AEROPHYSICS RESEARCH
CORPORATION
TECHNICAL NOTE

JTN-07

(NASA-CR-141593) PLOTTER: AN INDEPENDENT COMPUTER PROGRAM FOR THE GENERATION OF GRAPHICAL DISPLAYS Technical Report, Jun. 1973 - Dec. 1974 (Aerophysics Research Corp., Houston Tex.)

CSCL 09B G3/61

N75-17119

Unclas
09634

PLOTTR: AN INDEPENDENT COMPUTER PROGRAM FOR THE GENERATION OF GRAPHICAL DISPLAYS.

by: C. R. Glatt and G. N. Hirsch

Prepared for:
National Aeronautics and Space Administration
Houston, Texas 77058

November 1974

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-		

4. Title and Subtitle	5. Report Date
PLOTTR: AN INDEPENDENT COMPUTER PROGRAM FOR THE GENERATION OF GRAPHICAL DISPLAYS.	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
C. R. Glatt and G. N. Hirsch	JTN-07
	10. Work Unit No.

9. Performing Organization Name and Address	11. Contract or Grant No.
Aerophysics Research Corporation 18100 Nassau Bay Drive, #47 Houston, Texas 77058	NAS9-13584
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Contractor Report
National Aeronautics and Space Administration Johnson Space Center	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

A computer program is described for generating graphical information from input data or auxiliary analysis programs on a variety of graphical devices. Information can be presented in the usual X-Y plot sense or can be presented as contour plots.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Plotting, graphical analysis, computer aided design	Unclassified - Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified		

Preface

This report describes an Independent Computer Program for the Generation of Graphical Displays. The program was written for inclusion in the Engineering Design Integration (EDIN) System Library in support of NASA Contract NAS9-13584, "Extended Optimal Design Integration (Extended ODIN) Computer Program." The study was conducted during the period from June 1973 through December 1974 with funds provided by the National Aeronautics and Space Administration, Johnson Spacecraft Center, Engineering Analysis Division. The contract was monitored by the Launch Analysis Section.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

PLOTTR: AN INDEPENDENT COMPUTER PROGRAM FOR THE GENERATION OF GRAPHICAL DISPLAYS.

By: C. R. Glatt

SUMMARY

A computer program is described for generating graphical information from input data or auxiliary analysis programs on a variety of graphical devices. Information can be presented in the usual X-Y plot sense or can be presented as contour plots. The user selects the segments of data to be plotted as well as the scaling, annotation and titling of the resulting plot. Options also exist for tabulating the data in columnar format and for plotting auxiliary text in the vicinity of the plotted information. Display device selection is accomplished by interfacing the basic computer code through routines which convert the internally generated plot vectors to hardware commands for the display device. Separate computer programs are available for the following devices.

Tektronix 4012 Display Terminal

Hazeltine 4000G Terminal (MOPS System)

CALCOMP Drum Plotter

SD4060 Film Plotter

The plotting techniques employed in the computer program are discussed in detail. User's instructions are presented with examples which illustrate the use of the program in generating plotted information from various sources and presenting the information in alternate plot formats. Technical discussion of the computer code is presented giving the physical characteristics, computer loading instructions and descriptions of the subroutines.

INTRODUCTION

The display of engineering information pertaining to a vehicle design is one of the most important aids in the design analysis process. Historically, design analysis data has been generated on the computer and returned to the engineer in a printed format. The data was normally transformed to a more usable graphical presentation through hand plotting by the engineer or engineering aide. More recently, hardware techniques have been developed which permit automatic plotting of preselected analysis data. The programmer normally provides precoded logic in the technology module to generate plot vector files. The plot vector files are then processed by the hardware plotting device and the graphical display generated is returned to the engineer. The classical approach described above requires that the plot vector file generation software be imbedded in every technology program from which analysis information is anticipated. The analysis plots are generally rigidly coded and the engineer has little choice in parameter selection. Furthermore, the comparison of current data with previously generated data (or data generated by other technology modules) is severely limited.

The development of the multiple program technological analysis techniques of references 1 and 2 has significantly increased the requirements for the presentation of analysis data of an interdisciplinary nature. The use of plot vector generation code within the technology modules does not meet the interdisciplinary or the comparison requirements. The objective in the development of a plot program, which is independent of the technology programs, is to provide a means of plotting any data which was previously generated by a technology module and stored in a data base.

The plot program specifications require the generation of X-Y plots, cross plots and contour plots which compare not only data from a specific analysis or any individual programs but also to provide comparisons with previously generated data from any source. The independent plot program developed meets the desired specifications or objectives. It assumes no prior knowledge of the information that is to be plotted at the time of execution. Plot scales, annotations, titles, etc. are described by the user through input. Plot data requests are specified by the user and the data can come from many sources.

The independent plot program was an evolutionary development based on requirements which were developed through use of the multiple technology analysis technique. It has served well during the period of development and seems to be adequate for most technological data sources.

TECHNICAL DISCUSSION

The independent plot program provides for the generation of x-y plots and contour plots on a number of hardware display devices. Auxiliary plot text and tabulations of plot data can also be obtained from the program.

The plot instructions are read from the input data file and the data to be plotted can be read from either the input file or from auxiliary data files. The input instructions are read using a single NAMELIST input list.

```
$PLOTIN.....$
```

Default values are preset for all input variables. The default values tend to minimize the amount of user input required to generate a plot. Generally the input specifications are patterned after the procedure one might follow in preparing plot by hand. The user can select such options as grid, axis generation, annotation, titles, auxiliary text, line type, symbol specifications, etc. Those options not specifically selected by the user are generally bypassed in the program.

A simple self tutoring procedure for a new user can be described as simply to scan the input specifications in a later section and include values for the input variables which require changing. A discussion of the plotting options which are available is presented in the following paragraphs.

Plot Data Input Option

The data to be plotted is read into the program in either of two formats, array format or observation format. The data may come from the normal input or from a binary file in accordance with the following specifications:

INPUT = 0	Data will be loaded directly into the OBSTH array from the normal input unit.
INPUT = n	n Specifies the logical unit number from which the OBSTH array will be loaded.

Array Format. - Array format specifies the numerical values to be plotted are arranged in groups of similar observations such as time, attitude or velocity. Mathematically, array format is the sequence of numerical values:

$$OBSTH_{ij};\ i = 1,\ NUMP;\ j = 1,\ NOBSER$$

where NUMP is the number of observations and NOBSER is the number of observation functions. The plot data is actually a single dimensional array $OBSTH_k$ where k is the array element defined as:

$$k = (j - 1)\ NUMP + 1$$

The independent plot program normally reads data from the input file in array format but other options are also available.

Observation Format. - Observation format specifies the numerical values are arranged in groups called observations. Each observation represents a sequence of values defining one element in each plot array. Observation format may be expressed mathematically as:

$$OBSTH_{k(i,j)};\ i = 1,\ NOBSER;\ j = 1,\ NUMP$$

where NOBSER is the number of observation functions and NUMP is the number of observations. k Is the array element defined as:

$$k(i,j) = (j - 1)\ NOBSER + 1$$

The independent plot program has the capability of reading data as observation format when the alternate value variable:

ALTVAL = .TRUE.

option is specified. If specified, a transposition of the observation format to array format is performed and the array format ultimately overwrites the input values of OBSTH.

Alternate Data File. - In addition to the two formats which can be selected for reading from input, an alternate data file can be specified:

INPUT = n

When this option is specified, the plot data is read from the logical unit n in observation format, transposed and placed into the OBSTH array in accordance with the specified values of NUMP on NOBSER. No file positioning is done by the program but the user can manipulate the file with the following input variables:

REWIND = .TRUE.	Rewinds n.
NSKIPF = m	Skips forward m Fortran files.
NSKIPR = m	Skips forward m Fortran records.

The alternate file format may include as the first record one word specifying the number of records on the file. If the one word record exists, the plot program can read it and store the value as NUMP. This option is activated by the specification:

NUMP15 = .TRUE.

The CDC 6600 has a Fortran callable binary blocking feature. If the alternate file was generated as a "binary blocked" file, the variable:

BLOCK = .TRUE.

must be set to read the file properly. The above input has no affect on the Univac version.

Plot Output Control

The output type specification can be x-y plots, contour plots, plot text or a printed tabulation in accordance with the options illustrated in figure 1. Some combinations of options are permissible but others are not. Contour plots and x-y plots are generated in different sections of the program and therefore, are mutually exclusive options. Multiple cases may be executed for as many plot cases as desired. Therefore, different options may be specified on successive cases. On the last case the parameter:

STOP = .TRUE.

is set which causes program termination. No other plot instructions are executed in the last case.

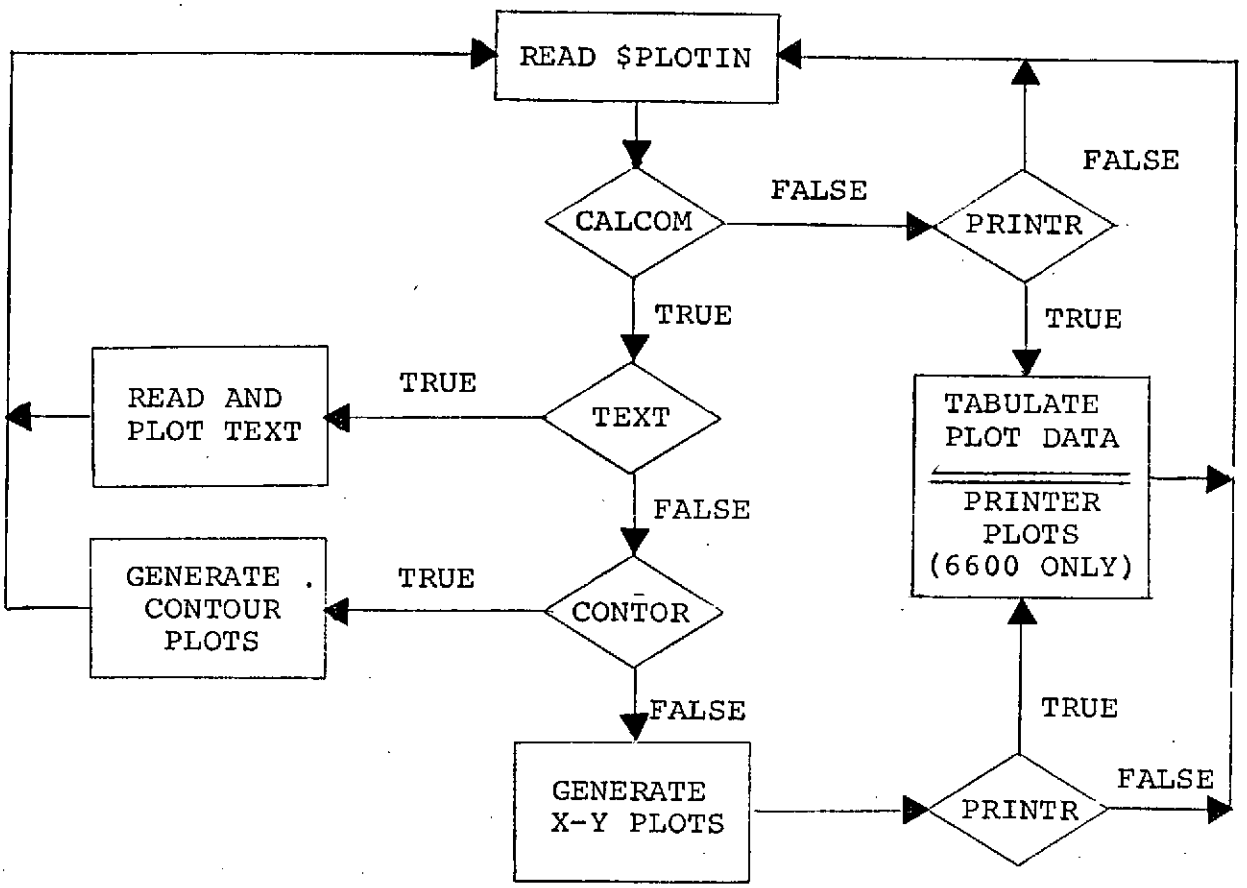
X-Y Plots

The PLOTTR program provides for the display of numerical information of the form:

$$Y_i = f_j(X_i); \; i = 1, \text{NUMP}; \; j = 1, \text{NPLOT}$$

where X and Y are arrays of observations of known length, NUMP. Any array may be plotted with respect to any other array and any number of pairs may be presented on a single plot. NPLOT is the number of PLOTS desired. Figure 2 illustrates the display of two multiple curve plots. The input data for this plot is discussed in a later section.

Plotting is specified in accordance with the input variable array:



(A) PROGRAM SCHEMATIC.

DESIRED OUTPUT	INPUT VARIABLE OPTION			
	CALCOM	CONTOR	TEXT	PRINTR
X-Y PLOTS	TRUE	FALSE	FALSE	T or F
CONTOUR PLOTS	T or F	TRUE	FALSE	T or F
PLOT TEXT	TRUE	FALSE	TRUE	NA
TABULATION	T or F	T or F	FALSE	TRUE

(B) OPTION SPECIFICATION.

FIGURE 1 PLOT OPTIONS.

