OBSCURED SO WE NEED TO PLOT SEGMENTS.
3299 C
3300 SW(IXA-1) = .FALSE.
3301 SW(IXB+1) = .FALSE.
3302 DO 44 K = IXA, IXB
3303 YK = Y2(I-1)*(IXB-K)*DIX + Y2(I)*(K-IXA)*DIX
3304 IF (SW(K) .AND. .NOT.SW(K-1)) XA = K
3305 IF (SW(K) .AND. .NOT.SW(K-1)) YA = YK
3306 IF (SW(K) .AND. .NOT.SW(K+1)) CALL SURF5 (XA, YA, FLOAT(K), YK)
3307 44 CONTINUE
3308 33 CONTINUE
3309 C
3310 31 CONTINUE
3311 RETURN
3312 END

```

```

3313 SUBROUTINE SURF4 (XVAL, YVAL, I, J, ZIN)
3314 C
3315 C * * * * *
3316 C
3317 C THIS IS AN AUXILIARY ROUTINE TO SURFACE. IT PERFORMS A TRILINEAR I
3318 C TERPOLATION IN THE 3D RECTANGLE OUTLINED BT (X(I), Y(I)) ON THE
3319 C PLOTTING SURFACE FOR I = 1, ..., 8. THE TEXT WILL BE SUBSTITUTED
3320 C IN LINE WHEN THE ASC COMPILER BUG HAS BEEN FIXED.
3321 C
3322 C * * * * *
3323 C
3324 C REAL X(8), Y(8)
3325 C COMMON /SURCMN/ X, Y, ZMIN, ZMAX, RDZ, RNXM1, RNYM1, NNX, NNY
3326 C
3327 C F11 = FLOAT(I-1)*RNXM1
3328 C FJ1 = FLOAT(J-1)*RNYM1
3329 C FIN = FLOAT(NNX-I)*RNXM1
3330 C FJN = FLOAT(NNY-J)*RNYM1
3331 C ZVAL = AMAX1 (ZIN, ZMIN)
3332 C ZVAL = AMIN1 (ZVAL, ZMAX)
3333 C XVAL = RDZ*(ZVAL-ZMIN)*(FJN*(F11*X(6) + FIN*X(5))
3334 C 1 + FJ1*(F11*X(8) + FIN*X(7)))
3335 C 2 + RDZ*(ZMAX-ZVAL)*(FJN*(F11*X(2) + FIN*X(1))
3336 C 3 + FJ1*(F11*X(4) + FIN*X(3)))
3337 C YVAL = RDZ*(ZVAL-ZMIN)*(FJN*(F11*Y(6) + FIN*Y(5))
3338 C 1 + FJ1*(F11*Y(8) + FIN*Y(7)))
3339 C 2 + RDZ*(ZMAX-ZVAL)*(FJN*(F11*Y(2) + FIN*Y(1))
3340 C 3 + FJ1*(F11*Y(4) + FIN*Y(3)))
3341 C RETURN
3342 C
3343 C * * * * *
3344 C
3345 C THIS SUBROUTINE IS A GENERAL-PURPOSE GRAPHICS INTERFACE FOR USE
3346 C WITH THE SURFACE SUBROUTINE FOR 3D SURFACE PLOTS WITH HIDDEN LINES
3347 C REMOVED.
3348 C
3349 C * * * * *
3350 C
3351 C ENTRY SURF5 (X1, Y1, X2, Y2)
3352 C
3353 C IF ( ((X1-X2)**2 + (Y1-Y2)**2) .GT. 2.25)
3354 C 1 CALL LINE (X1, Y1, X2, Y2)
3355 C RETURN
3356 C END

```

```
3357          SUBROUTINE SYMBOL (XCHAR, YCHAR, CHARAC)
3358 C
3359          COMMON /DEV TYP/ IDEVIC, LSW, LTSW, XYCOORD(4), LUOUT, LPAGE
3360          COMMON/GRFTYP/ANGLE, IRVRSE, CHSIZE(9), ITKWIT, LUDIAG, LUHSET
3361          CHARACTER*1 CHARAC
3362 C
3363 C      THIS SUBROUTINE USES THE HARWARE CHARACTERS TO PLOT DATA
3364 C      POINTS AT THE LOCATION SPECIFIED BY (XCHAR,YCHAR) IN THE PLOTTING
3365 C      REGION DETERMINED BY XYCOORD.
3366 C
3367          IF (IDEVIC.EQ.4) CALL IOWAIT(8)
3368          X = XYCOORD(1)*XCHAR + XYCOORD(2) - CHSIZE(2)*CHSIZE(3)
3369          Y = XYCOORD(3)*YCHAR + XYCOORD(4) - CHSIZE(2)*CHSIZE(4)
3370          CALL PUTCH (X, Y, CHARAC)
3371          RETURN
3372          END
```

```

3373     SUBROUTINE VIEWTR(XDC,WINDOW,VERTEX,NV,EDGE,ESPEC,NE,
3374     . POLY,PSPEC,NP)
3375 C
3376 C THIS SUBROUTINE PLOTS EDGES AND POLYGONS WHICH ARE WITHIN THE WINDOW
3377 C EDGE SEGMENTS WHICH CROSS A PORTION OF THE WINDOW.
3378 C
3379     REAL VERTEX(3,NV), WINDOW(4), XYZ(3,8)
3380     INTEGER EDGE(2,NE), ESPEC(4,NE), POLY(8,NP), PSPEC(4,NP)
3381     REAL U1,V1,U2,V2,X(8),Y(8),XDC(4,4)
3382 C
3383 C DEFINE THE WINDOW SPACE
3384 C
3385     CALL DEFINE(WINDOW(1),WINDOW(2),WINDOW(3),WINDOW(4))
3386 C
3387     DOZ=10000.
3388     FANGLE=90.
3389     TANAL = 1.0
3390     WCX = 0.5 * (WINDOW(1)+WINDOW(3))
3391     WCY = 0.5 * (WINDOW(2)+WINDOW(4))
3392 C
3393 C PLOT ONE EDGE AT A TIME
3394 C
3395     LWD = -1
3396     LCL = -1
3397     LFL = -1
3398     DO 150 J=1,NE
3399 C
3400 C MOVE VERTICES INTO A LOCAL WORK ARRAY
3401 C
3402     DO 151 I = 1, 2
3403     DO 151 K = 1, 3
3404 151 XYZ(K,I) = VERTEX(K,EDGE(I,J))
3405 C
3406 C APPLY THE TRANSFORM
3407 C
3408     DO 152 I = 1, 2
3409     U1 = XYZ(1,I)
3410     U2 = XYZ(2,I)
3411     U3 = XYZ(3,I)
3412     XYZ(1,I) = XDC(1,1)*U1 + XDC(1,2)*U2 + XDC(1,3)*U3 + XDC(1,4)
3413     XYZ(2,I) = XDC(2,1)*U1 + XDC(2,2)*U2 + XDC(2,3)*U3 + XDC(2,4)
3414 152 XYZ(3,I) = XDC(3,1)*U1 + XDC(3,2)*U2 + XDC(3,3)*U3 + XDC(3,4)
3415 C
3416 C ADD PERSPECTIVE
3417 C
3418     D = DOZ/((DOZ+XYZ(3,1))*TANAL)
3419     U1=(XYZ(1,1)-WCX)*D + WCX
3420     V1=(XYZ(2,1)-WCY)*D + WCY
3421     D = DOZ/((DOZ+XYZ(3,2))*TANAL)
3422     U2=(XYZ(1,2)-WCX)*D + WCX
3423     V2=(XYZ(2,2)-WCY)*D + WCY
3424 C
3425 C CLIP LEFT EDGE
3426 C
3427     IF (U1.GE.WINDOW(1).AND.U2.GE.WINDOW(1)) GO TO 50

```

