M T S

The Michigan Terminal System

Volume 11: Plot Description System

August 1978

Updated April 1980 (Update 1)
Updated September 1982 (Update 2)
Updated April 1985 (Update 3)

The University of Michigan Computing Center
Ann Arbor, Michigan

```
******************************************************
*                                                    *
*      This obsoletes the April 1971 edition.        *
*                                                    *
******************************************************
```

DISCLAIMER

The MTS Manual is intended to represent the current state of the Michigan Terminal System (MTS), but because the system is constantly being developed, extended, and refined, sections of this volume will become obsolete. The user should refer to the Computing Center Newsletter, Computing Center Memos, and future updates to this volume for the latest information about changes to MTS.

PREFACE

The software developed by the Computing Center staff for the operation of the high-speed processor computer can be described as a multiprogramming supervisor that handles a number of resident, reentrant programs. Among then is a large subsystem, called MTS (Michigan Terminal System), for command interpretation, execution control, file management, and accounting maintenance. Most users interact with the computer's resources through MTS.

The MTS Manual is a series of volumes that describe in detail the facilities provided by the Michigan Terminal System. Administrative policies of the Computing Center and the physical facilities provided are described in a separate publication entitled *Introduction to Computing Center Services*.

The MTS volumes now in print are listed below. The date indicates the most recent edition of each volume; however, since volumes are updated by means of CCMemos, users should check the Memo List, copy the files *CCMEMOS or *CCPUBLICATIONS, or watch for announcements in the *Computing Center Newsletter*, to ensure that their MTS volumes are fully up to date.

Volume 1: *The Michigan Terminal System*, January 1984
Volume 2: *Public File Descriptions*, April 1982
Volume 3: *System Subroutine Descriptions*, April 1981
Volume 4: *Terminals and Networks in MTS*, March 1984
Volume 5: *System Services*, May 1983
Volume 6: *FORTRAN in MTS*, October 1983
Volume 7: *PL/I in MTS*, September 1982
Volume 8: *LISP and SLIP in MTS*, June 1976
Volume 9: *SNOBOL4 in MTS*, September 1975
Volume 10: *BASIC in MTS*, December 1980
Volume 11: *Plot Description System*, August 1978
Volume 12: *PIL/2 in MTS*, December 1974
Volume 13: *The Symbolic Debugging System*, April 1985
Volume 14: *360/370 Assemblers in MTS*, May 1983
Volume 15: *FORMAT and TEXT360*, April 1977
Volume 16: *ALGOL W in MTS*, September 1980
Volume 17: *Integrated Graphics System*, December 1980
Volume 18: *The MTS File Editor*, September 1982
Volume 19: *Tapes and Floppy Disks*, February 1983

Other volumes are in preparation. The numerical order of the volumes does not necessarily reflect the chronological order of their appearance; however, in general, the higher the number, the more specialized the volume. Volume 1, for example, introduces the user to

MTS and describes in general the MTS operating system, while Volume 10 deals exclusively with BASIC.

The attempt to make each volume complete in itself and reasonably independent of others in the series naturally results in a certain amount of repetition. Public file descriptions, for example, may appear in more than one volume. However, this arrangement permits the user to buy only those volumes that serve his or her immediate needs.

Richard A. Salisbury,

General Editor

August 1978

PREFACE TO VOLUME 11

This edition of Volume 11 replaces the April 1971 edition. It has been completely reorganized with a number of new sections. At the same time, major improvements were made to the subroutines in the Plot Description System (PDS). Unfortunately, these changes are not completely compatible with previous versions of the PDS. For this reason, some of the new subroutines have different names than the old ones. The old subroutines still exist and will continue to exist indefinitely. The old routines are described in Appendix B since a large number of programs exist which use them. No new programs should be written using these routines.

Most of the symbol sets in Appendix A were obtained from National Technical Information Service. The tables were originally developed by A. V. Hershey of the Naval Weapons Laboratory. They were augmented by Norman M. Wolcott and James Hilsenrath of the National Bureau of Standards.

This revision of the volume was done by Richard L. Wiersma, Computing Services Center, Wayne State University, and by Gail H. Lift, U of M Computing Center.

Contents

INTRODUCTION

The Plot Description System (PDS) is a library of subroutines to aid in producing plots. Subroutines are provided to draw commonly used objects such as lines, symbol strings, circles, axes, and so on. Coordinates given to the PDS may be either rectangular or polar and/or linear or logarithmic.

The subroutines in the PDS use the data passed to them to produce a plot description which is written to a disk file or a magnetic tape. This plot description file contains all information needed to draw the plot. Once the plot description file is produced, it may be queued for drawing on the Computing Center’s plotter using the *CCQUEUE program or it may be drawn on a wide variety of graphics terminals by the *PLOTSEE program.

The subroutines in the PDS are written partly in FORTRAN and partly in 360/370 assembler language. They may be called directly by programs written in FORTRAN, 360/370 assembler, or any other language which supports S-type calling sequences. MTS Volume 7, PL/I in MTS, describes how to call non-PL/I subroutines from PL/I programs. These techniques may be used to call the subroutines in the PDS.

All of the example programs in this volume are written in FORTRAN. Since the text of this volume was reduced in size about 10 percent for publication, the examples shown are smaller than the actual plots produced by running the programs.

BASIC USE OF THE PLOT DESCRIPTION SYSTEM

This section describes the use of the Plot Description System subroutines to produce plots. It is intended to provide a working knowledge of the PDS subroutines and their interactions. It is not intended to be a comprehensive description of all of the PDS subroutines. A complete description of all of the subroutines is in the section "PDS Subroutine Descriptions."

In using all of the subroutines in the PDS, it is very important that the correct argument types are used. If a subroutine requires a REAL number, the argument must be either a REAL variable or a REAL constant (with a decimal point). Similar rules apply to INTEGER arguments. A few PDS subroutines allow use of more than one type in some arguments. These exceptions are noted in the text. All variables used in examples in this volume follow the FORTRAN convention (A-H, O-Z are REAL, I-N are INTEGER).

Many of the examples in this section have subroutine arguments which are constants and are not explained in this text. These arguments are not relevant to the current topic and a constant value is used to prevent confusion. These arguments are explained either later in this section or in the section "PDS Subroutine Descriptions." The constant values used in the examples may be used for most applications.

RUNNING A PROGRAM

The object modules of the Plot Description System subroutines are in the file *PLOTSYS. These routines may be loaded by concatenating *PLOTSYS to the file containing the object of the main program. The PDS writes the plot description on logical I/O unit 9. Therefore, unit 9 must be assigned on the $RUN command. The $RUN command should look something like

```
$RUN objectfile+*PLOTSYS 9=plotfile
```

where "objectfile" is the name of the file containing the main program which calls the PDS subroutines and "plotfile" is a file used by the PDS to contain the plot description. "plotfile" may be either the name of a disk file or the pseudodevice name of a magnetic tape. If a magnetic tape is used, the logical record length must be at least 255.

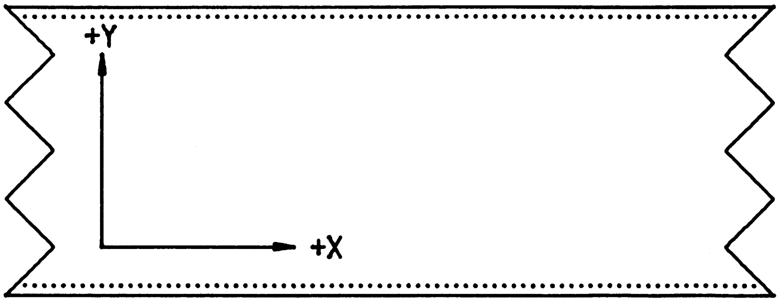
Once a plot description has been produced, it may either be queued for drawing on the Computing Center’s plotter or drawn on a graphics terminal (or both). The *CCQUEUE program submits plots to be drawn on the plotter. The *PLOTSEE program displays plots on a graphics terminal. *CCQUEUE is described in the section "Submitting Plots." *PLOTSEE is described in the section "Previewing Plots on Graphics Terminals."