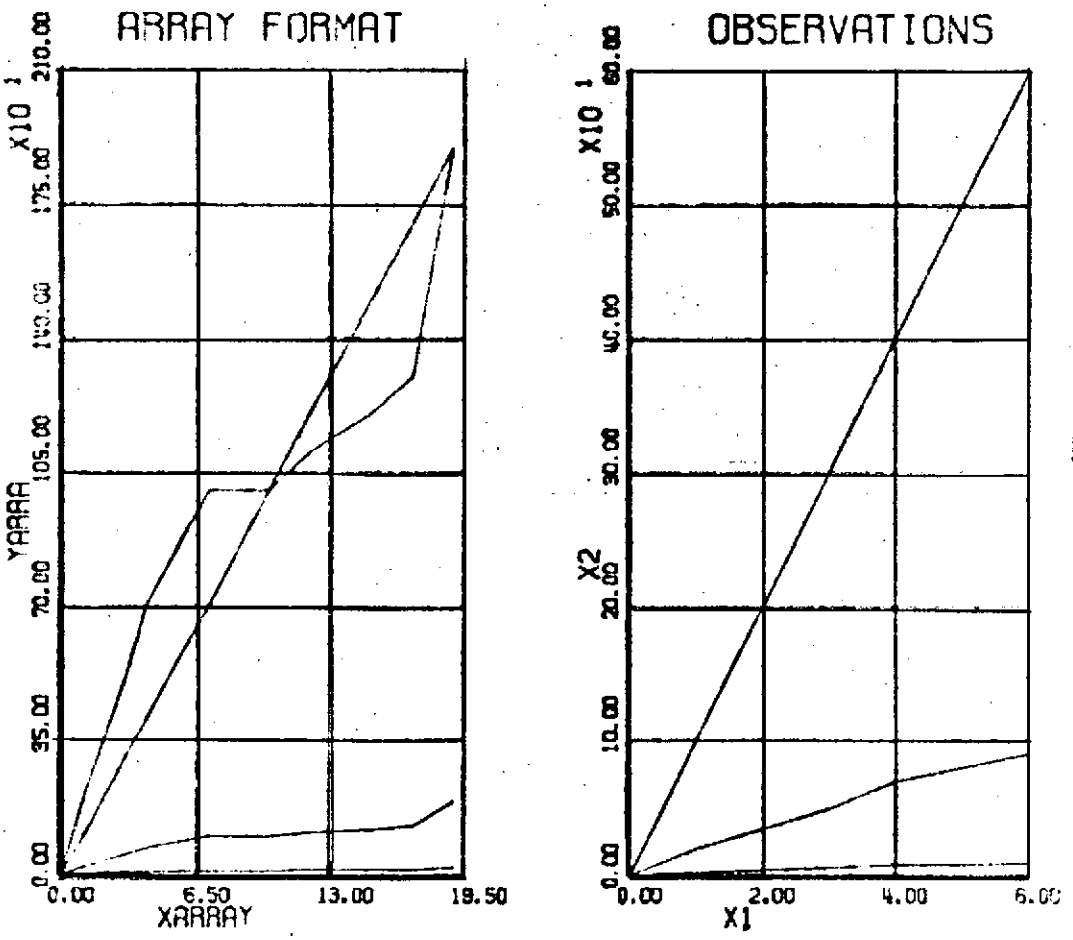


FIGURE 2 ILLUSTRATION OF X-Y PLOT.

OBSPLT = K_1, K_2 K_3---K_n

Each K is a packed integer which defines the array pair in the OBSTH array to be plotted:

K_i = XXYY

where XX represents a two-digit sequence defining the array to be plotted as the x-coordinate. YY represents a two-digit integer sequence defining the y-coordinate of the plotted line. Any number of K values may be specified, each representing a line on the plot. A plot sequence is terminated by a value:

$K_i = 0$

A second plot sequence may be specified by simply adding values of K:

K_{i+1} = XXYY

The last value of K should be:

$K_n = 7777$

which terminates the plotting and returns program control for new input data.

Plot Positioning. - The PLOTTR program has input parameters which control the defining of a new frame and the position of the plot within the frame. The parameters:

DXG and DYG

define the X- and Y- coordinates of the lower left corner of the plot in inches with respect to the lower left corner of the current frame. The parameters:

XMOVE and YMOVE

define the coordinates of the lower left corner of the next frame with respect to the lower left corner of the current frame. The frame includes all physical plotting which takes place as a result of a single set of plot instructions (defined by $PLOTIN). Frames of data may be superimposed to accomplish certain analysis objectives.

Line Type and Symbols. - The parameters, LINTYP control the type of lines that are to be drawn between the data points. The magnitude determines the frequency of symbols and the sign determines the combination of lines and symbols.

LINTYP = n Means a symbol is to be drawn every nth point.

If n > 0, Lines and symbols are drawn.

If n < 0, Only symbols are drawn.

The parameter INTEQ determines which symbol is to be used by the specification of an integer from 1 to 22. A value of 0 specifies no symbols.

Data Scaling. - Data arrays can be scaled absolutely in terms of the axis length on which they are plotted or the arrays can be scaled relative to one another. The parameters:

XSIZE and YSIZE

specify the x- and y- axis lengths in inches. The first curve specified (see OBSPLT) determines the scale factor and the relative starting position for all curves on the plot. The array:

$SCALEF_i$

specifies individual scale factors for each plot array. This allows meaningful comparisons of multiple curve plots where the range of the data array differs significantly.

Elimination of automatic scaling may be specified in either or both directions by the specification:

MYX = .TRUE.
and/or
MYY = .TRUE.

If the MYX option is specified, the user must specify the scale factor and starting value the X-axis:

SCALEX = units/inch

STARTX = inches

If the MYY option is specified, the user must specify the scale factor and starting value of the Y-axis.

SCALEY = units/inch

STARTY = inches

The scale factors and start values are set by the program each time automatic scaling (MYX and MYY = .FALSE.) is specified so the scaling from previous cases may be used by properly setting the option flags MYX and MYY.

Scale Annotation. - Under the automatic scaling option the program generates axes of the specified length with tick marks at one inch intervals. The tick marks are identified by numerical values below or to the left of the axis centered on the tick. The axes may be notated additionally by a 6-character input name centered on the axis. One name may be input for each observation function as follows:

$OBSERV_i = N_i$, i = 1, NOBSER

where N_i is a hollerith name of the form, nH name, and NOBSER is the number of observation functions.

Grid Generation. - A grid may be generated which represents vertical and horizontal lines at the tick marks by the input parameter:

GRID = .TRUE.

Title Generation. - A title from the plot may be generated by setting the parameter:

NREM = n

where n is the number of words of the title. The program will use the first n words of the REM array.

$REM = N_i$; i = 1, NREM

and place this title at the top of the plot. The height of the characters is in the title as specified by the parameter REMSIZ in inches.

Auxiliary Plot Text

The plotter program has the capability of generating auxiliary plot text by a separate case setup as follows:

```
$PLOTIN     TEXT = .TRUE.,$
 (text cards)
```

The text cards immediately follow the case data. The character strings given on the text cards are reproduced exactly starting the first card at a location specified by:

DXG and DYG

in inches. The character height specification is given by HTEXT. Cards will be read and the characters plotted with line spacing of:

1.5 * HTEXT

until a card is encountered with a numeric 2 in column 1. Control is then returned for a new case.

Virtual and Display Window

Virtual and display graphics deals with the translation of the user's data to a physical location on the display device. The virtual space may be of any dimension while the display space is limited by the physical size of the display device. The function of the virtual graphics package is to map the virtual space into the specified display area. With an understanding of the relationship between the virtual space and the display area, the user can freely manipulate the display to reflect his need. For example, he can plot three different sets of data in the same display area or he can display the same data to different scales to meet special needs.

The Virtual Window. - The user defines the scale (SCAL) of the virtual window in inches and the position (DXG and DYG) in inches with respect to the plot device origin. Graphic lines (vectors) and portions of lines which lie outside the virtual window are automatically eliminated or clipped by the windowing routines, while those which lie within or pass through the window are scaled and fitted into the appropriate portion of the display.

The user's data area may be conceived of as existing virtually within the computer, and is analogous to a coordinate system with infinite dimension. The unit of measurement in virtual space is arbitrary and representative of any unit the user wishes, from milligrams to light years. In the case of the PLOTTR program, the virtual window is established by XSIZE and YSIZE and includes the plot area itself plus some margin on all sides for plot annotation. In actuality, the virtual window is a square area whose sides are related to the larger of XSIZE or YSIZE. In effect, the user constructs the desired plot in virtual space and the virtual graphics package scales the data to fit into the

desired display area. Figure 3 illustrates the use of virtual graphics. The heavy outside border is the window which is scaled to the specified display size SCAL.

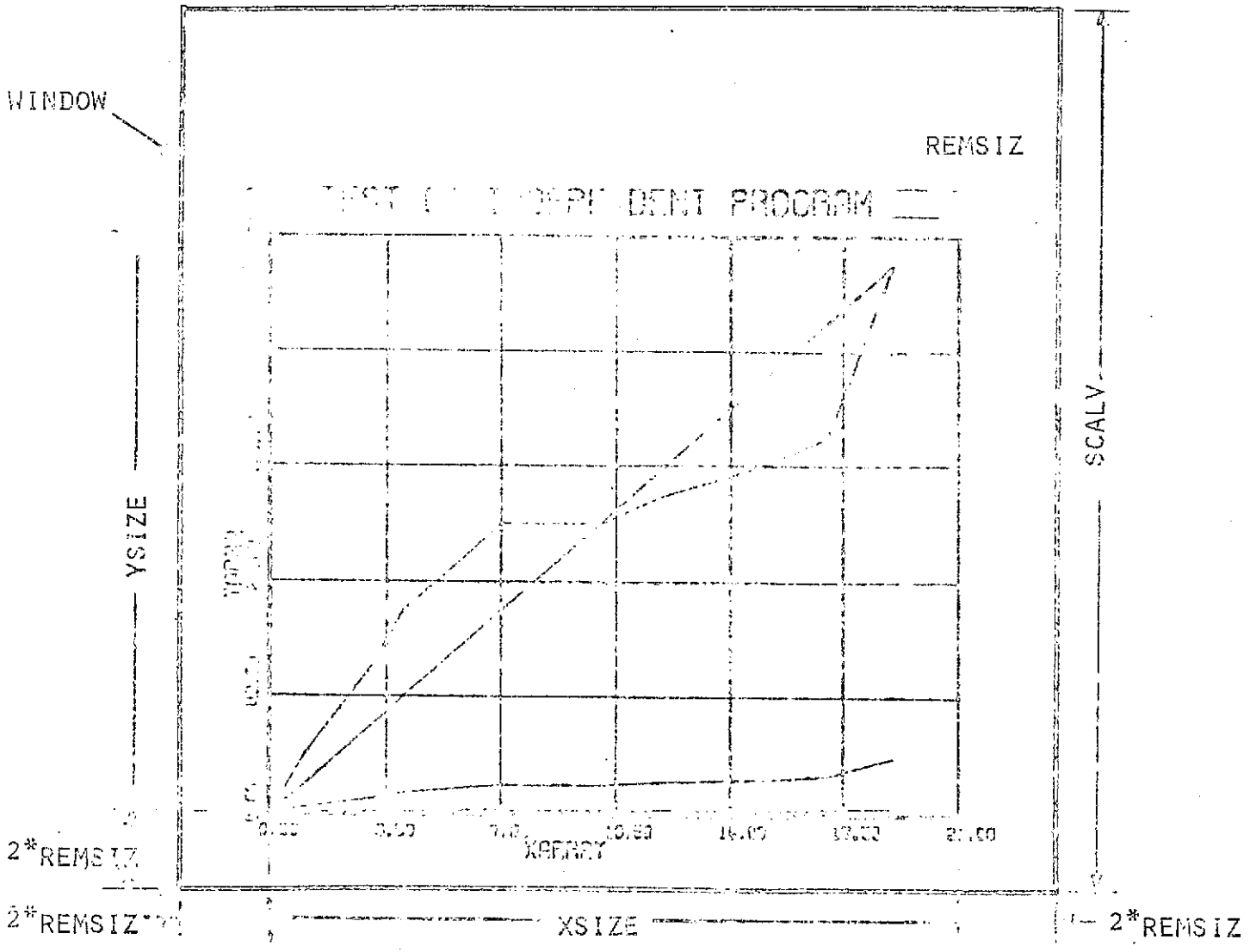


FIGURE 3 ILLUSTRATION OF VIRTUAL GRAPHICS.

Contour Plots

PLOTTR contains a simple and rapid code for producing contour plots of three dimensional functions. The function must be specified in the form:

$$Z_{ij} = F(X_i, Y_j);\ i = 1,2,..M,\ j = 1,2,..N \qquad (1)$$

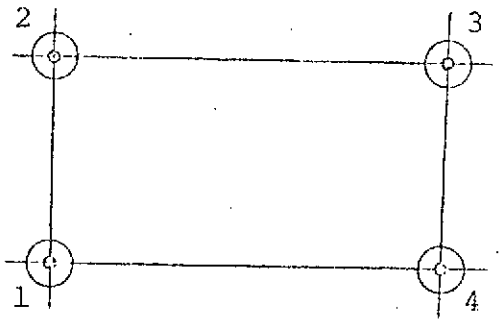
That is, the function must be defined over a regular grid in the X-Y plane. Contour plots are produced on the target display device but cannot be generated as printer plots. The resolution of the contours is dependent upon the mesh increments in X and Y.

Figure 4 illustrates the type of results which can be expected from the CONTOR option. The illustrated contour data has a "resolution" of 0.1 in X and Y.

Generation of Contour Vectors. - Suppose a function of two independent variables is defined over the regular mesh X_i, Y_j. Then (M-1)*(N-1) rectangular boxes can be selected from the adjacent points:

$$P = (X_i, Y_j),\ (X_i, Y_{j+1}),\ (X_{i+1}, Y_{j+1}),\ (X_{i+1}, Y_j) \qquad (2)$$

Let the corners of any such box be identified in a clockwise manner as illustrated below:



RECTANGULAR ELEMENT CORNER IDENTIFICATION.