```

3428         IF (U1.LT.WINDOW(1).AND.U2.LT.WINDOW(1)) GO TO 150
3429         IF (U1.GT.WINDOW(1)) GO TO 40
3430         V1 = (V1-V2)*U1/(U2-U1)+V1
3431         U1 = WINDOW(1)
3432         GO TO 50
3433 40        V2 = (V2-V1)*U2/(U1-U2)+V2
3434         U2 = WINDOW(1)
3435         C
3436         C CLIP RIGHT EDGE
3437         C
3438 50        IF (U1.LE.WINDOW(3).AND.U2.LE.WINDOW(3)) GO TO 70
3439         IF (U1.GT.WINDOW(3).AND.U2.GT.WINDOW(3)) GO TO 150
3440         IF (U1.GT.WINDOW(3)) GO TO 51
3441         V2 = (V2-V1)*(WINDOW(3)-U1)/(U2-U1)+V1
3442         U2 = WINDOW(3)
3443         GO TO 70
3444 51        V1 = (V1-V2)*(WINDOW(3)-U2)/(U1-U2)+V2
3445         U1 = WINDOW(3)
3446         C
3447         C CLIP BOTTOM EDGE
3448         C
3449 70        IF (V1.GE.WINDOW(2).AND.V2.GE.WINDOW(2)) GO TO 90
3450         IF (V1.LT.WINDOW(2).AND.V2.LT.WINDOW(2)) GO TO 150
3451         IF (V1.GT.WINDOW(2)) GO TO 80
3452         U1 = (U1-U2)*V1/(V2-V1)+U1
3453         V1 = WINDOW(2)
3454         GO TO 90
3455 80        U2 = (U2-U1)*V2/(V1-V2)+U2
3456         V2 = WINDOW(2)
3457         C
3458         C CLIP TOP EDGE
3459         C
3460 90        IF (V1.LE.WINDOW(4).AND.V2.LE.WINDOW(4)) GO TO 110
3461         IF (V1.GT.WINDOW(4).AND.V2.GT.WINDOW(4)) GO TO 150
3462         IF (V1.GT.WINDOW(4)) GO TO 100
3463         U2 = (U2-U1)*(WINDOW(4)-V1)/(V2-V1)+U1
3464         V2 = WINDOW(4)
3465         GO TO 110
3466 100       U1 = (U1-U2)*(WINDOW(4)-V2)/(V1-V2)+U2
3467         V1 = WINDOW(4)
3468         C
3469         C PLOT THE EDGE
3470         C
3471 110       IF(LWD.NE.ESPEC(1,J)) THEN
3472             LWD = ESPEC(1,J)
3473             CALL LINWID(LWD)
3474         ENDIF
3475         IF(LCL.NE.ESPEC(2,J)) THEN
3476             LCL = ESPEC(2,J)
3477             CALL COLOR(LCL)
3478         ENDIF
3479         CALL LINE(U1, V1, U2, V2)
3480 150       CONTINUE
3481         C
3482         C PLOT POLYGONS ON TOP OF EDGES

```

```

3483 C
3484 DO 10 J = 1, NP
3485 NUM=0
3486 C
3487 C DETERMINE THE NUMBER OF VERTICES (NON-ZERO ENTRIES) IN THE
3488 C POLYGON
3489 C
3490 DO 15 I=1,8
3491 IF(POLY(I,J).NE.0) NUM=NUM+1 ;NUMBER OF VERTICES IN POLYGON
3492 15 CONTINUE
3493 C
3494 C MOVE THE VERTICES INTO A LOCAL WORK ARRAY
3495 C
3496 DO 61 I = 1, NUM
3497 DO 61 K = 1, 3
3498 61 XYZ(K,I) = VERTEX(K,POLY(I,J))
3499 C
3500 C APPLY THE TRANSFORM
3501 C
3502 DO 62 I = 1, NUM
3503 U1 = XYZ(1,I)
3504 U2 = XYZ(2,I)
3505 U3 = XYZ(3,I)
3506 XYZ(1,I) = XDC(1,1)*U1 + XDC(1,2)*U2 + XDC(1,3)*U3 + XDC(1,4)
3507 XYZ(2,I) = XDC(2,1)*U1 + XDC(2,2)*U2 + XDC(2,3)*U3 + XDC(2,4)
3508 62 XYZ(3,I) = XDC(3,1)*U1 + XDC(3,2)*U2 + XDC(3,3)*U3 + XDC(3,4)
3509 C
3510 C ADD PERSPECTIVE
3511 C
3512 DO 60 IV=1,NUM
3513 D = DOZ/((DOZ+XYZ(3,IV))*TANAL)
3514 X(IV) = (XYZ(1,IV) - WCX)*D + WCX
3515 60 Y(IV) = (XYZ(2,IV) - WCY)*D + WCY
3516 C
3517 C PLOT THE POLYGON
3518 C
3519 IF(PSPEC(1,J).NE.LWD) THEN
3520 LWD = PSPEC(1,J)
3521 CALL LINWID(LWD)
3522 ENDIF
3523 IF(PSPEC(2,J).NE.LCL) THEN
3524 LCL = PSPEC(2,J)
3525 CALL COLOR(LCL)
3526 ENDIF
3527 IF(PSPEC(3,J).NE.LFL) THEN
3528 LFL = PSPEC(3,J)
3529 CALL FILTYP(LFL)
3530 ENDIF
3531 IF(PSPEC(4,J).GE.0) THEN
3532 CALL PLYGON(X, Y, NUM)
3533 ELSE
3534 CALL LINES(X(1), Y(1), X(2), Y(2), NUM-1)
3535 ENDIF
3536 10 CONTINUE ;GET NEXT POLYGON
3537 RETURN

```

3538 END
3539 C