BASIC CONCEPTS

Pictures to be drawn must ultimately be specified as a set of X,Y coordinates of the end points of straight line segments. The PDS subroutines transform all information given to them into such coordinates and write them into the plot description file.

The X and Y coordinates must always be positive and measure distances in inches. The paper used on the plotter comes in a roll 36 inches wide and 120 feet long. Because of the holes punched in the edges of the paper for the drive mechanism, only 33 inches of the width may be used. All 120 feet of the paper could be used. However, to prevent excessive accidental paper use, the length is limited. This limit may be up to 360 inches per plot. It is 36 inches per plot by default. The width may not, of course, be extended.

The X coordinate is directed along the length of the paper. The normal range for X coordinates is then 0 to 36 (inches). This may be extended to 0 to 360. The Y coordinate is directed across the width of the paper. The range for Y coordinates is 0 to 33 (inches). The figure below shows the layout on the paper.



INITIALIZATION AND TERMINATION

In normal use, the PDS does not require an initialization subroutine to be called. However, if 36 inches is not enough paper, a call must be made to PLTXMX to extend it. A call such as

```
CALL PLTXMX(XWID)
```

should be made before calling any other PDS subroutine. XWID is a REAL variable or constant specifying the maximum amount of paper that can be used per plot. It must not exceed 360. For instance,

```
CALL PLTXMX(72.)
```

requests a maximum of 72 inches of paper per plot. If more than one plot is produced, it is <u>not</u> necessary to call PLTXMX before each one. The limit remains in effect until the end of the run or until PLTXMX is called again.

The statement

```
CALL PLTEND
```

<u>must</u> appear as the last call to the PDS for each plot. If PLTEND is not called, the plot is likely to be incomplete. Calling PLTEND terminates the current plot and prepares for another using a new region of the paper. The coordinate system is reset so that the point (0,0) is at the lower-left corner of the new region.

<u>LINES</u>

The subroutines PENUP (pen up) and PENDN (pen down) are used to draw lines. PENDN draws lines and PENUP positions the pen. The statement

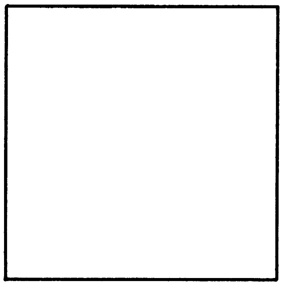
```
CALL PENUP(X,Y)
```

moves the pen (without drawing anything) from its current position to the coordinate (X,Y). X and Y must be REAL variables or constants and specify the new position in inches relative to the lower-left corner of the paper. The statement

```
CALL PENDN(X,Y)
```

draws a straight line from the current pen position to the point (X,Y). To draw a figure such as a square, PENUP must be called to move the pen to the first corner, and then PENDN must be called for each successive edge.

```
CALL PENUP(2.,2.)
CALL PENDN(2.,3.5)
CALL PENDN(3.5,3.5)
CALL PENDN(3.5,2.)
CALL PENDN(2.,2.)
CALL PLTEND
STOP
END
```



For all other examples, the assumption is made that the call to PLTEND and any necessary FORTRAN statements such as STOP and END are present. They are not printed so as to save space.

Calling PENUP and PENDN is adequate for drawing simple figures. However, if there are a large number of points in the figure, it can become quite tedious. The statement

```
CALL PLINE(X,Y,N,1,0,0,0)
```

accepts two REAL arrays X and Y describing a set of coordinate points. N is an INTEGER constant or variable specifying the number of points in X and Y. The PLINE subroutine calls PENUP for the first point in X and Y and calls PENDN for each of the other points. Therefore, the statements

```
REAL X(5)/2.,2.,3.5,3.5,2./
REAL Y(5)/2.,3.5,3.5,2.,2./
CALL PLINE(X,Y,5,1,0,0,0)
```

produce the same square as the previous example.

Several other types of lines are available. PLINE can produce lines with a symbol at each point or draw scatter plots. PARROW draws lines which end with arrow heads. PCTRLN draws centerlines. PDSHLN draws dashed lines. PFLINE draws a smooth curve through a set of points. All of these are described fully in the section "PDS Subroutine Descriptions."

TEXT

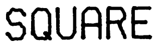
The PSYM subroutine converts a string of characters to line segments and draws them. In the statement

```
CALL PSYM(X,Y,HEIGHT,STRING,ANGLE,LENGTH,0)
```

X and Y are the coordinates of the lower-left corner of the first character. HEIGHT is the height of the characters (in inches). STRING is the character string to be drawn. It may be a constant string (enclosed in primes) or a variable or array containing the string. It may be of any type. ANGLE is the angular orientation of the string (in degrees). An angle of zero is horizontal from left to right. Angles increase in a counterclockwise direction. ANGLE is the orientation of the entire string. The orientation of the individual characters is always the same (perpendicular to ANGLE). LENGTH is the number of characters in STRING. The statement

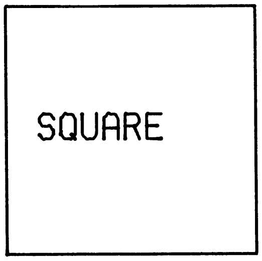
```
CALL PSYM(2.2,2.7,.15,'SQUARE',0.,6,0)
```

produces



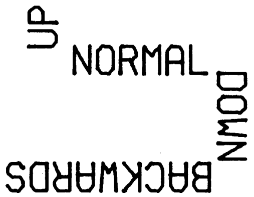
This may be used to label the previous example.

```
REAL X(5)/2.,2.,3.5,3.5,2./
REAL Y(5)/2.,3.5,3.5,2.,2./
CALL PLINE(X,Y,5,1,0,0,0)
CALL PSYM(2.2,2.7,.15,'SQUARE',0.,6,0)
```



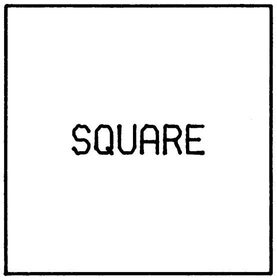
The following example demonstrates the use of the ANGLE parameter.

```
CALL PSYM(2.,3.,.15,'NORMAL',0.,6,0)
CALL PSYM(1.9,3.15,.15,'UP',90.,2,0)
CALL PSYM(2.8,3.,.15,'DOWN',-90.,4,0)
CALL PSYM(2.75,2.5,.15,'BACKWARDS',180.,9,0)
```



The PSMLEN function subroutine determines the length (in inches) of a character string as it would be drawn by PSYM. This is useful for centering titles. The title may be centered in the square previously used by using PSMLEN to determine its length as shown below.

```
REAL X(5)/2.,2.,3.5,3.5,2./
REAL Y(5)/2.,3.5,3.5,2.,2./
CALL PLINE(X,Y,5,1,0,0,0)
CLEN=PSMLEN('SQUARE',6,.15)
CALL PSYM(2.75-CLEN/2.,2.675,.15,'SQUARE',0.,6,0)
```



The three arguments to PSMLEN are the same as the STRING, LENGTH, and HEIGHT parameters to PSYM.

By default, all symbols drawn by PSYM use the standard character set. This set contains all symbols normally found on a terminal or keypunch. Several alternate character sets are available. The PALPHA subroutine selects the character set to be used by PSYM. Appendix A describes the alternate character sets available. Since all subroutines in the PDS which draw symbols call PSYM, calling PALPHA to select an alternate character set affects all symbols drawn by the PDS.