Given such a rectangle, the corner points and the function values at these corner points may also be uniquely defined by the rotation shown below:

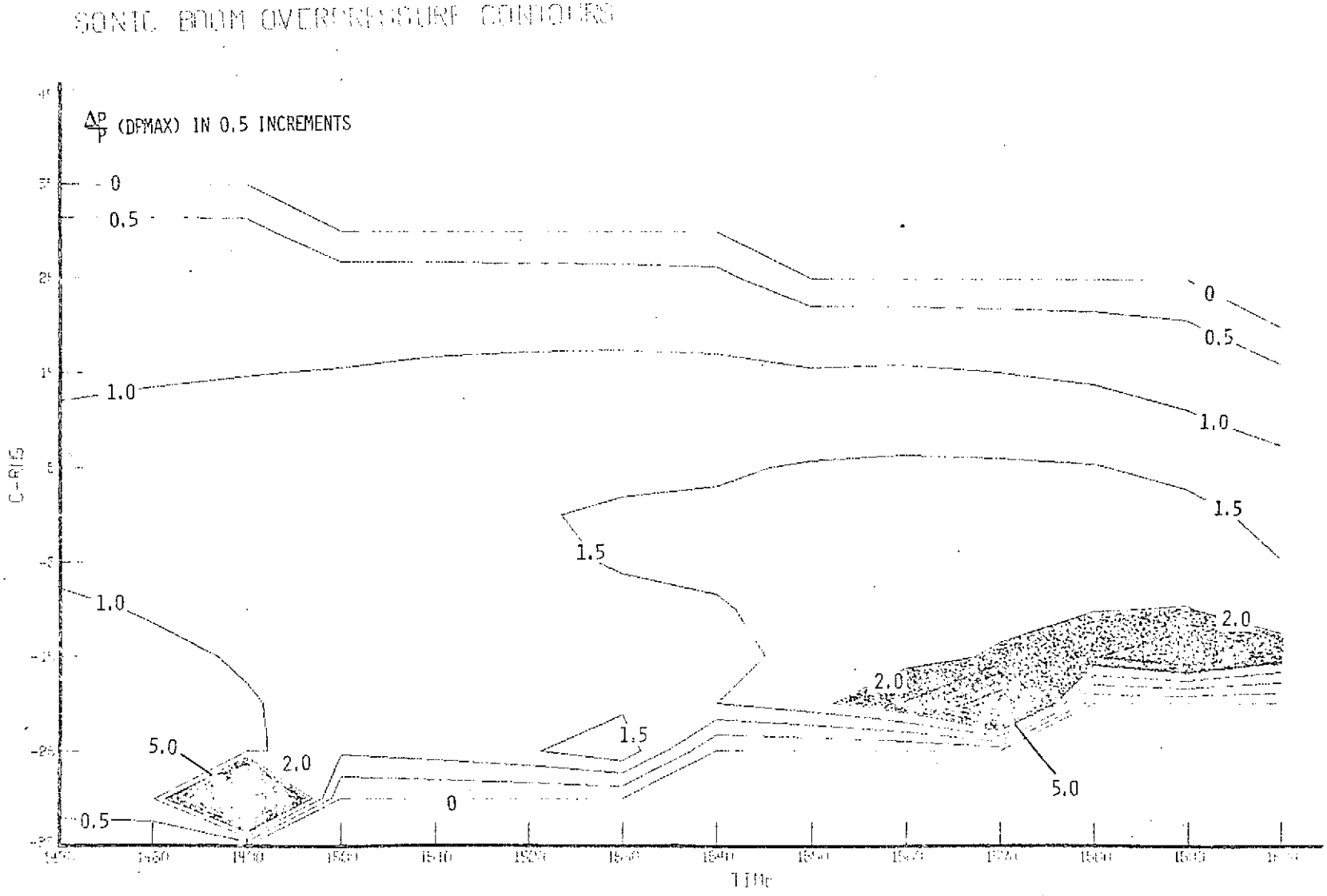
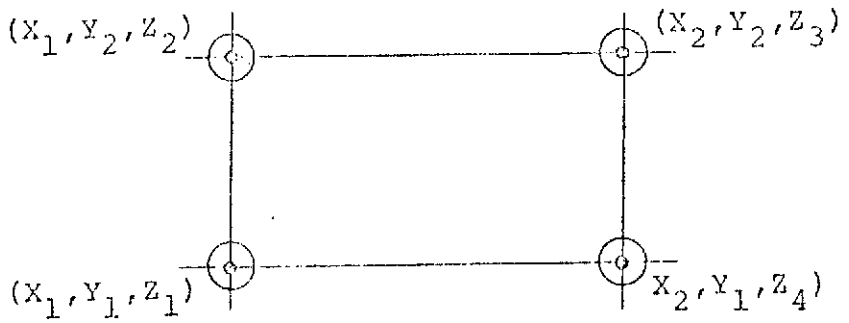


FIGURE 4 ILLUSTRATION OF CONTOUR PLOT DATA OUTPUT.



LOCAL COORDINATE AND FUNCTION VALUES

Now consider the possibility that a contour of value Z = Z' passes through a given elemental rectangle. First, transform the corner values of Z by subtracting Z' to give the modified corner values:

$$\overline{Z}_1 = Z_1 - Z'$$

$$\overline{Z}_2 = Z_2 - Z' \qquad (3)$$

$$\overline{Z}_3 = Z_3 - Z'$$

$$\overline{Z}_4 = Z_4 - Z'$$

Examining the topology of the contour line trace accross the elemental rectangle, it follows that sixteen (16) possible types of contour trace exist. For each modified corner value given by three (3) is either greater than or equal to 0, or less than zero, giving 2^4 topological types of trace. These trace types may be uniquely identified by a four digit binary number whose elements sequentially correspond to the corner points in the clockwise sequence of figure 5, where 1 signifies that $\overline{Z}_i > 0$. These sixteen types of trace are illustrated in figure 5 with their corresponding four (4) digit binary number and contour trace type.

It can be seen that only two (2) of the element topologies result in no contour trace. Two element topologies result in two contour traces across the element. (These contour traces may be of either form displayed in figure 5 for I = 1010 or I = 101.) Assuming linear interpolation for $\overline{Z}$ along the elemental rectangle sides and local straight line contour traces within the element, the end points of the contour trace are readily found to be given by the expressions such as:

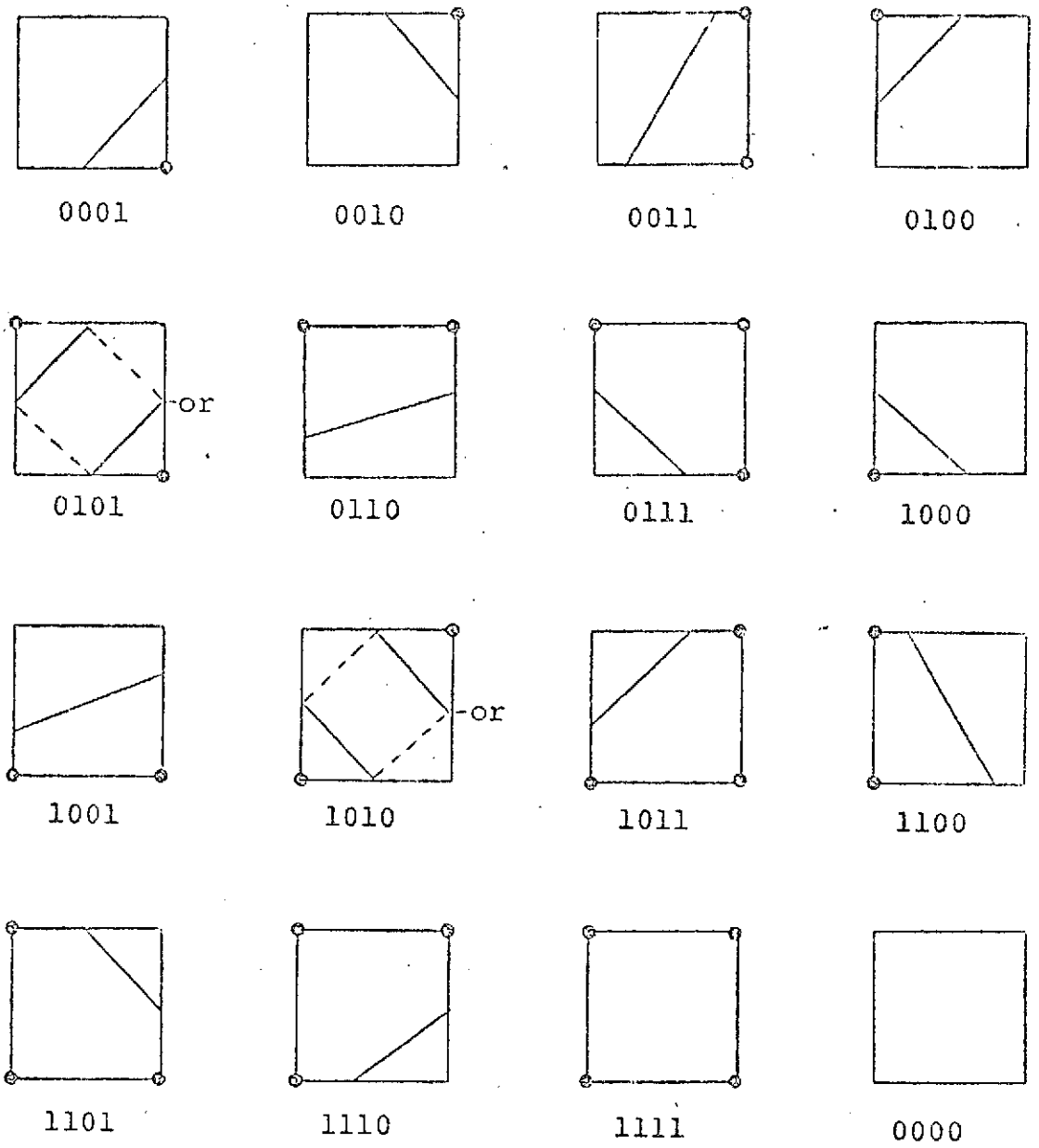


FIGURE 5 RECTANGULAR ELEMENT BINARY CODE AND CONTOUR TRACE TOPOLOGIES.

$$\underline{I = 1010}$$

$$X_{e_1} = X_1$$

$$Y_{e_1} = Y_1 + F(Y_1, Y_2, Z_1, Z_2) \qquad (4)$$

$$X_{e_2} = X_1 + F(X_1, X_2, Z_3, Z_4)$$

$$Y_{e_2} = Y_2$$

and

$$\underline{I = 1110}$$

$$X_{e_1} = X_1 + F(X_1, X_2, Z_1, Z_4)$$

$$Y_{e_1} = Y_1 \qquad (5)$$

$$X_{e_2} = X_2$$

$$Y_{e_2} = Y_1 + F(Y_1, Y_2, Z_4, Z_3)$$

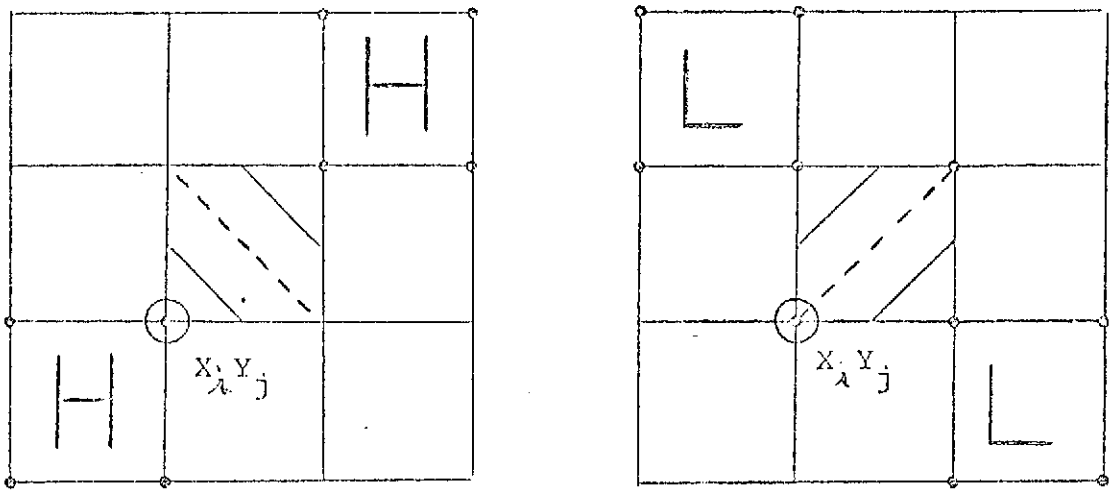
Where (X_{e_1}, Y_{e_1}) and (X_{e_2}, Y_{e_2}) are the contour trace end points and the function F is defined by:

$$F(A,B,C,D) = (B - A) \cdot \left| \frac{C}{D - C} \right|$$

Two Ambiguous Cases. - Two ambiguous cases arise for I = 1010 and I = 101. The contours may then be of either the solid or dotted line type in figure 6. The ambiguity is resolved when the "fold diagonal" is defined. Thus when I = 0101, if the principal diagonal is the fold diagonal, the dotted lines supply the correct contour types, if the other diagonal is the fold line, then the solid lines are the correct contour types. The converse is true when I = 1010.

Definition of the fold line requires more information than is available within a single elemental rectangle. Consider the case I = 1010 in the illustration below. In (a) the upper right and

lower left high elemental rectangle regions indicate a principal diagonal fold. In (b) the lower upper left and lower right elemental rectangle regions illustrate a fold about the other diagonal.



(a) PRINCIPAL DIAGONAL FOLD

(b) OTHER DIAGONAL FOLD

FIGURE 6 TWO ANALOGOUS CASES.

The program contains logic for resolving the ambiguous cases using the above technique. A weighted assessment of the probability of each diagonal by the fold line is incorporated within the code. This logic covers the possibility of any adjacent elemental rectangle being absent due to edge or corner conditions in the Z_{ij} array.

Loading the Data. - The contour data is loaded into the OBSTH array as follows:

$$OBSTH_k;\ k = 1,\ NUMP * NOBSER$$

where k is defined as:

$$k = (j - 1)\ NUMP + i$$

$$i = 1,\ NUMP$$

$$j = 1,\ NOBSER$$

Data may also be loaded from an alternate input file using the INPUT parameter as described earlier. Two additional arrays defining the x and y mesh parts must be loaded as:

$XMESH_i$; i = 1, NUMP

$YMESH_j$; j = 1, NOBSER

The values must correspond to the function values loaded in the OBSTH array.

Contour Definition. - The contours desired are specified by the user using the following input:

NZCUTS = n.

where n is the desired number of contours. The array:

$ZCUTS_i$; i = 1, NZCUTS

defines the individual contour values.

Title and Axis. - Title and axis options are the same as described earlier for x-y plots.

PROGRAM USAGE

The program can be used with two online and two offline display devices as follows:

DISPLAY DEVICE	TYPE	PROGRAM NAME
TEKTRONIXS	Online	*TEKTRON.OPLOT
MOPS	Online	*HAZEL.OPLOT
CALCOMP	Offline	*CALCOMP.OPLOT
SD4060	Offline	*SD4060.OPLOT

*File qualifier is EX42-00002.

Each device uses a different program. The data setup is identical so far as parameter names are concerned. However, the desired scaling value may be different because of the size of the available plotting area for the different devices.

Each plot interface has special instructions that are particular to the plot device. For example, the CALCOMP program requires the mounting of a physical tape which must be named 19:

```
@ASG,T 19,8C, CAL
@XQT EX42-00002*CALCOMP.OPLOT
 (Data)
```

The TEKTRONIX program requires the user to input the terminal line speed and the maximum number of vectors per buffer load:

```
@XQT EX42-00002*TEKTRON.OPLOT
 30 18
 (Data)
```

The above example is for 30 characters per second line speed and 18 vectors per buffer load. No more than 18 vectors should be specified.

The MOPS program requires the use of the function code 0 above the keyboard before data is entered:

```
@XQT EX42-00002*HAZEL.OPLOT
   Depress FC-0
 (Data)
```

No special input is required to use the SD4060 program:

Program Input

The basic input to PLOTTR is the $PLOTIN namelist specification. All data except auxiliary text may be read in this form if desired; but other forms of input may be activated in accordance with the options specified in $PLOTIN.