```
3540 SUBROUTINE VOLFRM (MODE1)
3541 C
3542 C THIS ENTRY PUTS A FRAME AROUND THE REGION IF DESIRED.
3543 C FIRST PLOT + AT ALL OF THE VERTICES.
3544 C
3545 COMMON/VOLCOM/XR,YR,A,TP,AP,DX,DY,DZ,X,Y,Z,SC1,FSEG,NTT
3546 LOGICAL A(1)
3547 INTEGER TP,AP
3548 DIMENSION XR(8),YR(8),DX(4),DY(4),DZ(4),X(4),Y(4),Z(4),SC1(4)
3549 DO 101 I = 1, 8
3550 101 CALL CHPLOT (SC1, XR(I), YR(I), '+', 1,1,1)
3551 C
3552 C NOW CHECK THE VARIOUS LINES.
3553 C
3554 IF (MODE1 .LE. 1) GO TO 102
3555 CALL LINE (XR(1),YR(1), XR(2),YR(2))
3556 CALL LINE (XR(6),YR(6), XR(2),YR(2))
3557 CALL LINE (XR(6),YR(6), XR(5),YR(5))
3558 CALL LINE (XR(1),YR(1), XR(5),YR(5))
3559 CALL VOLUM8 (XR(4),YR(4), XR(3),YR(3), T, NT)
3560 CALL VOLUM8 (XR(4),YR(4), XR(8),YR(8), T, NT)
3561 CALL VOLUM8 (XR(7),YR(7), XR(8),YR(8), T, NT)
3562 CALL VOLUM8 (XR(7),YR(7), XR(3),YR(3), T, NT)
3563 CALL VOLUM8 (XR(4),YR(4), XR(2),YR(2), T, NT)
3564 CALL VOLUM8 (XR(1),YR(1), XR(3),YR(3), T, NT)
3565 CALL VOLUM8 (XR(6),YR(6), XR(8),YR(8), T, NT)
3566 CALL VOLUM8 (XR(7),YR(7), XR(5),YR(5), T, NT)
3567 102 CONTINUE
3568 RETURN
3569 END
3570 C
```

```

3571     SUBROUTINE VOLUMO (Q, R, NX, NY, NZ)
3572 C
3573     REAL      X(8), Y(8), Q(8), R(8)
3574     RNX = 1.0/FLOAT(NX-1)
3575     RNY = 1.0/FLOAT(NY-1)
3576     RNZ = 1.0/FLOAT(NZ-1)
3577     FNX = NX
3578     FNY = NY
3579     FNZ = NZ
3580     DO 1 I = 1, 8
3581     X(I) = Q(I)
3582 1     Y(I) = R(I)
3583     RETURN
3584 C
3585 C
3586     ENTRY VOLUM6 (XA, YA, XG, YG, ZG)
3587 C
3588     DI1 = (FNX - XG)*RNX
3589     DJ1 = (FNY - YG)*RNY
3590     DK1 = (FNZ - ZG)*RNZ
3591     DIN = 1.0 - DI1
3592     DJN = 1.0 - DJ1
3593     DKN = 1.0 - DK1
3594     XA = ((X(1)*DI1+X(2)*DIN)*DJ1 + (X(3)*DI1+X(4)*DIN)*DJN)*DK1
3595 1     + ((X(5)*DI1+X(6)*DIN)*DJ1 + (X(7)*DI1+X(8)*DIN)*DJN)*DKN
3596     YA = ((Y(1)*DI1+Y(2)*DIN)*DJ1 + (Y(3)*DI1+Y(4)*DIN)*DJN)*DK1
3597 1     + ((Y(5)*DI1+Y(6)*DIN)*DJ1 + (Y(7)*DI1+Y(8)*DIN)*DJN)*DKN
3598     RETURN
3599     END
3600 C

```

```
3601      SUBROUTINE VOLUM7 (CLEVEL, F00,F10,F11,F01, NSEG, X, Y)
3602 C
3603      REAL      X(4), Y(4)
3604 C
3605 C      AT LEAST ONE CROSSING WILL BE FOUND. TEST THE LOWER SEGMENT FIRST.
3606 C
3607      ISEG = 0
3608      IF (F00.NE.F10) DX = (CLEVEL-F00)/(F10-F00)
3609      IF (F00.EQ.F10) DX = 0.0
3610      IF (DX.GE.1.0 .OR. DX.LE.0.0) GO TO 2
3611 C
3612 C      RECORD THE LOWER CROSSING.
3613 C
3614      ISEG = ISEG + 1
3615      X(ISEG) = DX
3616      Y(ISEG) = 0.0
3617 C
3618 C      TEST THE RIGHT SEGMENT.
3619 C
3620 2      IF (F10.NE.F11) DY = (CLEVEL-F10)/(F11-F10)
3621      IF (F10.EQ.F11) DY = 0.0
3622      IF (DY.GE.1.0 .OR. DY.LE.0.0) GO TO 3
3623 C
3624 C      RECORD THE RIGHT SIDE CROSSING.
3625 C
3626      ISEG = ISEG + 1
3627      X(ISEG) = 1.0
3628      Y(ISEG) = DY
3629 C
3630 C      TEST THE TOP SEGMENT.
3631 C
3632 3      IF (F11.NE.F01) DX = (CLEVEL-F01)/(F11-F01)
3633      IF (F11.EQ.F01) DX = 1.0
3634      IF (DX.GE.1.0 .OR. DX.LE.0.0) GO TO 4
3635 C
3636 C      RECORD THE TOP CROSSING.
3637 C
3638      ISEG = ISEG + 1
3639      X(ISEG) = DX
3640      Y(ISEG) = 1.0
3641 C
3642 C      TEST THE LEFT SIDE SEGMENT.
3643 C
3644 4      IF (F01.NE.F00) DY = (CLEVEL-F00)/(F01 - F00)
3645      IF (F01.EQ.F00) DY = 1.0
3646      IF (DY.GE.1.0 .OR. DY.LE.0.0) GO TO 5
3647 C
3648 C      RECORD THE LEFT SIDE CROSSING.
3649 C
3650      ISEG = ISEG + 1
3651      X(ISEG) = 0.0
3652      Y(ISEG) = DY
3653 C
3654 C      SAVE ANY LINE SEGMENTS FOUND
3655 C
```