NUMBERS

Labeling on a plot must often include quantities which are available to the program only as numbers instead of character strings. The PSYM subroutine cannot be called directly to draw these. There is also no convenient facility in the FORTRAN language to convert numbers to character strings. The PNUMBR subroutine is provided to draw numbers. It converts a number to a character string according to a specified format and calls PSYM to draw it. Unfortunately, the format conversion routines which FORTRAN uses are not readily available to the PDS. Therefore, PNUMBR uses the IOH subroutine package to perform conversions. Formats for IOH are similar to, but not exactly the same as FORTRAN formats. The section "Introduction to IOH" in this volume describes the most commonly used IOH formats. IOH is described completely in MTS Volume 5, System Services.

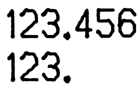
The format of a call to PNUMBR is

```
          CALL PNUMBR(X,Y,HEIGHT,FNUMB,ANGLE,FMT,0)
```

where X, Y, HEIGHT, and ANGLE are the same as the corresponding arguments to PSYM. FNUMB is a variable or constant containing the number to draw. It may be either integer or floating-point, depending on the format specified. FMT is a character string (usually a constant enclosed in primes) which is an IOH format. The statements

```
          Z=123.456
          CALL PNUMBR(2.5,3.,.15,Z,0.,'F3.3*',0)
          CALL PNUMBR(2.5,2.75,.15,Z,0.,'F3.0*',0)
```

produce



```
123.456
123.
```

As can be seen from the example, an IOH format is terminated with an asterisk, instead of being enclosed in parentheses as are FORTRAN formats. The F format term for IOH is similar to the F format term for FORTRAN. However, for IOH the first number in the term specifies the number of places to the left of the decimal instead of the field width. The second number specifies the number of places to the right of the decimal as usual. For a more complete description of IOH formats, including the use of integer formats, see the section "Introduction to IOH" in this volume.

RELATIVE COORDINATES

So far, all coordinate values used have been expressed in inches relative to the lower-left corner of the paper. Coordinates using this system are referenced in the rest of this volume as absolute. These are the coordinates stored in the plot description file produced by the PDS.

In many (if not most) cases, the data available are not so conveniently specified. It is possible, of course, to write instructions in the main program to scale the data, but the PDS provides a flexible facility to do this.

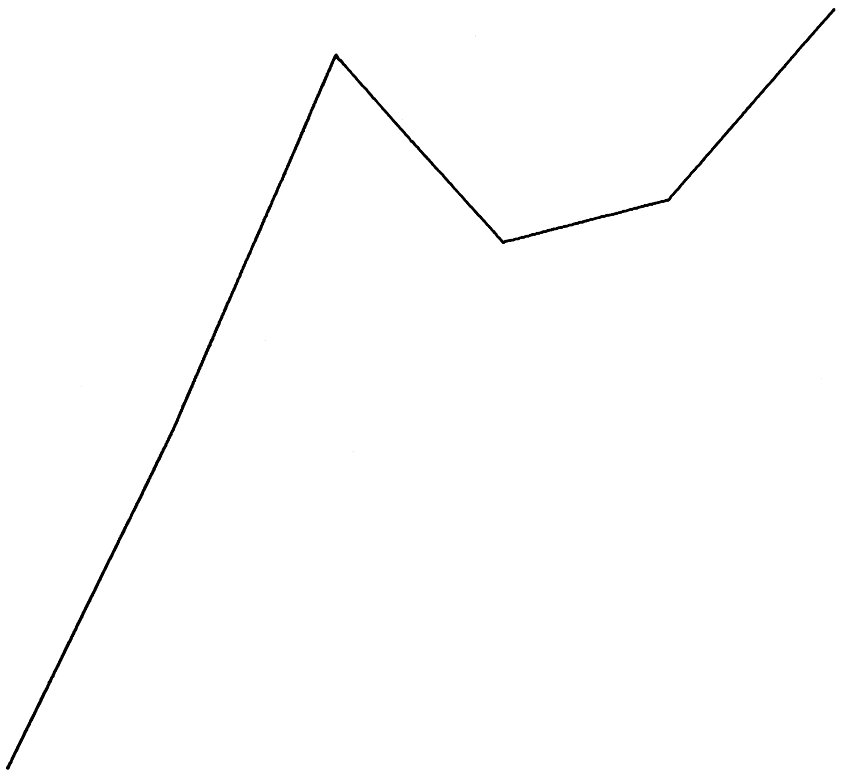
Suppose that some data are gathered from a timed experiment. The time values are in milliseconds and range from 0 to 500. Time is to be drawn in the X direction. The values measured during this time range from 0 to 10 and are to be drawn in the Y direction. If PLINE is called to draw the curve without performing any scaling, the plot is 10 inches high and 500 inches long. A plot this size might make nice wallpaper but is hardly useful for anything else. A better size is perhaps five inches square. To obtain this, the X values must be divided by 100 and the Y values must be divided by 2. This can be done by calling the PLTOFS subroutine. The general form is

```
                 CALL PLTOFS(0.,XFACT,0.,YFACT,0.,0.)
```

where XFACT is the X scale factor and YFACT is the Y scale factor. Using data as described above

```
               REAL TIME(6)/0.,100.,200.,300.,400.,500./
               REAL Y(6)/0.,4.,8.5,6.25,6.75,9./
               CALL PLTOFS(0.,100.,0.,2.,0.,0.)
               CALL PLINE(TIME,Y,6,1,0,0,1)
```

produces



The values passed to PLINE are divided by the scale factors specified in the call to PLTOFS--not multiplied by them. The scale factors specify data units per inch. In this example, the time scale factor is 100 milliseconds/inch. This means that some scale factors produce the opposite effect from what is expected. For instance, a scale factor of 0.5 inches makes the plot twice as large--not half as large.

Note that the value of the last argument to PLINE above is no longer zero. This parameter, referenced as MODE in the subroutine descriptions, controls the use of the parameters passed to PLTOFS. The MODE parameter is present in many of the PDS subroutines. It is usually the last parameter. If the value of MODE is zero, the coordinates are assumed to be absolute and are not scaled. If the value of MODE is anything else, the coordinates are assumed to be _relative_ and are scaled by the values passed to PLTOFS. (Relative coordinates are any coordinates which are not absolute.) The exact value of MODE does not matter; it is only tested for zero or nonzero. The presence of MODE allows a plot to be drawn using partly absolute and partly relative coordinates without making repeated calls to PLTOFS to change the scale factors.

The last parameter to PSYM and PNUMBR is the MODE parameter. (PNUMBR actually has an optional parameter following MODE which is not often used; it is described in the PNUMBR subroutine description.)

The use of the XFACT and YFACT parameters to PLTOFS is not sufficient for all data sets. Suppose that the range of values used previously is changed. If time (X) ranges from 500 to 1000 and Y ranges from -10 to 0, no values for XFACT and YFACT can produce a reasonable plot. Using 100 and 2 for XFACT and YFACT still scales the values so that the plot is the correct size, but it is not on the paper since the values for Y are negative. Two more parameters of PLTOFS are used to correct this problem.

```
               CALL PLTOFS(XMIN,XFACT,YMIN,YFACT,0.,0.)
```

The XMIN and YMIN parameters are set to the minimum values for the X and Y data, respectively. XFACT and YFACT are set as before. The specified values for XMIN and YMIN are subtracted from the X and Y data values before being scaled by XFACT and YFACT. This has the effect of moving the data set so that it always has a minimum value of zero. Using the new data set, the statements

```
               REAL TIME(6)/500.,600.,700.,800.,900.,1000./
               REAL Y(6)/-10.,-6.,-1.5,-3.75,-3.25,-1./
               CALL PLTOFS(500.,100.,-10.,2.,0.,0.)
               CALL PLINE(TIME,Y,6,1,0,0,1)
```

produce exactly the same plot as the previous example.