The use of NAMELIST input was chosen for the following reasons:

1. It is a simple name oriented input easily understood by most computer users.

2. The format is standard and does not require relearning from program to program.

3. It is easily modified by the engineer or programmer when adding input variables to the program.

When a NAMELIST read is encountered in the program, the entire input file is scanned up to an end-of-file or a record with a dollar ($) in column 2 followed by the NAMELIST name requested by the program. No imbedded blanks may be used. Succeeding data items are read until a second dollar is encountered signifying the end of the NAMELIST input. Any data on the input file before the requested NAMELIST is found will be ignored. All data between the opening and closing dollar is interpreted by the NAMELIST input routine. The data item within the NAMELIST statement may be in either of two forms:

$$V = C,$$

$$A(n) = D_1,\ldots,D_m,$$

V is a non-subscripted variable name. C is a constant which may be real, integer, logical (T or F) or hollerith (ℓHname). A is an array name and n is an integer constant subscript, $D_1,\ldots,D_m$, are simple constants or repeated constants of the form k*C, where k is the repetition factor. Data items and constants must be separated by commas. For a subscripted array name, the number of constants need not be equal but may not exceed the number of array elements needed to fill the array. More than one card may be used for input data and arrays may be split between cards. All except the last record must end with a constant followed by a comma and no sequence numbers may appear. The first column of each record is ignored. The

set of data items may consist of any subset of the variable names associated with the NAMELIST name and the name need not be any particular order. More details on the use of NAMELIST are available in any FORTRAN users guide, but the above description should be sufficient for the operation of the PLOTTR program.

$PLOTIN Namelist Data. - This section contains an alphabetical list of all input variables in the $PLOTIN namelist set. The default value and a brief description of the usage of each variable is given.

Name	Default(s)	Description of Input
ALTVAL	.FALSE.	Logical variable. If .True. data will be read as observation functions. Otherwise data will be read as plot arrays.
BLOCK	.FALSE.	Logical variable. If .True., binary blocking of the observation unit will be expected. (CDC 6600 only)
CALCOM	.TRUE.	Logical variable. If .True., x-y plots will be generated. This variable is used to activate any device to which the program is linked such as Tektronics, SD4060 or Varian.
CONTOR	.FALSE.	Logical variable. If .True., a contour plot will be generated from OBSTH data CALCOM must be set .True.
COPY	.FALSE.	Generate hard copy (online devices).
DIALOG	.FALSE.	Logical variable. If .True., data base output routine will be called.
DXG	1.0	X-axis origin for the current chart relative to the current device origin specified by XMOVE.
DYG	1.0	Y-axis origin for the current chart relative to the current device origin specified by YMOVE.

Name	Default(s)	Description of Input
GRID	.FALSE	Logical variable. If .TRUE., one inch grid will be generated on x-y plots.
HTEXT	.21	Height of title in inches.
INPUT	0	Zero for reading plot data from cards .GT.0 for reading plot data from another unit.
INTEQ	1	Integer from 0 to 22 indicating the plot symbol.
LINTYP	0	Control parameter which describes the type of line to be drawn through the data points. The magnitude determines the frequency of plotted symbols. I.E. LINTYP=4 Means every fourth point. LINTYP=0 Straight lines with specified symbol at the end of the line (see INTEQ). LINTYP=+ Lines and symbols. LINTYP=- Symbols only.
MYX	.FALSE.	Logical variable. If .TRUE., user may input STARTX and SCALEX.
MYY	.FALSE.	Logical variable. If .TRUE., user may input STARTY and SCALEY.
NAMES (100)	.FALSE.	Logical variable. If .TRUE., NOBSER six-character names will be read. These are the plot array titles.
NAXIS	1	Number of times the beam or pen will trace the x- and y- axes.
NDECP	2	Number of decimal places used in scale annotation.
NOBSER	NONE	Number of observation functions (or plot arrays).
NPAGE	0	Page number of the plot (printer only).
NREM	0	Number of words of title to be read. If not zero, NREM 10-character words will be read immediately after the $PLOTIN namelist.

Name	Default(s)	Description of Input
NSKIPF	0	Number of observation function files that will be skipped on observation unit before starting to read data from it.
NSKIPR	0	Number of observation functions to skip on observation unit before starting to read data.
NUMP	NONE	Number of plot points per plot array or the number of observations. There is an internal limit of 213 points.
NUMP15	.FALSE.	Logical variable. If .TRUE., NUMP will be read from observation unit as the first record.
NZCUTS	NONE	Number of contours requested.
OBSERV (100)	BLANK	Observation function names to be used for scale annotation. They are read in namelist format as: OBSERV=nHname1,nHname2,----.
OBSPLT (120)	7777	Integer definition of the plot functions. Each pair of plot arrays or observation functions is defined by a 4-digit number, the first two represent the independent variable, the second two represent the dependent variable according to the input order of the plot arrays or OBSTH functions. A zero indicates the end of one chart. More charts can be generated up to a limit of 120 entries in the OBSPLT array. User should enter a value of 7777 after the last chart.
OBSTH (2000)	NONE	Plot data in one of the following formats. For ALTVAL=.FALSE., plot arrays are loaded, A(1),--,A(N),B(1),--B(N),C(1)--C(N), For ALTVAL=.TRUE., plot arrays are loaded, A(1),B(1),C(1),------,A(N),B(N),C(N), NOTE: The maximum number of plot points is 2000 but can be changed to the alteration of 3 Fortran statements in main program as follows:

Name	Default(s)	Description of Input
		COMMON/AESOPD/ADATA(n+1)
		REALOBSTH(n)
		DATA NADATA/n/
		The value of n may be set to any desired value. Program load size is altered in direct relation to the number n.
PAGE	.FALSE.	Start a new frame at XMOVE, YMOVE from previous origin.
PRINTR	.FALSE.	Logical variable. If .TRUE., tabulation will be generated.
PRINTO	.FALSE.	Logical variable. If .TRUE., the namelist input data will be printed.
REM (30)	BLANK	Title to be placed at the top of the chart. It is read in namelist format as:
		REM=nH title of plot,
REMSIZ	0.21	Height of title in inches.
REWIND	.FALSE.	Logical variable. If .TRUE., observation unit will be rewound before reading it.
SCAL	7.5	Size of the plot device window in inches. Virtual plot specified by the maximum of XSIZE and YSIZE will be scaled to this dimension.
SCALEF (100)	1.0	Scale factor array. One for each plot array or observation function. It is used to scale one plot array relative to the others for plot purposes but does not affect the original data in OBSTH.
SCALEX	1.0	Units per inch for X.

Name	Default(s)	Description of Input
SCALEY	1.0	Units per inch for Y.
STARTX	0.0	Starting value for X-axis.
STARTY	0.0	Starting value for Y-axis.
STOP	.FALSE.	Logical variable. If .True., program will stop without generating additional plot information.
TEXT	.FALSE.	Logical variable. If .True., card images will read and plotted scaling will be in accordance with the following formula: 6*NREM*HTEXT/SCAL
XMESH (100)	None	Array of points defining the X-axis of a contour plot.
XMOVE	8.5	X-distance between plot origins. For on-line devices, a screen erasure is affected.
XORIGN	1.0	X-axis origin for the current chart relative to the current device origin specified by XMOVE.
XSIZE	6.0	X-size length in inches for virtual plot. It also specifies number of scale (and grid) divisions which will be employed. Origin is moved before plotting if the PAGE option is involked.
YMESH (100)	None	Array of points defining the Y-axis of a contour plot.
YMOVE	0.0	Y-distance between plot origins. For on-line devices, a screen erasure is affected.
YORIGN	1.0	Y-axis origin for the current chart relative to the current device origin specified by YMOVE.

Name	Default(s)	Description of Input
YSIZE	8.0	Y-axis length in inches for virtual plot. It also specifies number of scale (and grid) divisions which will be employed. Origin is moved before plotting if the PAGE option is involked.
ZCUTS (25)	None	Values of the contours.

Sample Case

A sample case illustrating the use of the program for two plots with several curves each is shown in figure 7. This sample case is stored on EX42-00002*DATA3.PLOTTR1. Output plots for MOPS, CALCOMP and SD4060 programs are shown in figures 8 through 10. For comparison, the Tektronix 4012 output using the above case data is shown in figure 2.

```
 1:  $PLOTIN
 2:   CALCOM=.TRUE.,
 3:   GRID=.TRUE.,
 4:   NAXIS=3,
 5:   XSIZE=3,
 6:   YSIZE=6,
 7:   NUMP=12,
 8:   OBSTH=0,.1,.3,.4,.6,.7,.9,.10,.12,.15,.17,.19,
 9:   0,100,300,400,600,700,900,1000,1200,1500,1700,1900,
10:   0.,2.,5.8,7.5,9.5,10.1,10.7,10.8,11.3,12.1,13.5,19.9,
11:   00.,20.,50.8,70.5,90.5,100.1,100.7,100.8,110.3,120.1,130.5,
      190.9,
12:   000.,200.,500.,700.,900.,1000.,1000.,1000.,1100.,1200.,
      1300.,1900.,
13:   NOBSER=5,
14:   OBSPLT=0102,0103,0104,0105,0,
15:   NREM=6,
16:   REM=12HARRAY FORMAT,
17:   NAMES=.TRUE.,
18:   OBSERV=6HXARRAY,6HYARRAY,
19:   XMOVE=0.,
20:   NPAGE=1, $
21:  $PLOTIN
22:   ALTVAL=T,
23:   INPUT=9,
24:   OBSTH=0,0,0,0,
25:   1,100,2,20,
26:   3,300,6,50,
27:   4,400,8,70,
28:   6,600,9,90,
29:   NOBSER=4,
30:   NUMP=5,
31:   OBSPLT=0102,0103,0104,0,7777,
32:   XORIGN=3.5,
33:   REM=12HOBSERVATIONS,
34:   OBSERV=2HX1,2HX2,2HX3,2HX4,
35:  $
36:  $PLOTIN STOP=T,$
```

FIGURE 7 INDEPENDENT PLOT PROGRAM TEST CASE INPUT.

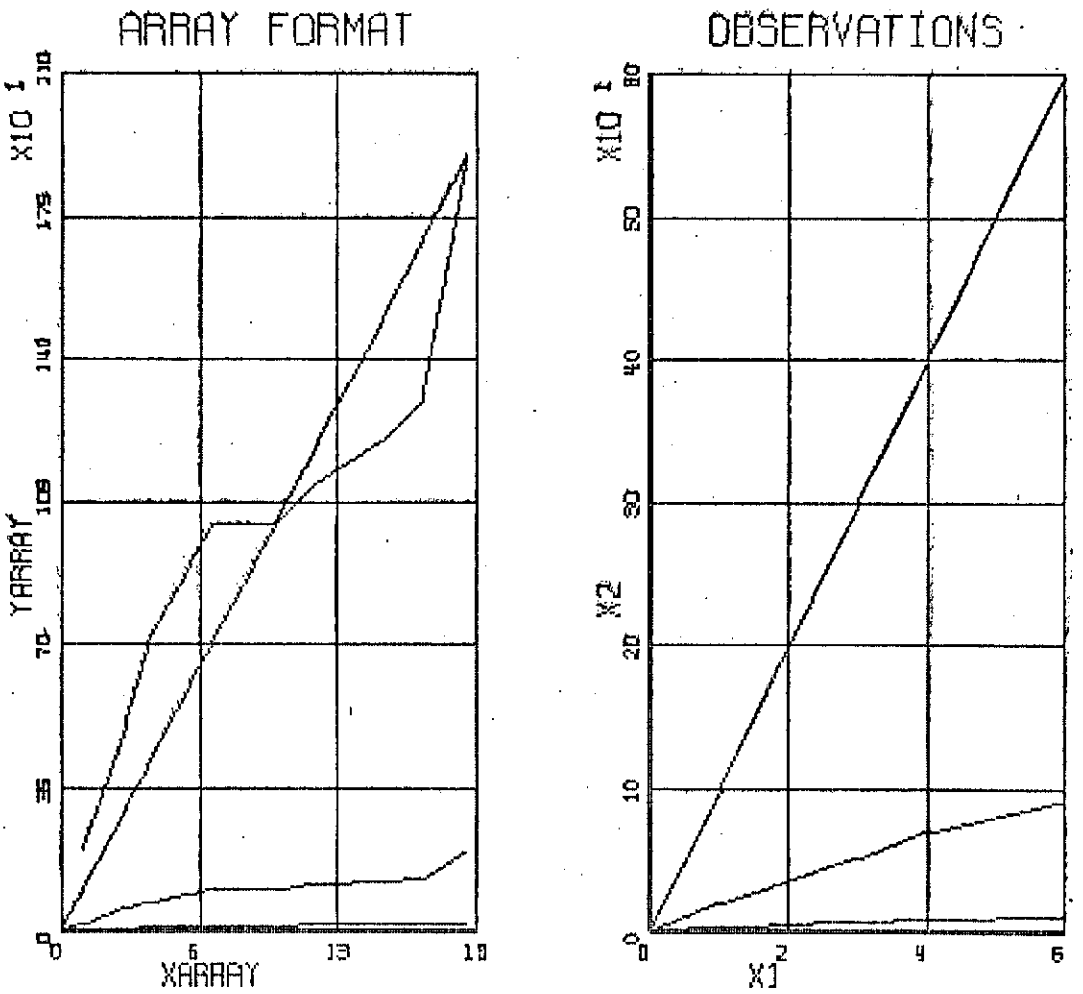


FIGURE 8 MOPS HARD COPY OUTPUT.