```
3656      5      NSEG = ISEG  
3657          RETURN  
3658          END
```

```

3659     SUBROUTINE VOLUM8 (XA, YA, XB, YB, T, NT)
3660   C
3661         COMMON/VOLCOM/XR,YR,A,TP,AP,DX,DY,DZ,P,Q,R,SC1,FSEG,NTT
3662         DIMENSION XR(8),YR(8),DX(4),DY(4),DZ(4),P(4),Q(4),R(4),SC1(4)
3663         LOGICAL T(NT, NT)
3664         DATA      KINC / 1 /
3665         LOGICAL  SW1
3666   C
3667         IF (SQRT((XB-XA)**2 + (YB-YA)**2) .LT. 1.9) RETURN
3668   C
3669   C     CHECK FOR HIDDEN PORTIONS OF THE LINE.
3670   C
3671         RSEG=FLOAT(NTT)/1024.
3672         IAT = RSEG*XA
3673         IBT = RSEG*XB
3674         JAT = RSEG*YA
3675         JBT = RSEG*YB
3676         KI = ABS(XB-XA)
3677         KJ = ABS(YB-YA)
3678         IF (KI.GT.120 .OR. KJ.GT.120) GO TO 1
3679         IF (.NOT.T(IAT,JAT) .OR. .NOT.T(IBT,JBT)) GO TO 1
3680         CALL LINE (XA,YA, XB,YB)
3681         RETURN
3682   C
3683   C     AT LEAST PART OF THE LINE IS HIDDEN. FIND REASONABLE LIMITS.
3684   C
3685   1     KMAX = (MAXO(KI,KJ)/KINC)*KINC + 1
3686         IF (KMAX .LT. 2) RETURN
3687   C
3688   C     NOW CHECK EACH SEGMENT ALONG THE LINE.
3689   C
3690         SW1 = .FALSE.
3691         DO 2 K = 1, KMAX, KINC
3692         FA = FLOAT(K-1)/FLOAT(KMAX-1)
3693         X = XA + FA*(XB - XA)
3694         Y = YA + FA*(YB - YA)
3695         IX = RSEG*X
3696         IY = RSEG*Y
3697         IF (SW1) GO TO 3
3698   C
3699   C     UPDATE THE FIRST POINT IF SW1 IS .FALSE.
3700   C
3701         IF (.NOT.T(IX,IY)) GO TO 3
3702         XST = X
3703         YST = Y
3704         SW1 = .TRUE.
3705   C
3706   C     SET THE END POINT.
3707   C
3708   3     IF (T(IX,IY) .AND. K.NE.KMAX) GO TO 2
3709         IF (.NOT.SW1) GO TO 2
3710         FB = FLOAT(K-1-KINC)/FLOAT(KMAX-1)
3711         XND = XA + FB*(XB - XA)
3712         YND = YA + FB*(YB - YA)
3713         CALL LINE (XST,YST, XND,YND)

```

```
3714      SW1 = .FALSE.  
3715 2     CONTINUE  
3716     RETURN  
3717     END  
3718 C
```

3719 SUBROUTINE VOLUME (MODE, F, NX,NY,NZ, CLEVE, NCL, T, NT)

3720 C

3721 C

3722 C

3723 C

3724 C

3725 C

3726 C

3727 C

3728 C

3729 C

3730 C

3731 C

3732 C

3733 C

3734 C

3735 C

3736 C

3737 C

3739 C

3740 C

3741 C

3742 C

3743 C

3744 C

3745 C

3746 C

3747 C

3748 C

3749 C

3750 C

3751 C

3752 C

3753 C

3754 C

3755 C

3756 C

3757 C

3758 C

3759 C

3760 C

3761 C

3762 C

3763 C

3764 C

3765 C

3766 C

3767 C

3768 C

3769 C

3770 C

3771 C

3772 C

3773 C

THIS SET OF ROUTINES IS DESIGNED TO DRAW ON A DEVICE, SUCH AS A PRINTER, A PLOTTER, THE SC4020 OR THE TEKTRONICS GRAPHICS TERMINAL, A TWO-DIMENSION PROJECTION OF A THREE DIMENSIONAL ARRAY (FIGURE). THE HIDDEN LINES ARE ELIMINATED TO SIMULATE THE EFFECT OF "3D".

THE COLOR OF THE FIGURES (FOR THE SC4020 ONLY) MUST BE SET BY THE USER IN THE CALLING PROGRAM USING THE APPROPRIATE CALLS TO THE GRAPHICS PACKAGE.

THE AXIS ARE LABELED AS FOLLOWS: X IS LEFT TO RIGHT
Y IS INTO THE PAGE
Z IS BOTTOM TO TOP

TO INITIALIZE THE PLOTTING PACKAGE, "VOLSET" MUST BE CALLED. IF A FRAME SURROUNDING THE DRAWING IS WANTED, CALL "VOLFRM". THE ROUTINE LOOPS OVER THE Y AXIS FROM FRONT TO BACK, PLOTTING EACH SUCCESSIVE (X,Z) PLANE. AT THE END OF THE (X,Z) CYCLE, THE ARRAY "T" IS SET TO INDICATE WHAT WILL BE HIDDEN BY THE CURRENT PLANE.

. CALLING SEQUENCE OR OPERATIONAL PROCEDURE:

CALL VOLSET(XP, YP, T, NT)
CALL VOLUME (MODE, F, NX, NY, NZ, CLEVE, NCL)
CALL VOLFRM(MODE1)

. ARGUMENTS (TYPE AND SIGNIFICANCE) AND/OR INITIAL CONDITIONS:

XP CONTAINS THE LOCATIONS OF THE X POSITION OF THE VERTICES (8)
YP CONTAINS THE CORRESPONDING Y VALUES (8)
T IS THE PLOTTING ARRAY (BOOLEAN) TO ELIMINATE THE HIDDEN LINES WHICH IS NT X NT IN SIZE (THIS ARRAY TOGETHER WITH NX,NY,NZ DETERMINE THE FINESSE OF THE PLOT).
MODE DETERMINES WHICH SURFACE IS PLOTTED (USED WITH CLEVE)
ABS(MODE)=1 WILL PLOT THE CONTOUR LEVEL "CLEVE" IN EACH PLANE AND THEN FIXES THE HIDDEN LINE MATRIX "T".
ABS(MODE)=2 SIMPLY FIXES THE HIDDEN LINE MATRIX "T" WITHOUT PLOTTING. THIS IS USEFUL IN AVOIDING MULTIPLE PLOTS OF THE SAME FIXED FIGURE.
MODE<0 IS USED WHEN THE VALUE OF THE FUNCTION ON THE "OUTSIDE" IS LESS THAN "CLEVE".
MODE>0 IS USED WHEN THE VALUE OF THE FUNCTION ON THE "OUTSIDE" IS GREATER THAN "CLEVE".
F IS THE THREE-DIMENSIONAL FIGURE TO BE PLOTTED.
NX,NY,NZ ARE THE DIMENSIONS OF THIS ARRAY <F(NX,NY,NZ)>
CLEVE IS THE CONTOUR SURFACE LEVEL FOR PLOTTING.
NCL IS THE NUMBER OF PAIRS OF (MODE,CLEVE) TO BE PLOTTED.
MODE1 IS THE MODE OF FRAMING 0-> "+" AT THE EIGHT VERTICES;
1-> A BOX FORMED BY LINES.