Using the specifications above, the relative coordinate (XMIN,YMIN) is drawn at the absolute coordinate (0.,0.). This means that the plot is drawn against the bottom and left edges of the paper. This is not always desirable if a title or axis is to be drawn below or to the left of the curve. The other two parameters to PLTOFS resolve this problem.

```
              CALL PLTOFS(XMIN,XFACT,YMIN,YFACT,XORG,YORG)
```

The values used for XORG and YORG specify the amount (in inches) which the relative coordinates are to be shifted. Using values of 3.0 for XORG and YORG shifts the plot up and to the right three inches. The statements

```
               REAL TIME(6)/500.,600.,700.,800.,900.,1000./
               REAL Y(6)/-10.,-6.,-1.5,-3.75,-3.25,-1./
               CALL PLTOFS(500.,100.,-10.,2.,3.,3.)
               CALL PLINE(TIME,Y,6,1,0,0,1)
```

produce the same plot as before, shifted up and to the right three inches. This has the effect of drawing the relative coordinate (XMIN,YMIN) at the absolute coordinate (XORG,YORG) instead of at (0.,0.). Appendix F describes the equations used to transform relative to absolute coordinates.

The PENUP and PENDN subroutines described previously use only absolute coordinates. There are two additional subroutines, PENUPS and PENDNS, which correspond exactly to PENUP and PENDN and use only relative coordinates. There is a fifth subroutine, PLOTCC, which combines the functions of all four of the others by adding a third argument. PLOTCC may be more convenient to use in some cases. PLOTCC is called by

```
CALL PLOTCC(X,Y,IC)
```

The function performed by PLOTCC depends on the value of IC and is given by the table below.

IC Value	Function Performed
2	PENDN
3	PENUP
12	PENDNS
13	PENUPS

DRAWING AXES

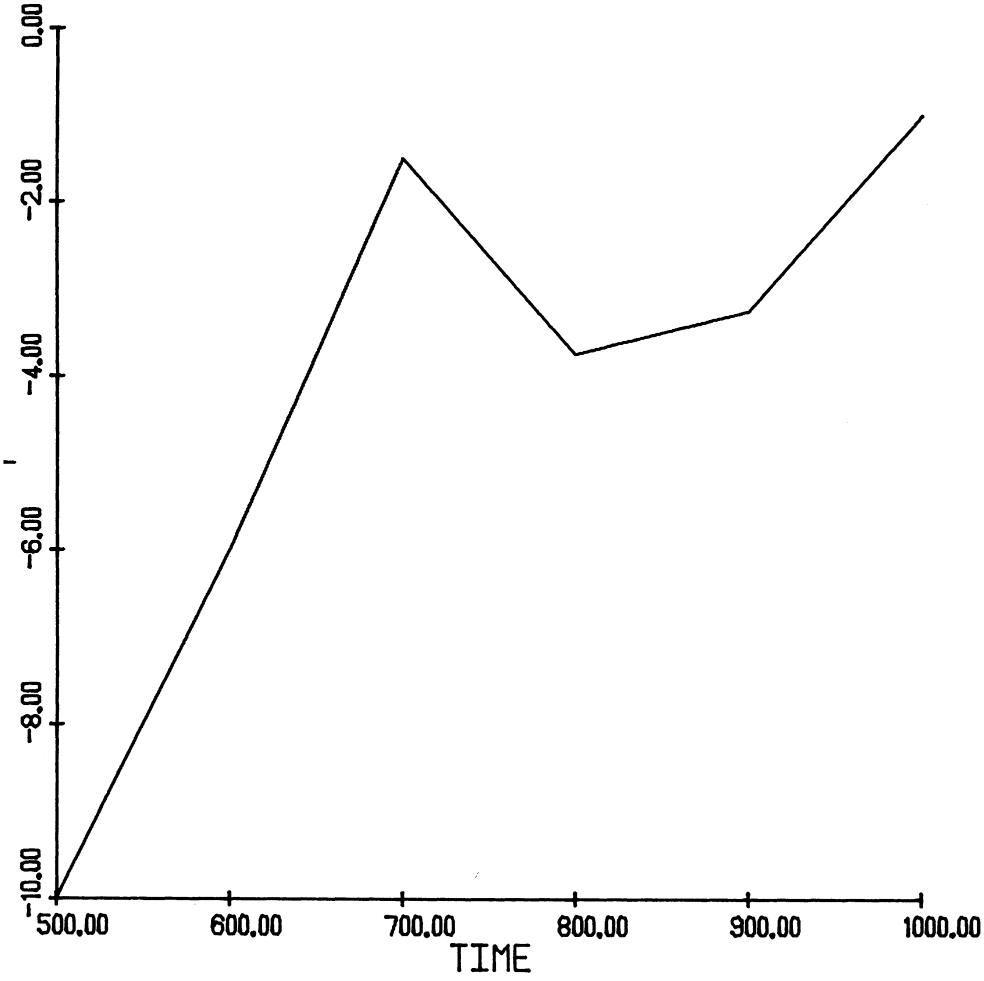
When using relative coordinates, it is often desirable to label the drawing by adding axes. The subroutine PAXIS draws an axis. To produce the normal X and Y axes, PAXIS must be called twice. PAXIS is called by

```
CALL PAXIS(X,Y,TITLE,NCHAR,AXLTH,ANGLE,XMIN,XFACT,1.)
```

X and Y are the absolute coordinates of the initial end of the axis. This is usually the left end for an X axis or the lower end for a Y axis. The values of X and Y are normally the same as the values of XORG and YORG used in PLTOFS. TITLE and NCHAR specify a title. TITLE is either a character string constant (enclosed in primes) or a variable or array containing the title. The value of NCHAR serves a dual purpose. The magnitude of NCHAR is the number of characters in TITLE. The sign of NCHAR indicates the orientation of the lettering for the title and numbering. If NCHAR is negative, the labeling is drawn on the clockwise side of the axis; otherwise it is drawn on the counterclockwise side. In most cases, the labeling for an X axis is placed on the clockwise side while the labeling for a Y axis is placed on the counterclockwise side. This keeps the labeling outside of the area in which the curve is drawn. AXLTH is the length of the axis in inches. ANGLE is the angle at which the axis is drawn. It is in degrees and increases in a counterclockwise direction. Zero is horizontal from left to right and 90 is vertical from bottom to top. XMIN and XFACT are used to label the axis. They are the same as the PLTOFS parameters XMIN and XFACT (for an X axis) or YMIN and YFACT (for a Y axis).

Combining a pair of axes with the example from the previous section produces a complete plot.

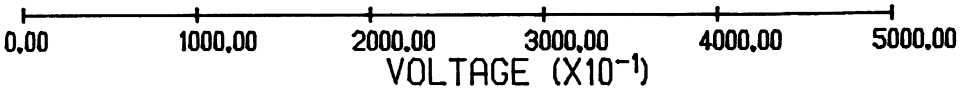
```
REAL TIME(6)/500.,600.,700.,800.,900.,1000./
REAL Y(6)/-10.,-6.,-1.5,-3.75,-3.25,-1./
CALL PLTOFS(500.,100.,-10.,2.,3.,3.)
CALL PLINE(TIME,Y,6,1,0,0,1)
CALL PAXIS(3.,3.,'TIME',-4,5.,0.,500.,100.,1.)
CALL PAXIS(3.,3.,'Y',1,5.,90.,-10.,2.,1.)
```



The above example shows the conventional orientation and placement of X and Y axes. It is certainly not necessary to use these conventions. The axes may be placed in any position, such as to the right and above the curve. If the plot is large, it may be useful to draw duplicate axes on both sides of the curve. This may be done by changing the values of X and Y on the call to PAXIS.

If the numbers have too many digits small to be drawn by the axis between the tick marks, PAXIS scales them by a power of 10 and draws the scaled numbers. It also draws the scale factor beside the title as shown below. The scale factor is the number to multiply the data values by to get the values drawn by the axis. This scaling does not affect the scaling of the data.

```
          CALL PAXIS(2.,2.,'VOLTAGE',-7,5.,0.,0.,10000.,1.)
```



This automatic scaling may be suppressed by providing an alternate numbering format to fit the data values. If an alternate format is specified (via the PAXFRM subroutine), the scaling is not done. The size of the lettering and numbering, as well as its presence or absence is also controllable. See the descriptions of the PAXIS, PAXFRM, PAXTIC, PAXTTL, and PAXVAL subroutines for information.