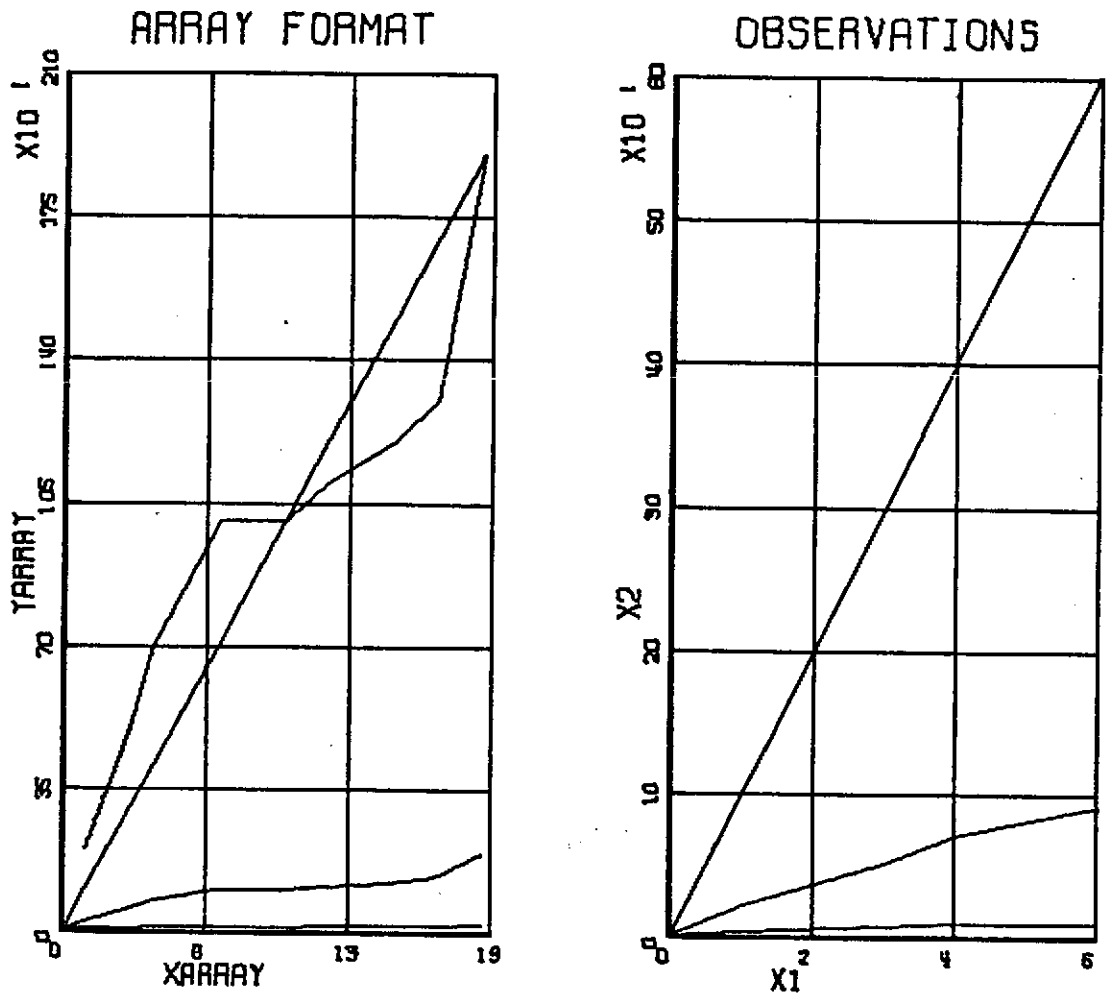


FIGURE 9 CALCOMP OUTPUT.

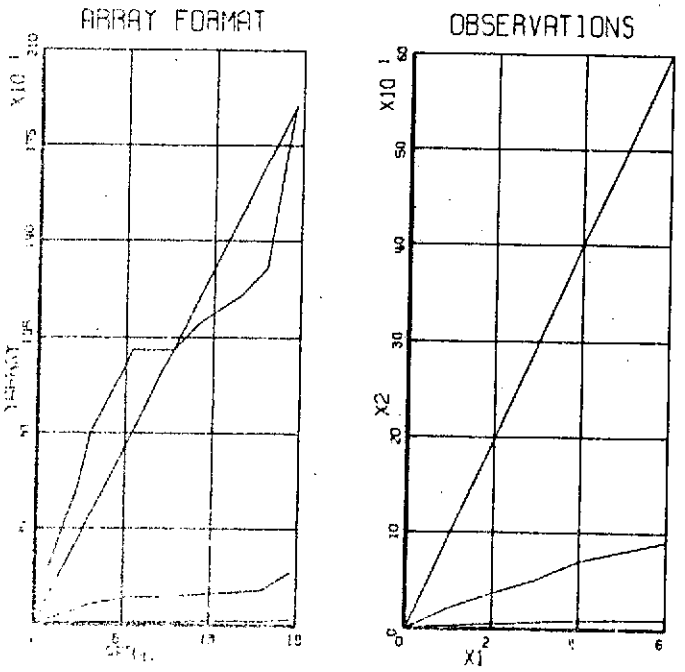


FIGURE 10 SD4060 OUTPUT.

PROGRAM DESCRIPTION

The basic PLOTTR program is written in Fortran language and interfaced to the four plot devices as illustrated in figure 11. The control program subroutines reference the CALCOMP subroutines AXIS, SCALE, NUMBER, SYMBOL and LINE. These routines have been modified slightly to call PLTT (rather than PLOT). All display device interface is accomplished through the single driver PLTT. PLTT has three basic entries:

Initialization

Plotting

Termination

which perform special functions depending upon the device interfaced. The windowing and clipping package is called by PLTT at the "plotting" entry. The data is scaled to the specified window size and vectors outside the window are deleted or modified.

Program Loading

The program is overlayed to execute in approximately 20,000 decimal but the size varies with the interface software requirements. The four program sizes are shown below:

TEKTRONIX	19400
CALCOMP	20100
SD4060	21400
MOPS	18700

The construction of the program requires the following control cards:

```
@QUAL EX42-00002
@COPY,R * PLOTTR
@MAP file .OPLOT,OPLOT
```

The basic plot software resides on the file EX42-00002*PLOTTR. There is a file containing interface software for each hardware device. The file used depends upon the interface. Map directive elements are also stored on the interface files:

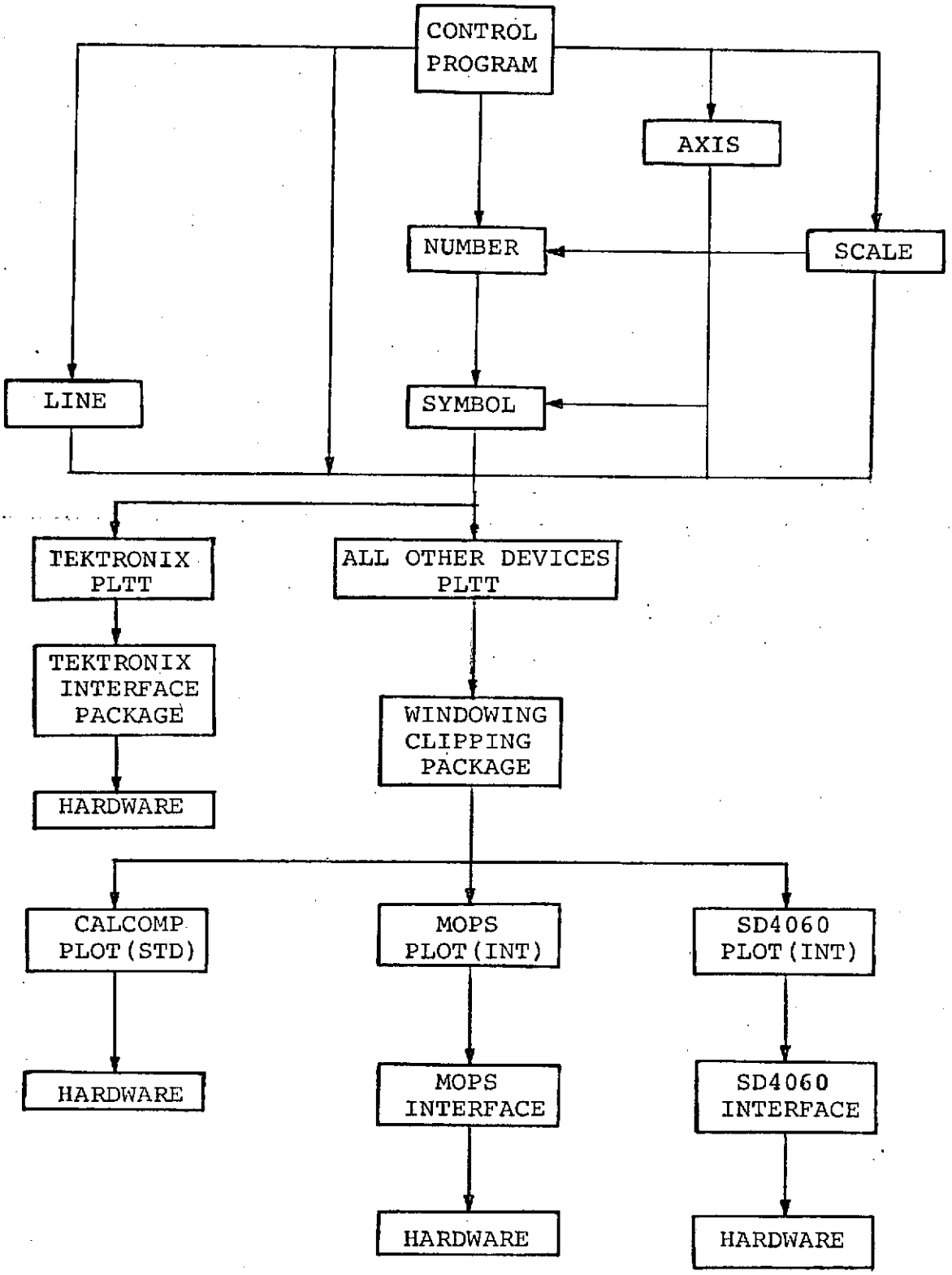


FIGURE 11 PLOTTR INTERFACE.

```
TEKTRONIX    EX42-00002*TEKTRON.OPLOT
CALCOMP      EX42-00002*CALCOMP.OPLOT
SD4060       EX42-00002*SD4060.OPLOT
MOPS         EX42-00002*HAZEL.OPLOT
```

The TEKTRONIX map directives are illustrated below:

```
LIB *TEKTRON.,*TEKLIB.
LIB *TEKTRON.,*TEKLIB.
NOT TPF$.PPLNLN
NOT TPF$.NOGRAF
NOT TPF$.DMPBUF
NOT TPF$.XYCNUT
NOT TPF$.MOVEA
NOT TPF$.DRAWA
NOT TPF$.VECMOD
NOT TPF$.SCRENE
NOT TPF$.HDCOPY
NOT TPF$.PLTT
NOT *TEKLIB.CHARS
NOT *TEKLIB.CHARS/TEKPLOT
NOT *TEKLIB.TKCH
NOT IMAGE
SEG MAIN
IN PLOTTR
SEG AA*
IN CONPLT
SEG BB*.(MAIN)
IN THROBS
SEG BB1*. (BB)
IN HPLNLN
SEG BB2*. (BB)
IN PPLNLN
END
```

Map directive elements for the other three device interfaces are shown in figure 12.

```
1:LIB *CALCOMP,MSC*LOCALIB,MSC*PLTLIB
2:LIB MSC*LOCALIB,MSC*PLTLIB,*CALCOMP
3:NOT IMAGE
4:SEG MAIN
5:IN PLOTTR
6:SEG AA*
7:IN CONPLT
8:SEG BB*,(MAIN)
9:IN THROBS
10:SEG BB1*,(BB)
11:IN HPLNLN
12:SEG BB2*,(BB)
13:IN PPLNLN
14:END
```

(A) CALCOMP.

```
1:LIB *SD4060,MSC*LOCALIB,MSC*PLTLIB
2:LIB MSC*LOCALIB,MSC*PLTLIB,*SD4060
3:NOT IMAGE
4:SEG MAIN
5:IN PLOTTR
6:SEG AA*
7:IN CONPLT
8:SEG BB*,(MAIN)
9:IN THROBS
10:SEG BB1*,(BB)
11:IN HPLNLN
12:SEG BB2*,(BB)
13:IN PPLNLN
14:END
```

(B) SD4060.

```
1:LIB *HAZEL.
2:NOT IMAGE
3:NOT HAZEL.MODE2
4:NOT HAZEL.MAIN
5:SEG MAIN
6:IN PLOTTR
7:SEG AA*
8:IN CONPLT
9:SEG BB*,(MAIN)
10:IN THROBS
11:SEG BB1*,(BB)
12:IN HPLNLN
13:SEG BB2*,(BB)
14:IN PPLNLN
15:END
```

(C) MOPS.

FIGURE 12 MAP ELEMENTS FOR PLOTTR.

SUBROUTINE DESCRIPTIONS

This section contains an alphabetically arranged list of descriptions for the Independent Plot Program Subroutines.

Program PLOTTR

PLOTTR is the main program for the Independent Plot Program. Written in Fortran, PLOTTR establishes nominal values for input variables, reads data, sets up program options, initializes the plot devices, establishes certain scaling parameters and controls the plotting of text and data. Figure 13 shows a functional flow chart for PLOTTR. The flow logic begins with the reading of namelist data called $PLOTIN. An input parameter PRINTR provides the user with the option of printing all the input data except the actual data to be plotted. If the target plot device requires initialization, the logic for initializing the device is called on the first case but not thereafter. A logical input variable, STOP, provides for a normal stop after the last case. Logic is included to obtain the actual data in three different formats.

First the data may be read in BCD format through the normal namelist input channels via an input variable called OBSTH. The data may be read in array format (the format used internally by the program) or it may be read in observation format. The latter requires that the data be reordered by the program before use. The reordering is done by a routine called WRITE15. WRITE15 simply writes the data out in binary format on a user specified unit to be read back in array format.

A third option allows the data to be read in binary observation format from a file generated outside the independent plot program. In the latter case the data is converted to array format as it is read into the program. After all data is read in the plot size is established through a call to SCRENE and the main plot generation subroutine GENPLT is called. GENPLT controls the generation of contour and x-y plot options. After return from GENPLT, the flow logic loops back to the beginning of PLOTTR to read more data.

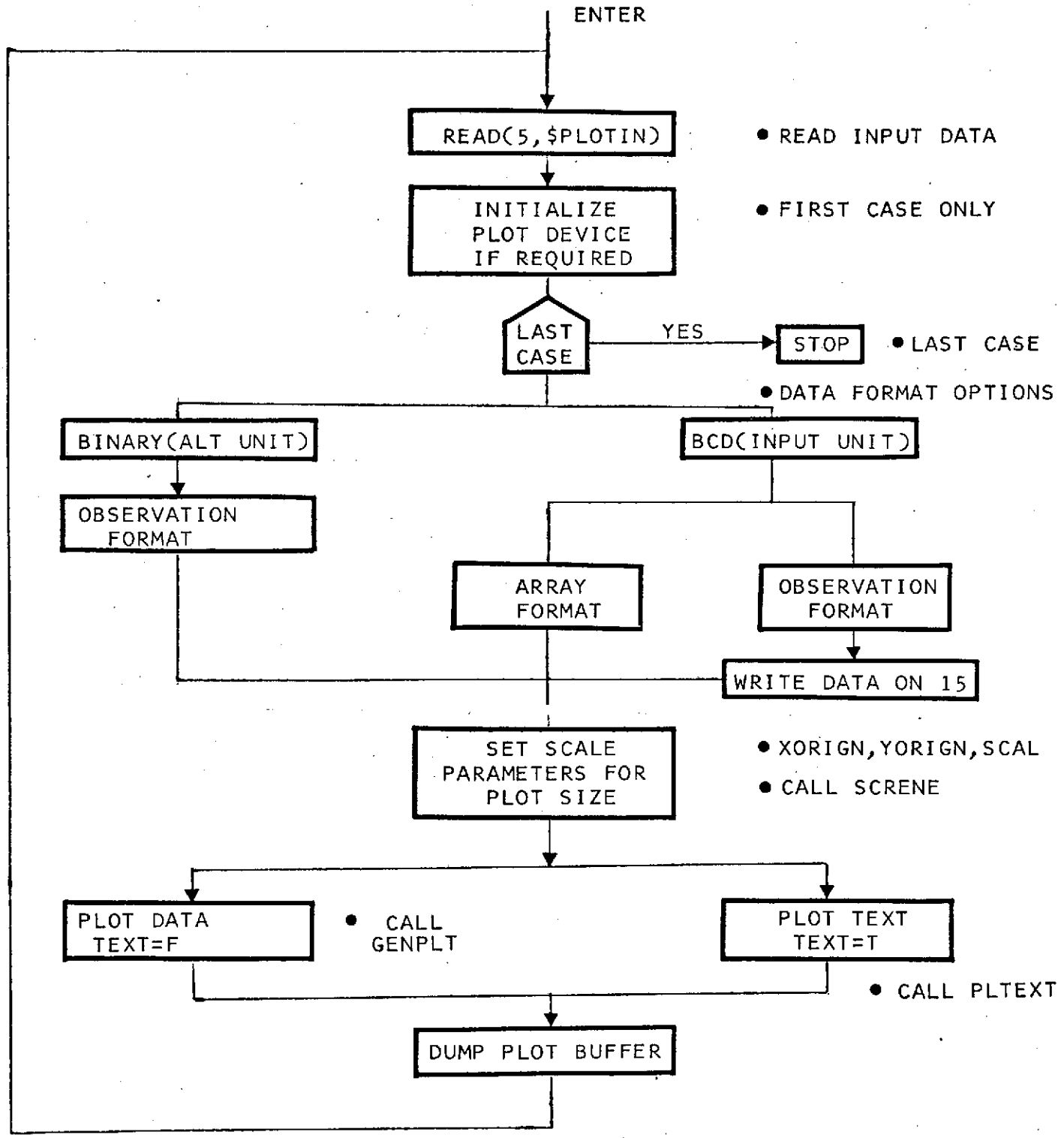


FIGURE 13 FUNCTIONAL FLOW CHART - PLOTTR.