```

3774 C
3775 COMMON/VOLCOM/XR,YR,A,TP,AP,DX,DY,DZ,X,Y,Z,SC1,FSEG,NTT
3776 LOGICAL T(NT,NT)
3777 DIMENSION XR(8),YR(8),DX(4),DY(4),DZ(4),X(4),Y(4),Z(4),SC1(4)
3778 INTEGER MODE(NCL)
3779 REAL F(NX,NY,NZ), CLEVE(NCL)
3780 REAL XP(8), YP(8)
3781 C
3782 CALL VOLUMO (XR,YR, NX,NY,NZ)
3783 C
3784 C THE OUTER LOOP IS OVER THE I,K SURFACES MOVING BACK.
3785 C
3786 DO 1 J = 1, NY
3787 IF (J.EQ.1) GO TO 2
3788 C
3789 C THEN LOOP OVER THE VARIOUS CONTOUR LEVELS.
3790 C
3791 DO 20 LL = 1, NCL
3792 CLEVEL = CLEVE(LL)
3793 MODEL = MODE(LL)
3794 IF (IABS (MODEL) .EQ. 2) GO TO 20
3795 C
3796 C FIRST COMPUTE THE FRONT-TO-BACK LINE SEGMENTS.
3797 C
3798 DO 3 I = 1, NX
3799 DO 4 K = 2, NZ
3800 FMAX = AMAX1 (F(I,J-1,K-1), F(I,J,K-1), F(I,J,K), F(I,J-1,K))
3801 FMIN = AMIN1 (F(I,J-1,K-1), F(I,J,K-1), F(I,J,K), F(I,J-1,K))
3802 IF (FMAX.LT.CLEVEL .OR. FMIN.GT.CLEVEL) GO TO 4
3803 CALL VOLUM7 (CLEVEL, F(I,J-1,K-1), F(I,J,K-1),
3804 1 F(I,J,K), F(I,J-1,K), NSEG, DY, DZ)
3805 IF (NSEG .LE. 1) GO TO 4
3806 DO 5 L = 1, NSEG
3807 X(L) = FLOAT(I)
3808 Y(L) = FLOAT(J-1) + DY(L)
3809 5 Z(L) = FLOAT(K-1) + DZ(L)
3810 CALL VOLUM6 (XA,YA, X(1),Y(1),Z(1))
3811 CALL VOLUM6 (XB,YB, X(2),Y(2),Z(2))
3812 CALL VOLUM8 (XA,YA, XB,YB, T, NT)
3813 IF (NSEG.NE.4) GO TO 4
3814 CALL VOLUM6 (XA,YA, X(3),Y(3),Z(3))
3815 CALL VOLUM6 (XB,YB, X(4),Y(4),Z(4))
3816 CALL VOLUM8 (XA,YA, XB,YB, T, NT)
3817 4 CONTINUE
3818 3 CONTINUE
3819 DO 6 K = 1, NZ
3820 DO 7 I = 2, NX
3821 FMAX = AMAX1 (F(I-1,J-1,K), F(I,J-1,K), F(I,J,K), F(I-1,J,K))
3822 FMIN = AMIN1 (F(I-1,J-1,K), F(I,J-1,K), F(I,J,K), F(I-1,J,K))
3823 IF (FMAX.LT.CLEVEL .OR. FMIN.GT.CLEVEL) GO TO 7
3824 CALL VOLUM7 (CLEVEL, F(I-1,J-1,K), F(I,J-1,K),
3825 1 F(I,J,K), F(I-1,J,K), NSEG, DX, DY)
3826 IF (NSEG .LE. 1) GO TO 7
3827 DO 8 L = 1, NSEG
3828 X(L) = FLOAT(I-1) + DX(L)

```



```

3829      Y(L) = FLOAT(J-1) + DY(L)
3830      8      Z(L) = FLOAT(K)
3831      CALL VOLUM6 (XA,YA, X(1),Y(1),Z(1))
3832      CALL VOLUM6 (XB,YB, X(2),Y(2),Z(2))
3833      CALL VOLUM8 (XA,YA, XB,YB, T, NT)
3834      IF (NSEG.NE.4) GO TO 7
3835      CALL VOLUM6 (XA,YA, X(3),Y(3),Z(3))
3836      CALL VOLUM6 (XB,YB, X(4),Y(4),Z(4))
3837      CALL VOLUM8 (XA,YA, XB,YB, T, NT)
3838      7      CONTINUE
3839      6      CONTINUE
3840      20     CONTINUE
3841  C
3842  C      NOW PLOT THE X-Z SURFACE CONTOURS BEFORE CORRECTING THE HIDDEN
3843  C      LINE ARRAY.
3844  C
3845      2      DO 21 LL = 1, NCL
3846      CLEVEL = CLEVE(LL)
3847      MODEL = MODE(LL)
3848      IF (IABS (MODEL) .EQ. 2) GO TO 21
3849      DO 9 K = 2, NZ
3850      DO 10 I = 2, NX
3851      FMAX = AMAX1 (F(I-1,J,K-1), F(I,J,K-1), F(I,J,K), F(I-1,J,K))
3852      FMIN = AMIN1 (F(I-1,J,K-1), F(I,J,K-1), F(I,J,K), F(I-1,J,K))
3853      IF (FMAX.LT.CLEVEL .OR. FMIN.GT.CLEVEL) GO TO 10
3854      CALL VOLUM7 (CLEVEL, F(I-1,J,K-1), F(I,J,K-1),
3855      1      F(I,J,K), F(I-1,J,K), NSEG, DX, DZ)
3856      IF (NSEG .LE. 1) GO TO 10
3857      DO 11 L = 1, NSEG
3858      X(L) = FLOAT(I-1) + DX(L)
3859      Y(L) = FLOAT(J)
3860      11     Z(L) = FLOAT(K-1) + DZ(L)
3861      CALL VOLUM6 (XA,YA, X(1),Y(1),Z(1))
3862      CALL VOLUM6 (XB,YB, X(2),Y(2),Z(2))
3863      CALL VOLUM8 (XA,YA, XB,YB, T, NT)
3864      IF (NSEG.NE.4) GO TO 10
3865      CALL VOLUM6 (XA,YA, X(3),Y(3),Z(3))
3866      CALL VOLUM6 (XB,YB, X(4),Y(4),Z(4))
3867      CALL VOLUM8 (XA,YA, XB,YB, T, NT)
3868      10     CONTINUE
3869      9      CONTINUE
3870      21     CONTINUE
3871  C
3872  C      FILL THE HIDDEN LINE ARRAY AFTER PLOTTING IN THE NEW PLANE.
3873  C
3874      DO 22 LL = 1, NCL
3875      CLEVEL = CLEVE(LL)
3876      MODEL = MODE(LL)
3877      DO 12 K = 2, NZ
3878      DO 13 I = 2, NX
3879      IF (MODEL.GT.0 .AND. AMIN1(F(I-1,J,K-1), F(I,J,K-1),
3880      1      F(I-1,J,K), F(I,J,K)).GT.CLEVEL) GO TO 13
3881      IF (MODEL.LT.0 .AND. AMAX1(F(I-1,J,K-1), F(I,J,K-1),
3882      1      F(I-1,J,K), F(I,J,K)).LT.CLEVEL) GO TO 13
3883  C

```