AUTOMATIC SCALING

Determining the minimum and scale factor used by PLTOFS and PAXIS is relatively easy if the range of the data is known beforehand. If the range is not known, it may be easily calculated from the data values. Given the range, it is trivial to calculate the scale factor. The formula for such a calculation is (MAX-MIN)/LENGTH where MAX and MIN are the maximum and minimum from the data and LENGTH is the distance over which the values are drawn (usually the axis length). Blind use of this formula, however, can lead to very poor graphs. It is possible to end up with numbers on axis tick marks in a sequence like 0.37, 1.59, 2.81, 4.03, ... making it very difficult to interpolate. The PSCALE subroutine calculates the range of the values in an array passed to it and attempts to produce "nice" values for the minimum and scale factor.

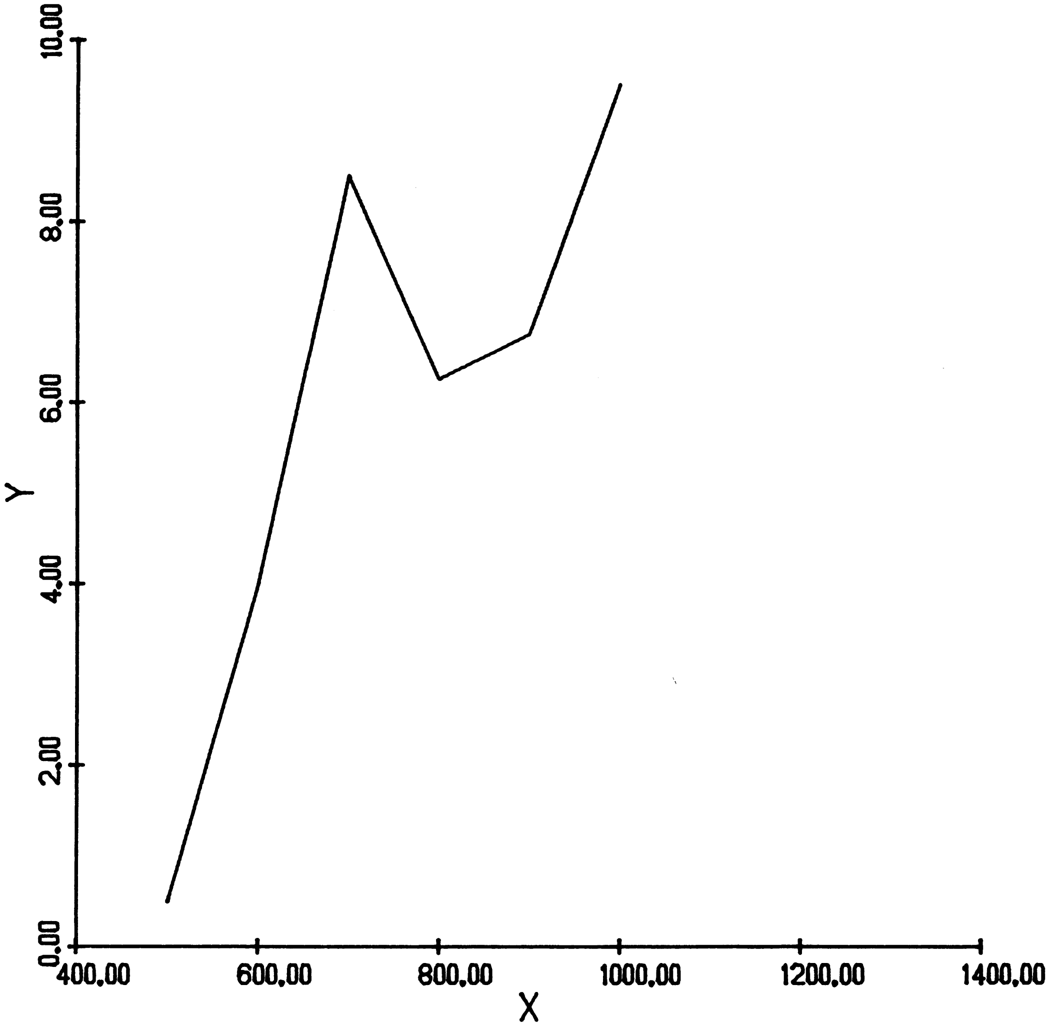
The general form of a call to PSCALE is

```
                CALL PSCALE(AXLTH,1.,XMIN,XFACT,X,N,1)
```

X is an array containing the data values to be drawn. N is the number of elements in X. AXLTH is the distance (in inches) over which the data are to be drawn. This is usually the same as the axis length given to PAXIS. XMIN and XFACT are calculated by PSCALE. They are set to the adjusted minimum and scale factor. These may be used in a call to PLTOFS or PAXIS. Since they are changed, they must be variables.

If the ranges of both the X and Y variables are unknown, PSCALE may be called twice; once to find the X scale factors and once to find the Y scale factors. The example below demonstrates the use of PSCALE.

```
REAL X(6)/500.,600.,700.,800.,900.,1000./
REAL Y(6)/.5,4.,8.5,6.25,6.75,9.5/
CALL PSCALE(5.,1.,XMIN,XF,X,6,1)
CALL PSCALE(5.,1.,YMIN,YF,Y,6,1)
CALL PLTOFS(XMIN,XF,YMIN,YF,2.,2.)
CALL PLINE(X,Y,6,1,0,0,1)
CALL PAXIS(2.,2.,'X',-1,5.,0.,XMIN,XF,1.)
CALL PAXIS(2.,2.,'Y',1,5.,90.,YMIN,YF,1.)
```



MULTIPLE DATA SETS

Often it is desirable to plot more than one variable against another variable on the same axes. This allows easy comparison of the relationships. Two problems arise when doing this. One is determining a single set of scale factors to use for all of the lines. The other is distinguishing between the lines on the plot.

The PSCALE subroutine allows more than one array of coordinates to be used. It calculates a single minimum and scale factor based on all of the arrays. This is done by

```
CALL PSCALE(AXLTH,1.,XMIN,XFACT,X1,N1,1,X2,N2,1,...)
```

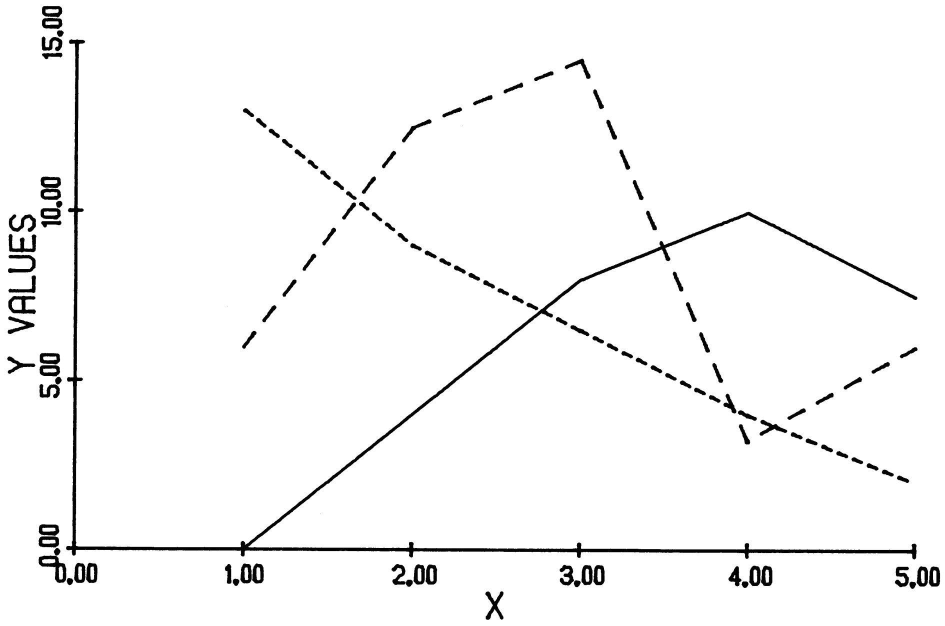
AXLTH, XMIN, and XFACT are as previously described. X1 is an array of data to be scaled. N1 is the number of elements in X1. X2 and N2 similarly describe a second set of points. Any number of such sets may be passed to PSCALE at once. An example below shows PSCALE used in this manner.