Subroutine AXIS

AXIS is a modified CALCOMP subroutine written in Fortran for generating annotated axes with tick marks at one inch intervals. The use of the subroutine is as follows:

```
Call AXIS(X,Y,BCD,NC,SIZE,THETA,YMIN,DY)
```

X, Y	Coordinates of the starting point of the axis with respect to the plotting area origin in inches.
BCD	Character label for the axis.
NC	Number of characters in the label.
SIZE	Length of the axis in inches.
THETA	Angle of rotation measured clockwise from the x-axis in degrees.
YMIN	Functional value to be assigned to the first position on the axis.
DY	Change in the functional value for inch.

The AXIS routine is normally called after scale which determines the YMIN and DY values. AXIS calls PLOT and SYMBOL to generate the lines, tick marks and annotation on the axes. NUMBER and ROUND are used to convert binary numbers to characters which can be plotted as scale and annotation.

Subroutine CLIPT

CLIPT is a subroutine written in Fortran for clipping vectors which exceed dimensions of a specified window. The use of CLIPT is as follows:

```
Call CLIPT(BUFIN,OUTBF)
```

BUFIN A 4 element array containing the input vector to the CLIPT routine.

OUTBF A 4 element array containing the output vector from the CLIPT routine.

The routine performs a series of tests on the input vector to determine the position of this vector relative to a specified window. If the vector is completely outside the window, a flag is set to ignore the vector completely. If the vector is partially inside the window, the intersection of the window boundary is computed and a modified vector is generated which is stored in OUTBF. If the vector is completely inside the window, the output vector is the same as the input vector.

Subroutine CONPLT

CONPLT is the main subroutine for generating contour plots from an input mesh of data defined in the input array, OBSTH. The data is defined in equal increment mesh points in x and equal increments of mesh points in y. The boundary of each incremental window is searched for intersections with specified contours. Figure 14 is a functional flowchart of the subroutine CONPLT. Upon entering the subroutine, a title and scale are generated for the data. The x-limits, y-limits and z-level are set in a triple DO-LOOP. The type of intersection is determined from the values of z at the four corners of the incremental window. Figure 15 illustrates the types of intersections which can be accounted for. Each has a separate algorithm which is coded at the statement label identified in the figure. Once the boundary points are computed, the subroutine LINE is used to generate the contour vector. Contour vectors are determined for each z-level and for each incremental window. The result of all evaluations is the appearance of the continuous contour plot for the entire region specified by the x-mesh and y-mesh.

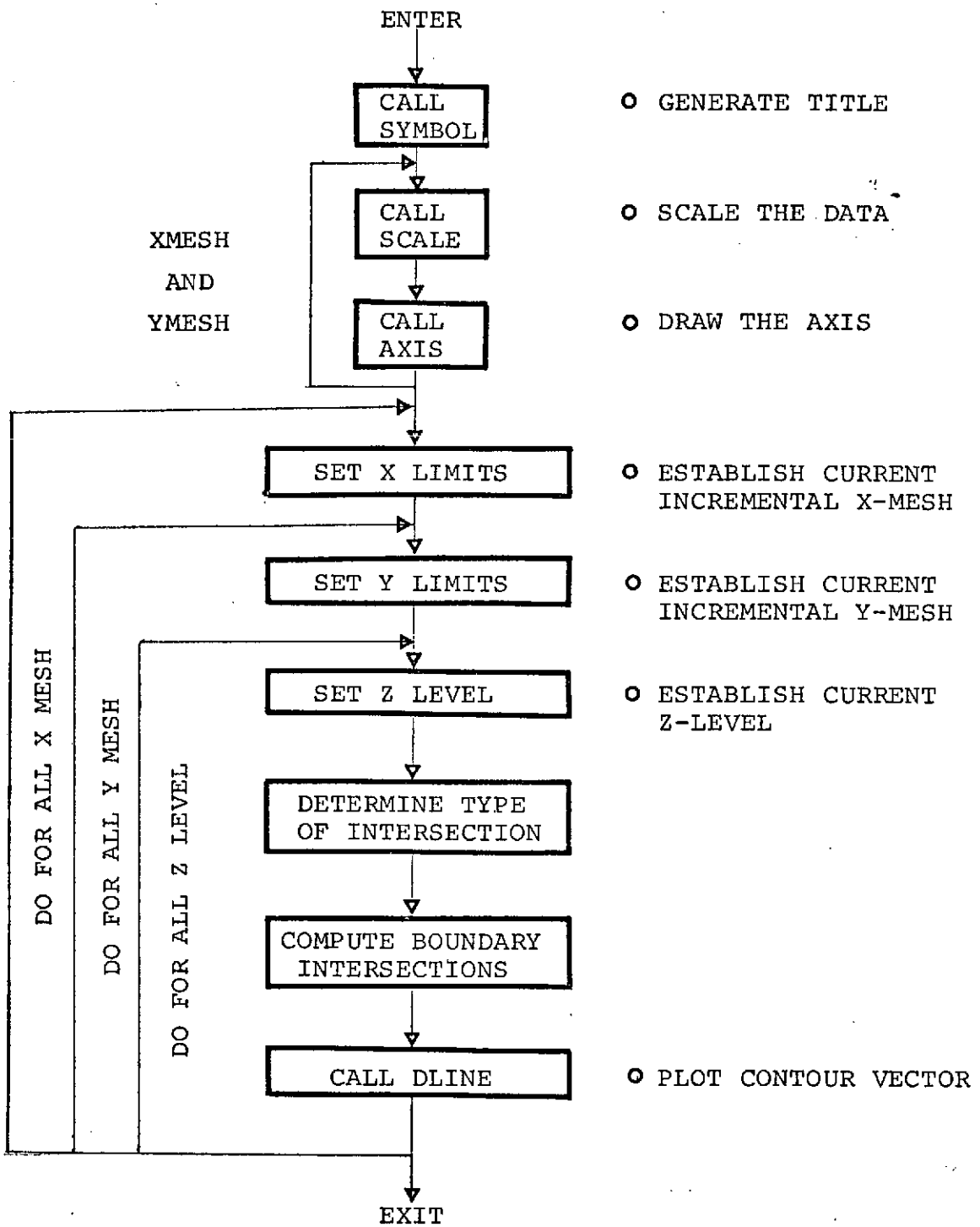
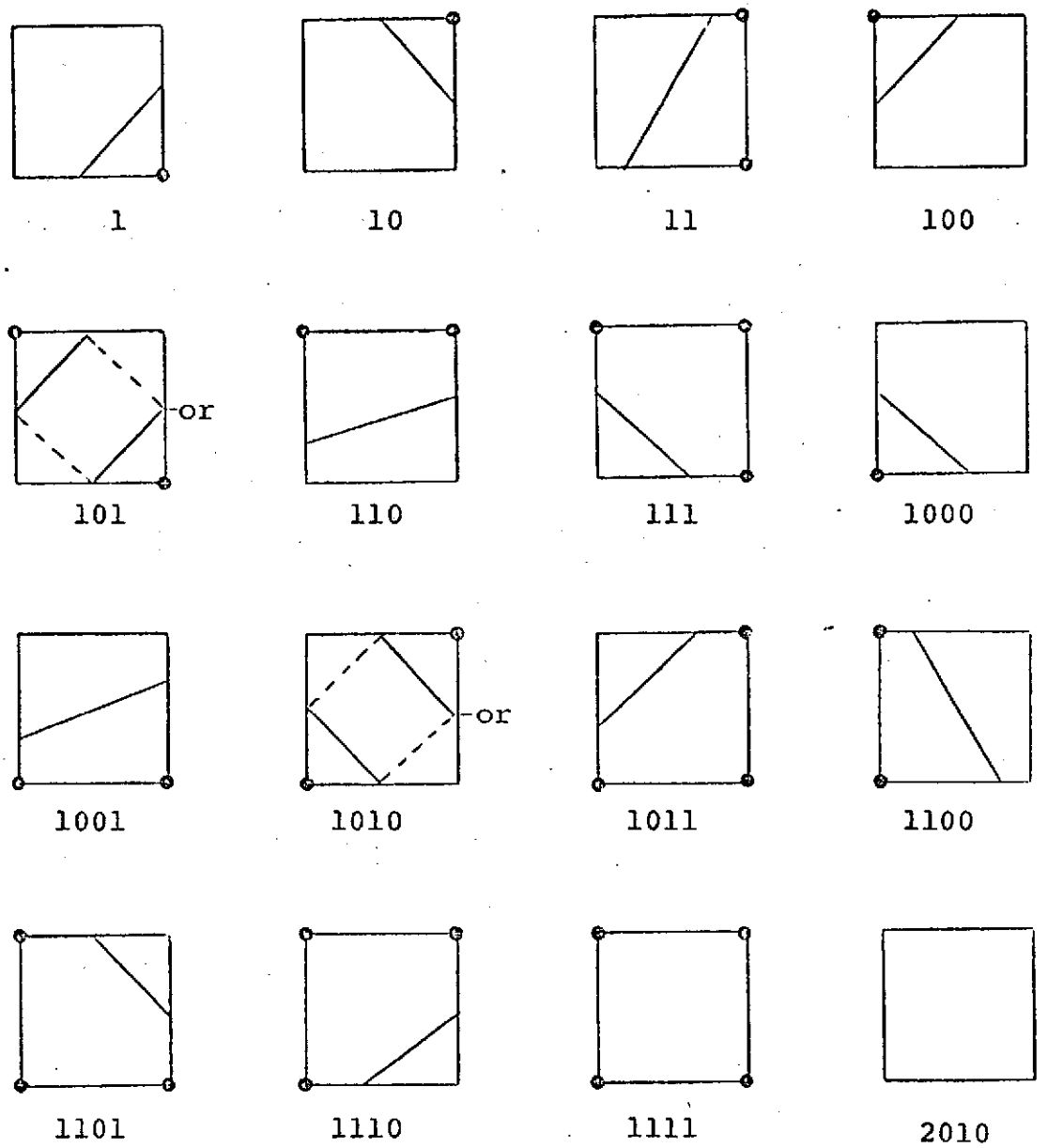


FIGURE 14 FUNCTIONAL FLOW CHART, CONPLT.



NOTE: Identification numbers represent the statement labels where the algorithm occurs.

FIGURE 15 INTERSECTION TYPES FOR CONTOUR VECTORS.

Subroutine DEF

DEF is a Fortran subroutine for ejecting a page and placing a title on the new page.

Subroutine DMPBUF

This subroutine DMPBUF is used for dumping the plot vector buffer at the completion of plot frames. The calls to DMPBUF are important for online devices (Tektronix 4012 and MOPS terminals) where completion of the plot vectors is important to the analysis. For the offline devices, (CALCOMP and SD4060) the buffer dumping function is not important since the plots are not viewed until the complete plotting process is finished. Therefore, for offline devices, DMPBUF routine is a dummy subroutine.

Subroutine DRAWA

DRAWA is a modified version of the standard Tektronixs 400 series interface routine by the same name. The usage is as follows:

```
Call DRAWA(X,Y)
```

X X-position to which a vector is to be drawn.

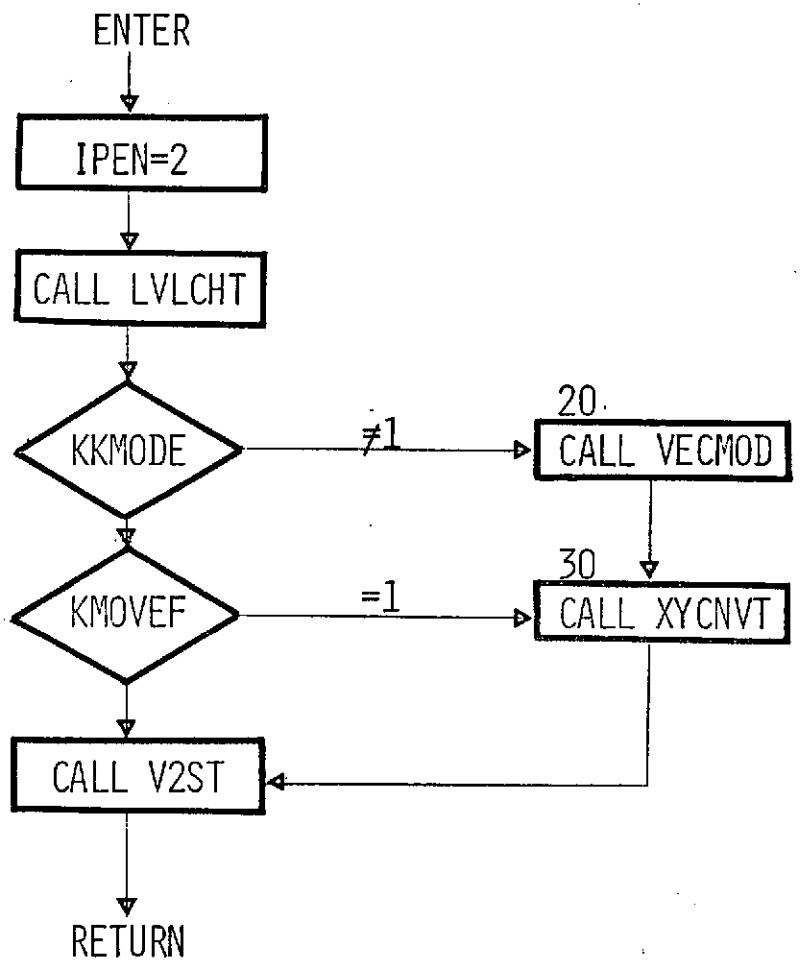
Y Y-position to which a vector is to be drawn.