```

8884 C   THERE IS HIDDEN STUFF SOMEWHERE IN THE CELL. FIRST FIND THE LIMITS
8885 C   OF THE CELL.
8886 C
8887       CALL VOLUM6 (X00,Y00, FLOAT(I-1), FLOAT(J), FLOAT(K-1))
8888       CALL VOLUM6 (X10,Y10, FLOAT(I ), FLOAT(J), FLOAT(K-1))
8889       CALL VOLUM6 (X11,Y11, FLOAT(I ), FLOAT(J), FLOAT(K ))
8890       CALL VOLUM6 (X01,Y01, FLOAT(I-1), FLOAT(J), FLOAT(K ))
8891       DXMAX = AMAX1 (ABS(X10-X00),ABS(X11-X01))
8892       DYMAX = AMAX1 (ABS(Y01-Y00),ABS(Y11-Y10))
8893       NSEGX = (DXMAX + FSEG)/FSEG
8894       NSEGX = (DYMAX + FSEG)/FSEG
8895       DII = 1.0/NSEGX
8896       DKK = 1.0/NSEGX
8897       DO 14 II = 1, NSEGX
8898         XGRID = (FLOAT(II) - 0.5)*DII
8899         DO 15 KK = 1, NSEGX
8900           ZGRID = (FLOAT(KK) - 0.5)*DKK
8901           CALL VOLUM6 (XA,YA,XGRID+FLOAT(I-1),FLOAT(J),ZGRID+FLOAT(K-1))
8902           IXA = XA/FSEG
8903           IYA = YA/FSEG
8904           IF (.NOT.T(IXA,IYA)) GO TO 15
8905 C
8906 C   FIND THE VALUE OF F AT THE POINT UNDER SCRUTINY.
8907 C
8908       FVAL = (F(I-1,J,K-1)*(1.0-XGRID)+F(I,J,K-1)*XGRID)*(1.0-ZGRID)
8909 1      + (F(I-1,J,K)*(1.0-XGRID) +F(I,J,K)*XGRID)*ZGRID
8910       IF (MODEL.GT.0.AND.FVAL.LT.CLEVEL) T(IXA,IYA) = .FALSE.
8911       IF (MODEL.LT.0.AND.FVAL.GT.CLEVEL) T(IXA,IYA) = .FALSE.
8912 15     CONTINUE
8913 14     CONTINUE
8914 C
8915 13     CONTINUE
8916 12     CONTINUE
8917 22     CONTINUE
8918 1      CONTINUE
8919       RETURN
8920 C
8921       ENTRY VOLSET (XP, YP, T, NT)
8922 C
8923 C   THIS FIRST ENTRY INITIALIZES THE PLOT GEOMETRY AND FILLS THE
8924 C   HIDDEN LINE ARRAY T WITH THE INITIAL .TRUE. FOR TRANSPARENT. THE
8925 C   ARRAY T IS NT X NT (OR 16K WORDS MAX) AND IS PASSED IN FROM THE
8926 C   OUTSIDE. THIS CORRESPONDS TO 1024/NT RASTER UNITS ON THE SC4020.
8927 C
8928 C
8929       SC1(1) = 1.0
8930       SC1(2) = 0.0
8931       SC1(3) = 1.0
8932       SC1(4) = 0.0
8933       DO 100 I = 1, NT
8934         DO 100 J = 1, NT
8935 100      T(I,J) = .TRUE.
8936       FSEG = 1024.0/FLOAT(NT)
8937       DO 103 I = 1, 8
8938       XR(I) = XP(I)

```

```
3939 103 YR(I) = YP(I)
3940     NTT = NT
3941     RETURN
3942     END
```

```

3943         SUBROUTINE WDCOUNT(W,NC)
3944 C
3945 C         COUNT THE NUMBER OF CHARACTERS (LEGITIMATE) IN A STRING
3946 C
3947         CHARACTER*1 W(*), E, P, B, H, U, D, L, C, R, O, M
3948         DATA E/'q'/,P/'.'/,B/'B'/,H/'H'/,U/'U'/,D/'D'/,L/'L'/'
3949         DATA C/'C'/,R/'R'/,O/'O'/,M/'M'/'
3950 C
3951         IP = 0 ; NO CHARACTERS
3952         I = 1
3953 100      IF(W(I).EQ.E) THEN
3954             IF(W(I+1).EQ.H) THEN
3955                 I = I + 2
3956             ELSE
3957                 IF(W(I+1).EQ.U) THEN
3958                     ELSE
3959                         IF(W(I+1).EQ.D) THEN
3960                             ELSE
3961                                 IF(W(I+1).EQ.L.OR.W(I+1).EQ.R.OR.W(I+1).EQ.M) THEN
3962                                     ELSE
3963                                         IF(W(I+1).EQ.B) THEN
3964                                             ELSE
3965                                                 IF(W(I+1).EQ.O) THEN
3966                                                     ELSE
3967                                                         IF(W(I+1).EQ.C) THEN
3968                                                             I = I + 2
3969                                                         ELSE
3970                                                             IF(W(I+1).EQ.E) THEN
3971                                                                 GO TO 200
3972                                                             ELSE
3973                                                                 IF(W(I+1).EQ.P) THEN
3974                                                                     GO TO 300
3975                                                                 ELSE
3976                                                                     GO TO 101
3977                                                                 ENDDIF
3978                                                             ENDDIF
3979                                                         ENDDIF
3980                                                     ENDDIF
3981                                                 ENDDIF
3982                                         ENDDIF
3983                                     ENDDIF
3984                                 ENDDIF
3985                             ENDDIF
3986                         ELSE
3987                             GO TO 200
3988                         ENDDIF
3989 C
3990 101      I = I + 2
3991             IF(I.GT.132) GO TO 400 ; QUIT ANYHOW
3992             GO TO 100
3993 C
3994 C         COUNT A CHARACTER
3995 C
3996 200      I = I + 1
3997             IP = IP + 1

```