One method used to denote different lines on a graph is to use dashed lines with different dash lengths. The PDSHLN subroutine draws dashed lines

```
CALL PDSHLN(X,Y,N,1,DSHLTH,MODE)
```

The X, Y, N, and MODE parameters are the same as the corresponding parameters for PLINE. DSHLTH is the length of the dashes (in inches). The example below illustrates the use of PSCALE to scale multiple data sets and the use of PDSHLN.

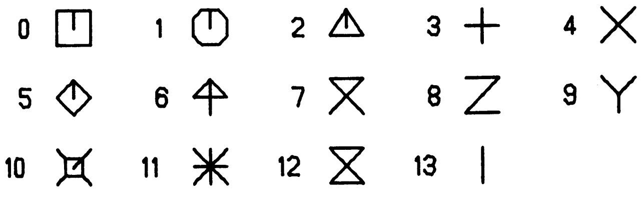
```
REAL X(5)/1.,2.,3.,4.,5./
REAL Y1(5)/0.,4.,8.,10.,7.5/
REAL Y2(5)/6.,12.5,14.5,3.25,6./
REAL Y3(5)/13.,9.,6.5,4.,2./
CALL PSCALE(3.,1.,YMIN,YF,Y1,5,1,Y2,5,1,Y3,5,1)
CALL PLTOFS(0.,1.,YMIN,YF,2.,2.)
CALL PLINE(X,Y1,5,1,0,0,1)
CALL PDSHLN(X,Y2,5,1,.1,1)
CALL PDSHLN(X,Y3,5,1,.05,1)
CALL PAXIS(2.,2.,'X',-1,5.,0.,0.,1.,1.)
CALL PAXIS(2.,2.,'Y VALUES',8,3.,90.,YMIN,YF,1.)
```



Another method is to draw a small symbol at intervals along the lines. The PLINE subroutine can do this.

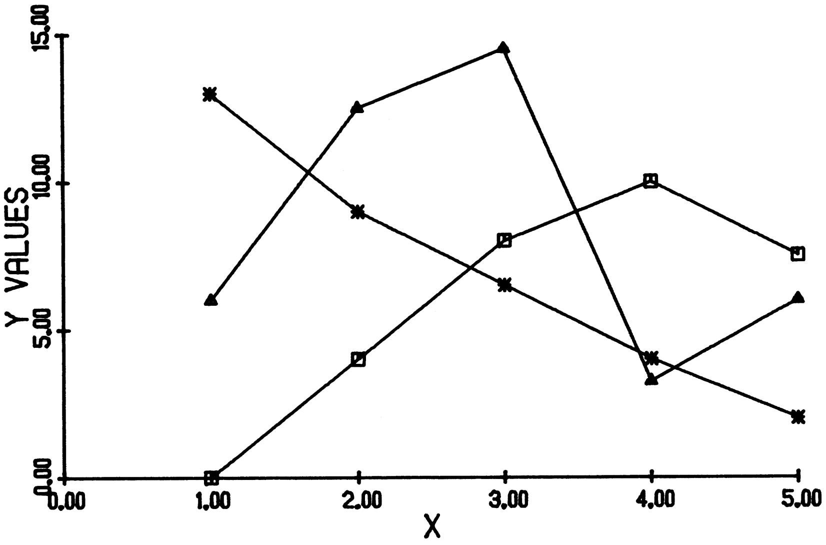
```
CALL PLINE(X,Y,N,1,ITYP,ISYM,MODE)
```

X, Y, N, and MODE have been described previously. ISYM is a number between zero and thirteen which specifies the symbol to be used. The available symbols are shown below.



If ITYP is 0, a solid line is drawn and the value for ISYM is ignored (it must still be present). If ITYP is 1, a solid line is drawn with the symbol centered at the coordinates of each data point. If ITYP is -1, a symbol is drawn at each point, but no line is drawn. This is useful for drawing scatter diagrams. The magnitude of ITYP controls the spacing of the symbols. If ITYP is 2, a line is drawn with a symbol at every second data point, and so on.

```
REAL X(5)/1.,2.,3.,4.,5./
REAL Y1(5)/0.,4.,8.,10.,7.5/
REAL Y2(5)/6.,12.5,14.5,3.25,6./
REAL Y3(5)/13.,9.,6.5,4.,2./
CALL PSCALE(3.,1.,YMIN,YF,Y1,5,1,Y2,5,1,Y3,5,1)
CALL PLTOFS(0.,1.,YMIN,YF,2.,2.)
CALL PLINE(X,Y1,5,1,1,0,1)
CALL PLINE(X,Y2,5,1,1,2,1)
CALL PLINE(X,Y3,5,1,1,11,1)
CALL PAXIS(2.,2.,'X',-1,5.,0.,0.,1.,1.)
CALL PAXIS(2.,2.,'Y VALUES',8,3.,90.,YMIN,YF,1.)
```

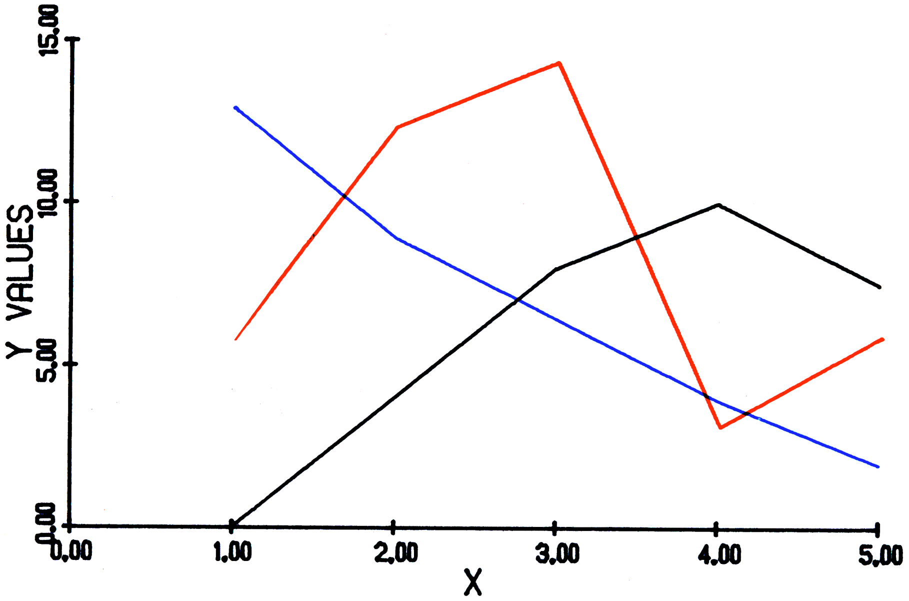


Color may also be used to differentiate lines. The PENCHG subroutine selects the pen to be used for drawing. After calling PENCHG, everything is drawn with the specified pen until the next call to PENCHG or PLTEND. The calling sequence for PENCHG is

```
CALL PENCHG(COLOR)
```

where COLOR is a four-character string specifying the color of the ink. It may be either a constant enclosed in primes or a variable containing the string. If the color name is longer than four characters, only the first four need be used. If the color name is less than four characters, it must be padded on the right with blanks. The default color is black. Appendix E lists the available pens.