The routine generates a vector on the target display device subject to the following constraints. The vector must be at least partially within the virtual window. If the vector is outside the window, no action is taken. If the vector is partially within the window, the vector is clipped so that only that portion of the vector which lies within the window is drawn. If the vector is completely inside the window, the whole vector is drawn. The routine contains logic to determine whether the previous vector was drawn and if not, an interim move of the pen or beam is performed before the vector is drawn. A software call is provided for placing the plot device in a vector mode if required. The actual data plotted is scaled from virtual coordinates to plot device coordinates before the vector is drawn.

Figure 16 is a detailed flow chart of the operation of DRAWA. Upon entry, a parameter IPEN is set to 2. This parameter is used later by the actual plot software to place the PEN in a "down" position or the beam in an "on" mode. The setting of this variable is the only modification to the DRAWA routine from the original Tektronixs code. The routine LVLCHT is called to locate the vector relative to the virtual window. A parameter KKMODE is tested to determine whether the plot device is in a vector mode. If KKMODE is not equal to 1, then a routine VECMOD is called to place the plot device in a vector mode. The library routine is a dummy routine but for those plot devices which require a special status, VECMOD can be written to perform this function.

KMOVEF is tested to determine whether an intermediate move is required before the vector is drawn. A value of one indicates that a move is required. If required, the routine XYCNVT is called to perform the pen or beam movement.

Finally, the routine V2ST is called to actually draw the vector. V2ST is coded to clip and scale the vector before drawing.



- PEN DOWN.
- LOCATE VECTOR RELATIVE TO VIRTUAL WINDOW.
- PLACE PLOT DEVICE IN VECTOR MODE.
- MOVE PEN INTO POSITION TO DRAW.
- DRAW THE VECTOR.

FIGURE 16 DETAILED FLOW CHART, DRAWA.

Subroutine ESCALE

The subroutine ESCALE is used in the generation of printer plots for the CDC 6000 version of the PLOTTR program. It scales the data to the size of the standard 11 x 16 printer page size.

Subroutine FTNBIN

FTNBIN is an assembly language routine which establishes binary blocking for sequential files on CDC 6600 computer. The usage is as follows:

```
Call FTNBIN(I,J,K)
```

I — Flag which determines whether blocking is to be on or off.

- I = 1 Binary blocking is to be turned on.
- I = 0 Binary blocking is to be turned off.

J — Number of files to be blocked or unblocked in accordance with the flag I.

K — Name of the array containing the integer numbers of the file to be blocked or unblocked.

FTNBIN can be called to block a specified set of files or can be called to block or unblock all files by setting the parameters J and K equal to zero. FTNBIN is a dummy subroutine in this library but can be replaced with the system FTNBIN routine when the program is compiled on the CDC 6600. Binary blocking is automatically provided for on Univac Exec 8 Fortran.

Subroutine GENPLT

GENPLT is a Fortran subroutine which controls the data acquisition from alternate files but also controls X-Y plot and contour plot options for the program. All data in this subroutine is passed through the common as follows:

```
Call GENPLT(NADATA,OBSTH,NUMP,INPUT,CALCOM,OBSERV,
            NOBSER,YOBSTH,CONTOR)
```

NADATA	Maximum size of the internal storage array OBSTH.
OBSTH	Internal storage array for plot data.
NUMP	Number of points for plot array.
INPUT	Logical unit for binary input data.
CALCOM	Vector plot option flag.
OBSERV	Names of the plot arrays.
NOBSER	Number of observations for plot arrays.
XOBSTH	An array containing a single observation.
CONTOR	Contour plot option flag.

A detailed flow chart for subroutine GENPLT is shown in figure 17. Upon entry, GENPLT tests the parameter INPUT to determine the source of the data. If 0, the data is assumed to be in core in the OBSTH array. Otherwise, the data will be read from the observation unit defined by the value of INPUT. The routine READ15 loads the data. Two tests are performed to determine the limits for the number of observation functions. If NOBSER is less than one, an error message is printed and the control is returned to the main program. If the number of observation functions, NOBSER is greater than NADATA/2, an error message is printed and the control is returned to the main program. The subroutine THROBS is then called to actually generate the X,Y plot. THROBS contains logic which controls the generation of printer plots, tabulations and/or vector plots. The contour plot option flag CONTOR is then tested to determine whether a contour plot is being generated. If the flag is true, the subroutine CONPLT is used to generate the contour plot. Control is then returned to the main program, PLOTTR.

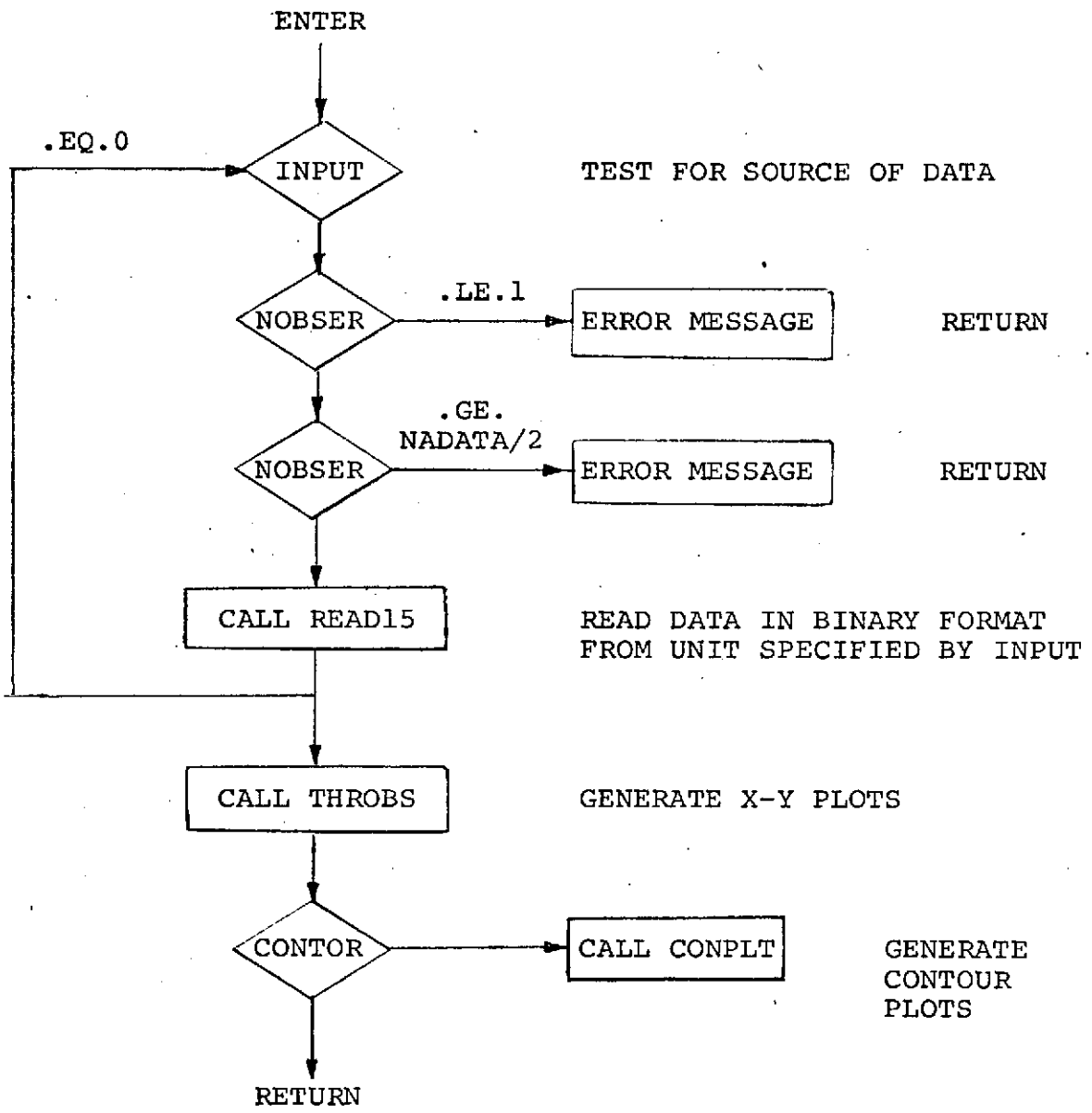


FIGURE 17 DETAILED FLOW CHART, GENPLT.

Subroutine GETMAX

This subroutine is used in the generation of paper plots for the 6600 version of the PLOTTR program. It scans a data array to determine the maximum value which can be placed on the paper plot.

Subroutine GRIDHP

GRIDHP is a Fortran subroutine used for generating for grids on a display device. The usage is as follows:

```
Call GRIDHP(XSIZE,ZSIZE)
```

XSIZE The length of the x-axis in inches.

YSIZE The length of the y-axis in inches.

The routine generates a rectangular grid pattern in one inch increments. The subroutine PLTT is used to draw the grid lines.

Subroutine GRIDXY

GRIDXY is a Fortran subroutine for generating the grid representation on printer plots. The usage of GRIDXY is as follows:

```
Call GRIDXY(XMIN,XMAX,YMIN,YMAX,BX,BY)
```

XMIN The minimum X value.

XMAX The maximum X value.

YMIN The minimum Y value.

YMAX The maximum Y value.

BX The X distance between grid lines.

BY The Y distance between grid lines.

The routine generates the representation of grid lines by placing periods (.) appropriately into the plot buffer. It also simulates the coordinate axes of the plot by placing the character I appropriately to represent segments of the Y-axis and the character (-) to represent segments of the X-axis. The character (+) is used at the grid intersection point.

Subroutine HPLNLN

This subroutine is the main driver for generating X,Y plots. It performs the data scaling, access generating, title generation and the drawing of the lines associated with the specified data arrays.

Subroutine LINE

This is a Fortran subroutine which makes the appropriate calls to generate a line on the output device. The subroutine has six calling arguments as follows:

CALL LINE(X,Y,N,K,J,L)

X,Y — X and Y are arrays containing the X and Y coordinates respectively for the n points to be plotted.

K — The significance of K is two fold.

(1) The magnitude of K specifies that the data is stored in every K cell.

(2) If the sign of K is positive, the pen will be moved to the first point. If the sign of K is negative, the pen will be moved to the first point in a lowered position.

J — The magnitude of J is the number of points for every point to be plotted. Where J equals 0, a line plot only. Where J equals 1, a point for every data point. Where J equals 5, a point will be plotted at every 5th. point. A minus sign of J specifies a point plot without connecting points. A positive J specifies a line connecting every data point.

L — The value of L specifies the type of symbol to be used. Integer numbers 1 through 22 may be specified. These symbols are different for different plot devices.

Subroutine LVLCHT

This subroutine is a calling program for the determination of whether the current pen position is inside or outside of the virtual window.

Subroutine MOVEA

This subroutine is used to move the pen position from the current position to the position requested by the calling sequence. The windowing and clipping software are called by this routine to determine whether the requested point is inside or outside the specified window. See DRAWA for clipping and windowing discussion.

Subroutine NUMBER

This subroutine is used to convert a floating point number to be BCD and to draw the resulting alpha numeric characters on the plot. The calling sequence is:

```
Call NUMBER(X,Y,HGHT,FPN,THETA,N)
```

X,Y — X and Y specify the position of the first character in inches.

HGHT — The height of the characters.

FPN — The value of the floating point number.

THETA — The angle at which the characters will be drawn, measured from the x-axis, positive in a counter-clockwise direction.

N — This is the number of decimal places to be retained in the conversion. N may be specified at any value from -1 to 11. If N is -1 or 0, no decimal places will be drawn. If N is -1, a decimal point will be suppressed. Any other value of N specifies the number of decimal places to be drawn.

The integer portion of the number is restricted to a maximum of 6 characters. The decimal portion is restricted to 11 digits.

Subroutine PAPERP

This subroutine serves as a calling program for controlling the X,Y plot options. The two options are as follows:

If PRINTR is .TRUE., then a paper plot will be generated on the normal output (6600 only).

If CALCOM is .TRUE., then a vector plot will be generated on whatever device is interfaced to the PLOTTR program.

Subroutine PARCLT

This subroutine contains the logic for determining whether a point is inside or outside a window bounded by specified maximum and minimum values in the X and Y direction.

Subroutine PCLIPT

This subroutine contains the logic for determining whether a X,Y point determined from the calling sequence is within the bounds specified by the common variables TMINVX, TMAXVX, TIMVY, and TMAXVY. If not within the specified bounds, the flag KERROR is set to 1.

Subroutine PLCPTS

This subroutine is used in the generation of paper plots on the CDC 6600. It contains the logic for placing plot characters within a buffer region in the appropriate position to simulate a plot when the buffer is printed on the normal 11 x 16 printer output. It also flags and identifies the points which did not fall within the plot region.

Subroutine PLTEXT

This subroutine is used for generating auxiliary plot text associated with X,Y plots or contour plots. The subroutine is called from the main program when the option TEXT is specified as .TRUE. Upon entering this subroutine, the plot window is established by input values of DXG, DYG and SCAL. Then cards are read from the input and the characters on the card are scaled and placed within the established window. Spacing between card images is automatically provided at 1.5 times the character height. The buffer is dumped at the completion of the plotting.

Subroutine PLTT

This subroutine is a replacement subroutine for PLOT. PLTT is called by the axis symbol and number drawing routines. It drives the clipping and windowing package. The calling sequence is:

```
Call PLTT(XPAGE,YPAGE,IPEN)
```

XPAGE X-coordinate of point to be plotted.

YPAGE Y-coordinate of point to be plotted.

IPEN Pen Position

IPEN = 2, Call DRAWA

IPEN = 3, Call MOVEA

Otherwise, Call PLOT

MOVEA and DRAWA are driver routines for the clipping and scaling package. After performing the scaling and clipping, the MOVEA/DRAWA, ultimately call PLOT, the hardware device interface. For IPEN values other than 2 and 3, the hardware device interface is called directly.