```
3998          GO TO 100
3999 C
4000 C      TERMINATE
4001 C
4002 300      NC = IP
4003          RETURN
4004 C
4005 C      TERMINATE
4006 C
4007 400      NC = 0
4008          RETURN
4009          END
```

```

4010          SUBROUTINE WDDRAW(XSTART, YSTART, DX, DY, SX, SXY, SY, W)
4011 C
4012 C      XSTART REAL    - X COORDINATE (MATHEMATICAL SPACE) OF THE FIRST
4013 C                      CHARACTER TO BE DRAWN
4014 C      YSTART REAL    - Y COORDINATE (MATHEMATICAL SPACE) OF THE FIRST
4015 C                      CHARACTER TO BE DRAWN
4016 C      DX      REAL    - INCREMENT TO BE ADDED TO THE X COORDINATE FOR
4017 C                      EACH CHARACTER DRAWN
4018 C      DY      REAL    - INCREMENT TO BE ADDED TO THE Y COORDINATE FOR
4019 C                      EACH CHARACTER DRAWN
4020 C      SX      REAL    - X MATH SPACE SIZE FOR CHARACTERS
4021 C      SXY     REAL    - SLANT MODIFIER FOR CHARACTERS
4022 C      SY      REAL    - Y MATH SPACE SIZE FOR CHARACTERS
4023 C      W       - CHARACTER STRING TO BE DRAWN.
4024 C
4025 C      CONTROL CHARACTERS ¶L - TEXT IS DRAWN TO THE LEFT OF XSTART,YSTART
4026 C                      ¶M - TEXT IS CENTERED AT XSTART,YSTART
4027 C                      ¶R - TEXT IS DRAWN TO THE RIGHT OF XSTART,YSTART
4028 C
4029 C                      ¶H - CHANGE CHARACTER SETS ... ¶H01, ¶H02,
4030 C                      ... ¶H20, ¶H21 FOR CHARACTER SETS
4031 C                      1, 2, ... 20, AND 21.
4032 C
4033 C                      ¶U - DRAW THE FOLLOWING CHARACTERS IN
4034 C                      SUPERSCRIT.
4035 C                      ¶D - DRAW THE FOLLOWING CHARACTERS IN SUBSCRIPT.
4036 C                      ¶O - RESET CHARACTER SIZE AND PLACEMENT FROM
4037 C                      SUPERSCRIT OR SUBSCRIPT TO ORIGINAL
4038 C                      SIZE.
4039 C
4040 C                      ¶B - BACKSPACE OVER LAST CHARACTER DRAWN.
4041 C                      WORKS ONLY FOR ONE CHARACTER. MULTIPLE
4042 C                      BACKSPACES WILL PRODUCE UNPREDICTABLE
4043 C                      RESULTS.
4044 C
4045 C                      ¶C - CHANGE COLOR, ¶C00, ¶C01, ... TO CHANGE
4046 C                      TO COLOR 0, 1, ...
4047 C
4048 C                      ¶. - END OF CHARACTER STRING.
4049 C*****
4050 C      WDDRAW
4051 C*****
4052 C
4053 C      INTEGER CHNUMB, SDFLT
4054 C      CHARACTER*1 W(1), E, P, B, CC(4), H, U, D, L, C
4055 C      CHARACTER*1 CENTER, R, O, M
4056 C      DATA E/'¶'/,P/'.'/,B/'B'/,H/'H'/,U/'U'/,D/'D'/,L/'L'/
4057 C      DATA C/'C'/,R/'R'/,O/'O'/,M/'M'/, SDFLT/4/, SE/4/
4058 C      REAL X(132),Y(132),SCL(132),XLL(125),YLL(125),PEN(125)
4059 C      COMMON /DEV TYP/ IDEVIC,LSW,LTSW,XYCOORD(4),LUOUT,LPAGE
4060 C      COMMON/GRFTYP/ANGLE,IRVRSE,CHSIZE(9),ITKWIT,LU DIAG,LUHSET
4061 C      EQUIVALENCE (CC,CHNUMB)
4062 C      INTEGER SETL(132), SE, CH(132), ICL(132)
4063 C
4064 C

```

```

4065 C      SET THE PLOTTING SPACE PARAMETERS
4066 C
4067         CHSIZE(2) = SY * XYCOORD(3) * CHSIZE(6)
4068         CHSIZE(7) = CHSIZE(2)*(CHSIZE(3)+.5*(1.8-CHSIZE(6)))/XYCOORD(1)
4069         CHSIZE(8) = CHSIZE(2)*(CHSIZE(4)+.5*(1.8-CHSIZE(6)))/XYCOORD(3)
4070 C
4071         XL = 0.0
4072         YL = 0.0
4073         IP = 0
4074         ICOL = 0
4075         XLAST = XL
4076         WIDTH = 0.0
4077         SC = 1.0
4078         IC = 1
4079         I = 1
4080         CENTER = 'R'
4081 C
4082 C      SCAN THE INPUT STRING FOR JUSTIFICATION AND END OF TEXT
4083 C
4084 100      IF(W(I).EQ.E) THEN
4085           IF(W(I+1).EQ.H) THEN
4086             SE = CTOI(W(I+2),IC)*10 + CTOI(W(I+3),IC)
4087             IF (SE.LT.1.OR.SE.GT.24) SE = SDFLT
4088             I = I + 2
4089           ELSE
4090             IF(W(I+1).EQ.U) THEN
4091               XL = XL + SX * 0.20 * SC
4092               YL = YL + SY * 0.8 * SC
4093               SC = SC * 0.6
4094             ELSE
4095               IF(W(I+1).EQ.D) THEN
4096                 XL = XL + SX * 0.06 * SC
4097                 YL = YL - SY * 0.20 * SC
4098                 SC = SC * 0.6
4099               ELSE
4100                 IF(W(I+1).EQ.L.OR.W(I+1).EQ.R.OR.W(I+1).EQ.M) THEN
4101                   CENTER = W(I+1)
4102                 ELSE
4103                   IF(W(I+1).EQ.B) THEN
4104                     XL = XL - WLAST
4105                   ELSE
4106                     IF(W(I+1).EQ.O) THEN
4107                       YL = 0.0
4108                       SC = 1.0
4109                     ELSE
4110                       IF(W(I+1).EQ.C) THEN
4111                         ICOL = CTOI(W(I+2),IC)*10 + CTOI(W(I+3),IC)
4112                         I = I + 2
4113                       ELSE
4114                         IF(W(I+1).EQ.E) THEN
4115                           GO TO 200
4116                         ELSE
4117                           IF(W(I+1).EQ.P) THEN
4118                             GO TO 300
4119                           ELSE

```