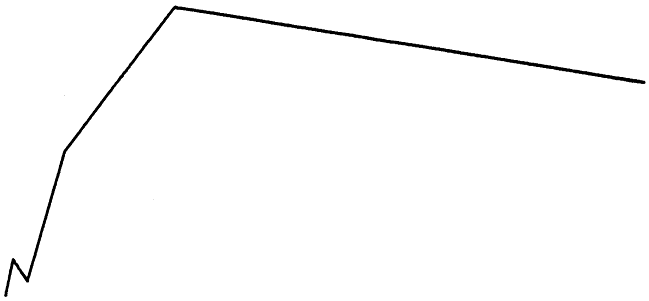
```
REAL X(5)/1.,2.,3.,4.,5./
REAL Y1(5)/0.,4.,8.,10.,7.5/
REAL Y2(5)/6.,12.5,14.5,3.25,6./
REAL Y3(5)/13.,9.,6.5,4.,2./
CALL PSCALE(3.,1.,YMIN,YF,Y1,5,1,Y2,5,1,Y3,5,1)
CALL PLTOFS(0.,1.,YMIN,YF,2.,2.)
CALL PLINE(X,Y1,5,1,0,0,1)
CALL PENCHG('RED ')
CALL PLINE(X,Y2,5,1,0,0,1)
CALL PENCHG('BLUE')
CALL PLINE(X,Y3,5,1,0,0,1)
CALL PENCHG('BLAC')
CALL PAXIS(2.,2.,'X',-1,5.,0.,0.,1.,1.)
CALL PAXIS(2.,2.,'Y VALUES',8,3.,90.,YMIN,YF,1.)
```



LOGARITHMIC COORDINATES

If the data values to be drawn have a very wide range and are closely spaced with small values and widely spaced with large values, too much detail may be lost with normal linear scaling. In the example below, the X coordinates range from 20 to 900. It is difficult to tell what the left part of the curve should look like.

```
REAL X(6)/20.,30.,50.,100.,250.,900./
REAL Y(6)/.5,.75,.6,1.5,2.5,2./
CALL PLTOFS(0.,200.,0.,1.,3.,3.)
CALL PLINE(X,Y,6,1,0,0,1)
```



The PDS allows the X or Y coordinates (or both) to be scaled logarithmically. This type of scaling tends to stretch small values and compress large values, making it easier to see details in small regions. This is done by taking the base 10 logarithm of each coordinate. Since the logarithm function is only defined for positive values, all coordinates to be so scaled must be positive.

The PLTLOG subroutine causes the PDS to begin logarithmic scaling.

```
CALL PLTLOG(I)
```

Value of I	Result
1	X is linear, Y is logarithmic
2	X is logarithmic, Y is linear
3	X and Y are logarithmic

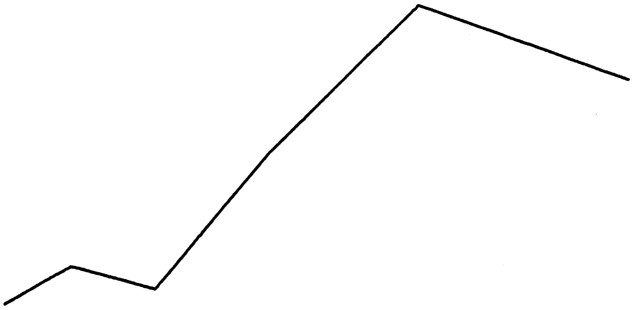
In this mode, all coordinates, whether relative or absolute, are scaled logarithmically. Because of this, only the curves should be drawn in this mode. All other things (such as axes and titles) should be drawn in linear mode. The statement

```
CALL PLTREC
```

returns to the default linear-rectangular mode.

In addition to calling PLTLOG, a call to PLTOFS is usually necessary. The values used for XORG and YORG are the same as described previously. If either axis is scaled linearly, the values for the minimum and scale factor are also computed as previously described (either by hand or by using PSCALE). For logarithmic coordinates, the units of the minimum and scale factor are different. The minimum should be set to the exponent of the largest power of 10 smaller than the data values. Using the previous example, XMIN is set to 1.0 since 10^1 is the largest power of 10 less than 20. The units for the scale factor are powers-of-ten per inch. This is found by taking the exponent of the smallest power of ten greater than the data values, subtracting the minimum value from it and dividing by the axis length. For a 5-inch axis, XFACT is 0.4 since 10^3 (1000) is the smallest power of ten greater than 900 and (3-1)/5=.4. The example below shows the use of this.

```
REAL X(6)/20.,30.,50.,100.,250.,900./
REAL Y(6)/.5,.75,.6,1.5,2.5,2./
CALL PLTOFS(1.,.4,0.,1.,3.,3.)
CALL PLTLOG(2)
CALL PLINE(X,Y,6,1,0,0,1)
CALL PLTREC
```

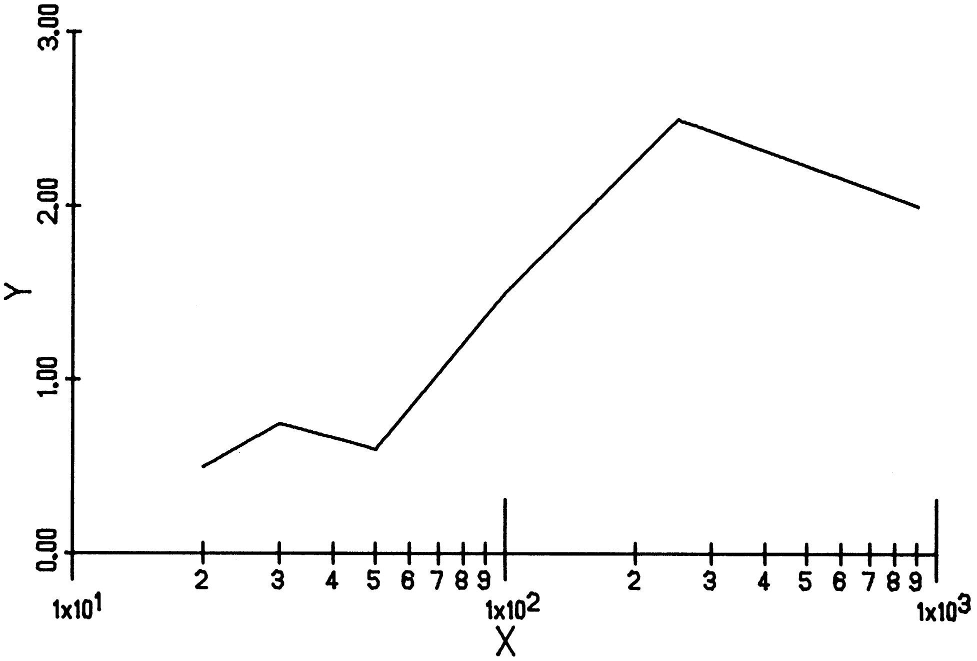


The PLGAXS subroutine draws a logarithmic axis. The general form is

```
CALL PLGAXS(X,Y,TITLE,NCHAR,AXLTH,ANGLE,XMIN,XFACT)
```

All of the arguments are the same as those to PAXIS. The only difference is that the values for XMIN and XFACT must be appropriate for logarithmic scaling. They are the same as the values used in the call to PLTOFS. The example below adds axes to the previous example. Note that PAXIS is used to draw the Y axis since the Y coordinates are scaled linearly.

```
REAL X(6)/20.,30.,50.,100.,250.,900./
REAL Y(6)/.5,.75,.6,1.5,2.5,2./
CALL PLTOFS(1.,.4,0.,1.,3.,3.)
CALL PLTLOG(2)
CALL PLINE(X,Y,6,1,0,0,1)
CALL PLTREC
CALL PLGAXS(3.,3.,'X',-1,5.,0.,1.,.4)
CALL PAXIS(3.,3.,'Y',1,3.,90.,0.,1.,1.)
```



The PLGSCL subroutine may be used to determine the minimum value and scale factor for logarithmic coordinates.

```
                  CALL PLGSCL(AXLTH,XMIN,XFACT,X,N,1)
```

The arguments correspond exactly to those of the PSCALE subroutine. (Note that PLGSCL has one less argument than PSCALE.) PLGSCL performs the same function as PSCALE, but sets XMIN and XFACT for use with logarithmic scaling. The statements

```
                  REAL X(6)/20.,30.,50.,100.,250.,900./
                  CALL PLGSCL(5.,XMIN,XFACT,X,6,1)
```

set XMIN and XFACT to 1.0 and 0.4 as used previously.