The subroutine PLTT is used for all device interfaces except the Tektronix terminal. For Tektronix, PLTT is an entry in the PLOTS hardware interface. Since clipping and scaling are an integral part of the Tektronix package, the PLOTTR software for this purpose is not used.

Subroutine PPLNLN

PPLNLN is a Fortran subroutine which generates X,Y plots as part of the normal printed output. The subroutine is used as a utility routine as follows:

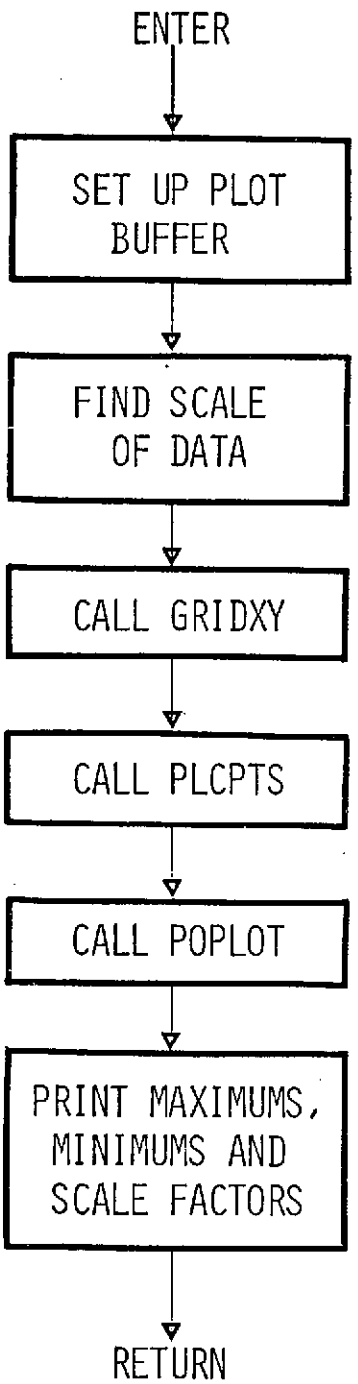
```
Call PPLNLN(X,Y,NPTS,XMAX,XMIN,YMAX,YMIN,NPLTS,TT,TB)
```

X	Array of X points to be plotted.
Y	Array of Y points to be plotted.
NPTS	Number of points to be plotted.
XMAX	Maximum X-value.
XMIN	Minimum X-value.
YMAX	Maximum Y-value.
YMIN	Minimum Y-value.
NPLTS	Number of points.
TT	Title array at the top of the plot.
TB	Annotation at the bottom of the plot.

Upon entry into the PPLNLN routine, a plot vector area is set up which will contain the characters representing the grid, scales and plotted information. Scale factors are determined based on the size of the paper (11 x 16). The plot area is hard coded to be seven inches on the Y-axis and ten inches on the X-axis. Therefore, the resolution is 100 units in the X direction and 42 units in the Y direction. The subroutine GRIDXY is called to set up a grid in the plot buffer. Vertical grid lines are made up of the character minus. The PLCPTS routine is called to place the plot points into the plot buffer. The plot data points are identified by characters in the following sequence:

```
X,O,+,$,/,2,3,4,5,A,B,C,D,E,F,G,H,K,M,N,P,R,S,U,V,W,X,Y,Z,
```

Thirty characters are supplied for up to thirty plots which may be specified for single chart. Plot points which fall within the same resolution area use the character asterisk to replace the normal plot character. The subroutine POPLOT is called to print out the plot buffer and place the title and annotation on the printed page. The maximum, minimum and scale factors for all of the data are printed at the bottom of the plot. The functional flow chart for PPLNLN is given in figure 18.



- SET UP GRID IN PLOT BUFFER.
- PLACE PLOT POINTS IN PLOT BUFFER.
- PRINT OUT PLOT BUFFER WITH TITLES.

FIGURE 18 FUNCTIONAL FLOW CHART - PPLNLN.

Subroutine READ15

This subroutine reads observation functions from the specified input unit and reorders the data into plot arrays. The subroutine contains an option to obtain the number of points as the first record on the observation function file.

Subroutine REVCOT

This subroutine converts a raster position on the plot device to X,Y coordinates in a virtual window using the virtual window scaling parameters which were previously stored in TKTRNX common.

Subroutine SCALE

The subroutine is used to find the minimum and maximum values for a given set of data and determines the reasonable scale values to be placed within the plot dimension. It also establishes scale factors for actually scaling the data to fit on the plot. The calling sequence is:

```
Call SCALE(X,S,N,K)
```

X	An array containing N data values which are to be scaled. The data being stored in every Kth. cell. That is, X(1) X(1+k) X(1+2k), etc.
S	The length of the plot over which the data is to be plotted in inches.
N	The number of points in the X-array.
K	The repeat cycle for data elements in the X-array.

The routine scans all of the elements in the array to find the maximum and minimum. It adjusts these values to give reasonable scale values for the start value and the maximum value. The two values are saved in the last elements of the X-array for use by the line subroutine in scaling the data for plotting. The X-array must be dimensioned two extra elements for every variable set in the array.

Subroutine SCRENE

This subroutine is used for establishing the plot device window size in terms of the number of raster units per inch for the particular device. The scale factor is different for each device. For example, the CALCOMP uses a scale factor of 100 raster units per inch.

Subroutine SWINDO

This subroutine stores the specified plot device window size and position into the TKTRNX common area for use in scaling plot vectors to fit within the plot device window.

Subroutine SYMBOL

Subroutine SYMBOL is the modified CALCOMP assembly language routine for converting a string (array) of alpha-numeric information into plot vector format. It works from a directory of stored plot vector sequences which represent the standard character set. The characters in the string are identified and their plot vector representation is converted to the proper format through calls to the subroutine PLTT. The user has the option of selecting the start position in terms of X and Y, the size of the characters to be used, the angle to which the data is to be placed and the number of characters to be placed. SYMBOL is called as follows:

```
Call SYMBOL (X,Y,SIZE,FDATA,THETA,N)
```

X	The X-coordinate of the left most raster unit of the first character to be plotted, inches.
Y	The Y-coordinate of the lower most raster unit of the first character to be plotted, inches.
SIZE	The height of the character, inches.
DATA	The starting location of the array of characters to be plotted.
THETA	The angle of inclination of the character string, degrees.
N	The number of characters to be plotted by the SYMBOL routine.

The character height is a variable in the subroutine but the width-to-height ratio is fixed at 4/7. Since the characters are stored in a series of bi-octal offset pairs for a 4 x 7 matrix, the reference origin for the offset pairs, which define each character, is the lower left corner of the matrix. The X and Y values define the location of the lower left hand corner of the first character to be plotted for this entry. Subsequent characters to be plotted are spaced from the previous character origin by 6/7 of the specified character heights.

Subroutine THROBS

This is the main control program for the generation of tabulated output and also serves some scaling functions for both printer plots and vector plots. The subroutine sorts the data which is stored in array format and prints it eight columns and 54 rows per page. The array title is printed at the top of each page.

Subroutine VECMOD

This subroutine is device dependent and is used for interfaces which require different modes of operation for generating vectors and for generating characters. A call to VECMOD for these device interfaces would place the device in a vector mode of operation.

Subroutine VWINDO

This subroutine establishes the lower lefthand corner and the extent in X- and Y- of the virtual window. For the PLOTTR program the virtual window is defined by the larger of XSIZE and YSIZE plus a margin of twice the title height on all sides of the specified grid. The user has no control over the virtual window size.

Subroutine V2ST

This subroutine controls the clipping, windowing and plotting of the actual data defined in virtual space and placing that data in the specified window on the plot device.

Subroutine WINCOT

This subroutine scales the input data in accordance with previous specified virtual and plot device window sizes and calls the routine which actually places the point on the display.

Subroutine WRITLU

This subroutine is used for reformatting data, which was input as observation functions, into plot arrays for storage and use by the program.

Subroutine XYCNVT

This subroutine controls the actual plotting or placement of data on the plot device. There are two versions available. In the Tektronix version, this routine actually fills the character buffers which are transferred from the host computer to the Tektronix terminal for plotting. The other version acts as an interface which calls the appropriate hardware conversion routine that generates points and vectors.

APPENDIX A

Common /COMON/ Description

Location	Local Name	Description
1	NUMP	Number of points per plot array.
2	ONDAT	The logical unit number for the file containing observation functions in binary format. INDAT is set in accordance with the input variable INPUT. If INPUT is 0, the observation file is not used. Otherwise, data is expected on the unit number specified in INPUT.
3	CALCOM	A logical variable. If true, vector plot file will be generated.
4	OBSERV(100)	Hollerith array containing the annotation names (left justified and blank filled) for the plot arrays.
104		Not used.
105	NOBSER	Number of observation functions for plot arrays.
106	OBSPLT(120)	An integer array containing the plotting instructions. See program input section for definition.
226		Not used.
227	REM(30)	An integer array containing the title of the plot in hollerith characters.
257		Not used.
258	XOBSTH(100)	A real array used for reading observations from the observation function file.
358		Not used.
359	NPAGE	Integer page number used for printed plots.

Location	Local Name	Description
360	SCALEF(100)	An array of scale factors used for scaling plot arrays relative to one another.
460		Not used.
461	XSIZE	The length of the X-axis in the virtual window in inches.
462	YSIZE	The length of the Y-axis in the virtual window in inches.
463	LINTYP	An integer variable defining the type of line used for the current plot. See program input for description.
464	INTEQ	An integer variable defining the plot symbol to be used.
465	STARTX	Starting location for the X-axis in the virtual window in inches.
466	STARTY	Starting location for the Y-axis in the virtual window in inches.
467	SCALEX	Scale factor of the X-axis data in the virtual window in data units per inch.
468	SCALEY	Scale factor of the Y-axis data in the virtual window in data units per inch.
469	MYX	A logical variable set by input. If true, no X-axis or scaling will be generated.
470	MYY	A logical variable set by input. If true, no Y-axis or scaling will be generated.
471		Not used.
472	NREM	Number of words used for the title.
473	REMSIZ HTEXT	Character height of the title in inches.
474	PRINTR	A logical variable set by input. If true, printer plot will be generated.

Location	Local Name	Description
475	XMESH(100)	An array of mesh points in the X-direction for contour plots.
575	YMESH(100)	An array of mesh points in the Y-direction for contour plots.
676	YCUTS(25)	An array containing desired contour values.
701	NZCUTS	The number of contour lines requested.
702	CONTOR	A logical variable set by input. If true, contour plots will be generated.
703	XMOVE	A value describing the movement of the plot origin in the X-direction after plotting.
704	YMOVE	A movement of the plot origin in the Y-direction after plotting.
705	EVRCAL	A logical variable set internally. If true, the plot device has been initialized.
706	NSKIPR	A number of records on the observation function file to be skipped before reading the data.
707	NSKIPF	A number of logical files to be skipped on the observation file before reading the data.
708	NAXIS	The number of lines incrementally displaced to be used to represent the plot axes.
709	GRID	A logical variable set by input. If true, a plot grid will be generated at one inch intervals in the virtual window.
710		Not used.
711	REWIND	Logical variable set by input. If true, the observation file will be rewound before reading the data.

Location	Local Name	Description
712	SCAL	The size of the plot device window in inches. The maximum axis length (defined by XSIZE and YSIZE) in the virtual space will be scaled to this dimension.
713	XORIGN DXG	The X-coordinate of the plot device window with respect to the plot device origin.
714	YORIGN DYG	The Y-coordinate of the plot device window with respect to the plot device origin.

Common /ADATA/ Description

Location	Local Name	Description
1	NADATA	The number of locations used for storing the data to be plotted. NADATA is set by a data statement in the main program PLOTTR.
2	OBSTH(2000)	The real array containing the data being plotted 2000 locations are provided for internal storage of data. This number may be altered at compile time by the adjustment of dimension statements and data statements in the main program, PLOTTR, as follows: COMMON/AESOPD/ADATA(N+1) REAL OBSTH(N) DATA NADATA/N The value of N is now 2000 but can be set to any value by the programmer. The size of the program is increased or decreased by the dimensions of this array.

Common /TKTRNX/ Description

Loc.	Name	Use	Description
1	KBAUDR	General	Characters per Second
33	KBEAMX	Direct Graphics	Beam X-Coordinate
34	KBEAMY	Direct Graphics	Beam Y-Coordinate
40	KDASHT	Virtual Graphics	Dash Specification
2	KERROR	General	General Error Flag
3	KGRAFL	General	Graphic Level Flag
4	KHOMEY	General	Home Y-Value
6	KHORSZ	A/N	Character Horizontal Size
13	KHORZT(10)	A/N	Horizontal Tab Table
8	KITALC**	A/N	Italic Flag
5	KKMODE	General	Mode
10	KLMRGN	A/N	Left Margin
43	KMAXSX	Virtual Graphics	Screen Window Maximum X
44	KMAXSY	Virtual Graphics	Screen Window Maximum Y
41	KMINSX	Virtual Graphics	Screen Window Minimum X
42	KMINSY	Virtual Graphics	Screen Window Minimum Y
35	KMOVEF	Direct Graphics	Move Flag
36	KPCHAR(4)	Direct Graphics	Previous Plot Characters
11	KRMRGN	A/N	Right Margin
9	KSIZEF**	A/N	Size Flag
12	KTBLSZ	A/N	Tab Table Size
7	KVERSZ	A/N	Character Vertical Size
23	KVERTT(10)	A/N	Vertical Tab Table
51	TIMAGX	Virtual Graphics	Imaginary Beam X
52	TIMAGY	Virtual Graphics	Imaginary Beam Y
47	TMAXVX	Virtual Graphics	Virtual Window Maximum X
48	TMAXVY	Virtual Graphics	Virtual Window Maximum Y
45	TMINVX	Virtual Graphics	Virtual Window Minimum X
46	TMINVY	Virtual Graphics	Virtual Window Minimum Y
53	TRCOSF	Virtual Graphics	Relative Vector Cosine Factor

Loc	Name	Use	Description
49	TREALX	Virtual Graphics	Real Beam X
50	TREALY	Virtual Graphics	Real Beam Y
55	TRSCAL	Vi tual Graphics	Relative Vector Scale Factor
54	TRSINF	Virtual Graphics	Relative Vector Sine Factor

**Used with implementations supporting the Tektronix 4002A Graphic Computer Terminal.

REFERENCES

1. Hague, D. S. and Glatt, C. R.: Optimal Design Integration of Military Flight Vehicles - ODIN/MFV. United States Air Force Flight Dynamics Laboratory Report AFFDL-TR-73-132, 1973

2. Glatt, C. R., Hague, D. S. and Watson, D. A.: DIALOG: An Executive Computer Program for Linking Independent Programs. National Aeronautics and Space Administration Contractor Report CR-2296. Washington D. C. 1973.