```

4120             I = I - 1
4121             GO TO 101
4122             ENDIF
4123             ENDIF
4124             ENDIF
4125             ENDIF
4126             ENDIF
4127             ENDIF
4128             ENDIF
4129             ENDIF
4130             ENDIF
4131             ELSE
4132             GO TO 200
4133             ENDIF
4134 C
4135 101      I = I + 2
4136             IF(I.GT.132) GO TO 300 ; QUIT ANYHOW
4137             GO TO 100
4138 C
4139 C      WE HAVE A CHARACTER
4140 C
4141 200      IF(SE.EQ.1) THEN
4142             CC(1) = W(I)
4143             CWMIN = 0.
4144             CWMAX = SX
4145             ELSE
4146             CHNUMB = IDCHAR(W(I))
4147             CALL HSETS (CHNUMB,XLL,YLL,PEN,NR,SE-1,IER)
4148             IF(IER.NE.0) THEN
4149             IF(LUDIAG.GT.0) WRITE(LUDIAG,201) CHNUMB,SE,W(I)
4150 201      FORMAT(' CHARACTER ',Z10,' NOT FOUND IN SET ',I4,Z5)
4151             GO TO 203
4152             ENDIF
4153             IF(NR.EQ.0) THEN
4154             CWMIN = 0.0
4155             CWMAX = SX * 0.6
4156             ELSE
4157             CWMIN = 1.0E+10
4158             CWMAX = 0.
4159             DO 202 J = 1, NR
4160             XLL(J) = XLL(J) * 0.216 * SX
4161             CWMIN = MIN (CWMIN, XLL(J))
4162 202      CWMAX = MAX (CWMAX, XLL(J))
4163             ENDIF
4164             ENDIF
4165 C
4166 C      STORE ITS PARAMETERS IN LOCAL SPACE
4167 C
4168             XLAST = XL
4169             IP = IP + 1
4170             X(IP) = XL - CWMIN
4171             Y(IP) = YL
4172             ICL(IP) = ICOL
4173             SCL(IP) = SC
4174             CH(IP) = CHNUMB

```



```

4175         SETL(IP) = SE
4176         WLAST = (0.500*SC+0.5)*(CWMAX-CWMIN) + 0.20*SC*DX
4177         XL = XL + WLAST
4178         YL = YL + DY
4179     203     I = I + 1
4180           GO TO 100
4181     C
4182     C     DRAW IT - FIRST SET THE CORRECT CENTERING
4183     C
4184     300     XOFF = X(1)
4185           WIDTH = XL - XOFF
4186           IF (CENTER.EQ.R) THEN
4187             DXX = 0.0
4188           ELSE IF (CENTER.EQ.M) THEN
4189             DXX = WIDTH / 2.
4190           ELSE IF (CENTER.EQ.L) THEN
4191             DXX = WIDTH
4192           END IF
4193           DO 301 I = 1, IP
4194     301     X(I) = X(I) - XOFF - DXX
4195     C
4196     C     DO ANY SHIFTING AND NECESSARY ROTATIONS
4197     C
4198           COSX = COS(ANGLE)
4199           SINX = SIN(ANGLE)
4200           SXX = ABS(XYCOORD(1))
4201           SYX = ABS(XYCOORD(3))
4202           OSX = 1. / SXX
4203           OSY = 1. / SYX
4204           DO 304 I = 1, IP
4205             XI = X(I) * COSX * SXX - Y(I) * SINX * SYX
4206             YI = X(I) * SINX * SXX + Y(I) * COSX * SYX
4207             X(I) = XI * OSX + XSTART
4208     304     Y(I) = YI * OSY + YSTART
4209     C
4210     C     NOW DO THE DRAW
4211     C
4212           IF(IP.LE.0) GO TO 303 ; NOTHING TO DO
4213           ICOL = 0
4214           DO 305 J = 1, IP
4215             IF(ICL(J).NE.0.AND.ICL(J).NE.ICOL) THEN
4216               CALL COLOR(ICL(J))
4217               ICOL = ICL(J)
4218             ENDIF
4219     305     CALL HHDRAW (X(J), Y(J), SX*SCL(J), SXY, SY*SCL(J),
4220             CH(J), SETL(J), IER)
4221     303     CHSIZE(2) = CHSIZE(1)
4222           ANGLE = 0.0
4223           RETURN
4224     C
4225     C     CHANGE THE DEFAULT CHARACTER SET - BYPASS THE ESCAPE SEQUENCE
4226     C
4227           ENTRY CHRSET (ISET)
4228           IF(ISET.GT.0) SE = ISET
4229           IF(LUDIAG.GT.0) WRITE(LUDIAG,302) ISET

```

```
4230      RETURN
4231 302    FORMAT(' CHANGE THE DEFAULT CHARACTER SET TO #',I3)
4232      RETURN
4233      END
```

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET <i>(See instructions)</i>	1. PUBLICATION OR REPORT NO. NBSIR 85-3235	2. Performing Organ. Report No.	3. Publication Date October 1985
4. TITLE AND SUBTITLE <p style="text-align: center;">A Device Independent Graphics Kernel</p>			
5. AUTHOR(S) Walter W. Jones and Alicia B. Fadell			
6. PERFORMING ORGANIZATION <i>(If joint or other than NBS, see instructions)</i> NATIONAL BUREAU OF STANDARDS DEPARTMENT OF COMMERCE WASHINGTON, D.C. 20234 Gaithersburg, Maryland 20899		7. Contract/Grant No.	8. Type of Report & Period Covered
9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS <i>(Street, City, State, ZIP)</i>			
10. SUPPLEMENTARY NOTES <input type="checkbox"/> Document describes a computer program; SF-185, FIPS Software Summary, is attached.			
11. ABSTRACT <i>(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)</i> <p>This paper describes an interface for programs which allows one to write graphics primitives to several devices without regard for the type of device. The most salient features are that it has low overhead, is transportable and can be expanded as the nature of the input/output devices changes. A conscious effort has been made to include all normal graphics primitives together with the most useful high level routines without compromising the use of special features of custom display units.</p>			
12. KEY WORDS <i>(Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)</i> device independence; display devices; graphics			
13. AVAILABILITY <input checked="" type="checkbox"/> Unlimited <input type="checkbox"/> For Official Distribution. Do Not Release to NTIS <input type="checkbox"/> Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402. <input checked="" type="checkbox"/> Order From National Technical Information Service (NTIS), Springfield, VA. 22161		14. NO. OF PRINTED PAGES <p style="text-align: center;">241</p>	15. Price