POLAR COORDINATES

The PDS also provides polar-coordinate modes. In polar-coordinate mode, the radial values replace the X coordinates and the angular values replace the Y coordinates.

Polar mode is entered in a manner similar to logarithmic mode, by calling the PLTPOL subroutine.

```
CALL PLTPOL(SWITCH)
```

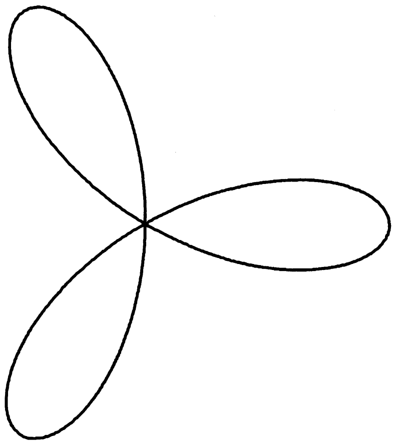
If the value of SWITCH is zero, angles for polar plotting are in radians. If SWITCH is nonzero, angles are in degrees. All angles are measured in a counterclockwise direction from the positive X axis. As before, the PLTREC subroutine returns to the default linear-rectangular mode.

The PLTOFS subroutine is slightly different for use with polar coordinates.

```
CALL PLTOFS(RMIN,RFACT,0.,1.,XORG,YORG)
```

The values for XORG and YORG are rectangular coordinates in inches (as before) and are the coordinates of the center of the coordinate system. The values for RMIN and RFACT are similar to those for linear-rectangular coordinates. The value for RMIN defines the coordinate to be drawn at the center of the coordinate system. RFACT is in units per inch, as before. This call sets up a system in which the angular coordinates correspond to the actual coordinates on the paper. Other systems are possible, but not widely used. Appendix F describes the use of such systems.

```
     DIMENSION R(361),TH(361)
     T=0
     DO 10 I=1,361
     TH(I)=T
     R(I)=3.*COS(3.*T*.0174533)
10   T=T+1
     CALL PLTOFS(0.,2.,0.,1.,6.,6.)
     CALL PLTPOL(1.)
     CALL PLINE(R,TH,361,1,0,0,1)
     CALL PLTREC
```

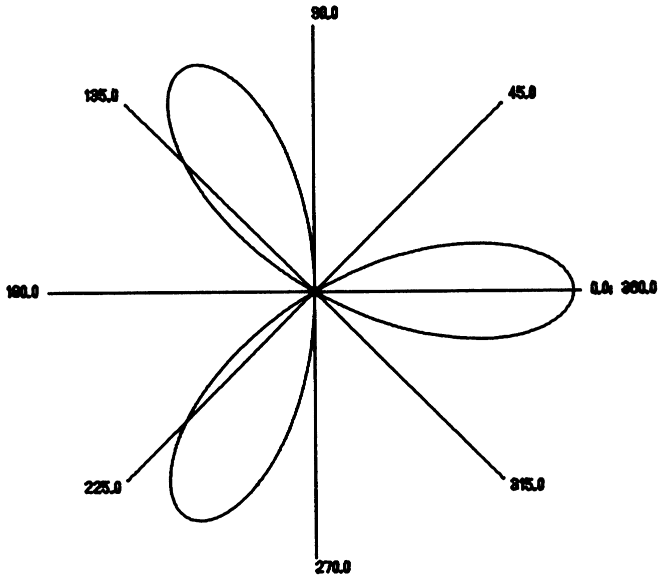


The PTHAXS subroutine draws a polar axis. A polar axis is a set of radial lines with numbers corresponding to the angles drawn at the end of each line.

```
          CALL PTHAXS(X,Y,R,THPF,DTHP,THPL,THSTR,DELTH,0,0)
```

X and Y are the linear rectangular coordinates in inches of the center of the axis. R is the length of the radial lines drawn for the axis. THPF and THPL are the angles (in degrees) of the first and last radial lines, respectively. They are usually 0. and 360. DTHP is the angle (in degrees) between the radial lines. THSTR is the number to be drawn at the first radial line. It is usually equal to THPF. DELTH is the increment to be added to THSTR to get the number to be drawn at each successive radial line. It is usually equal to DTHP. The example below adds a polar axis to the previous example.

```
                    DIMENSION R(361),TH(361)
                    T=0
                    DO 10 I=1,361
                    TH(I)=T
                    R(I)=3.*COS(3.*T*.0174533)
                10  T=T+1
                    CALL PLTOFS(0.,2.,0.,1.,6.,6.)
                    CALL PLTPOL(1.)
                    CALL PLINE(R,TH,361,1,0,0,1)
                    CALL PLTREC
                    CALL PTHAXS(6.,6.,1.5,0.,45.,360.,0.,45.,0,0)
```



It is possible to combine logarithmic and polar plotting modes by scaling the radial values logarithmically. (The angular values are always linear.) All that is necessary is to change the call to PLTPOL into a call to PLGPOL and to change the values for RMIN and RFACT to correspond to logarithmic mode. PLTPOL and PLGPOL have exactly the same argument.

Most of the capabilities of the Plot Description System have been discussed in this section, although some have only been touched upon. The subroutines in the PDS are described fully in the section "PDS Subroutine Descriptions." The section "Complete Example Programs" contains several more extensive examples of the use of the PDS.

SUBMITTING PLOTS

The public file *CCQUEUE contains a program for submitting plots. The program is invoked by either

```
(1)  $RUN *CCQUEUE PAR=request

(2)  $RUN *CCQUEUE
     request
     request
     .
     .
     $ENDFILE
```

"request" is a queuing request. The format of this request is described below. In case (1) above, a single request is processed. In case (2), requests are read from SCARDS. Plot receipt numbers and prompting messages for queueing requests are printed on SPRINT. Error comments are printed on SERCOM, and responses to error comments are read from GUSER.

QUEUEING REQUESTS

Each queueing request specifies a file or magnetic tape containing one or more plot descriptions. *CCQUEUE reads the plot descriptions, determines the cost of plotting, places the request in the system plotting queue, and charges the user for the plotting costs. At each scheduled plotting time, the system postprocessor removes the request from the queue, reads the user's file (or tape), and produces commands for the plotter.

*CCQUEUE prompts for queueing requests using the message, "ENTER PLOT REQUEST". The response may be a queueing request for a file or tape, a line consisting of the string "MTS", a line beginning with a "$", or an end-of-file indicator. If the queueing request is for a file, the request should consist of the file name, optionally followed by one or more blanks and a scale factor. Explicit concatenations are permitted, as are line-number ranges. Only permanent files may be queued--temporary files may not be queued.

The response "MTS" causes a return to MTS command mode. A response that begins with a dollar sign is treated as an MTS command that is to be executed immediately. For either response, "MTS" or an MTS command, execution of *CCQUEUE may be resumed by entering a $RESTART command (unless an MTS command causes *CCQUEUE to be unloaded). An end-of-file response terminates execution of *CCQUEUE.

If the request is for a magnetic tape, the tape should already be mounted. The request should contain first the pseudodevice name, followed by the tape ID. Parameters must be separated by one or more blanks and/or commas. There are three optional parameters. One is the scale factor, which may appear anywhere after the pseudodevice name. Another is the FILES parameter, which may also appear anywhere after the pseudodevice name. This takes the form FILES=n, where "n" is the number of files to be plotted. The default is FILES=1. Finally, the POSN parameter may appear anywhere after the tape ID. This takes the form POSN=string, where "string" specifies a file on the tape (any string legal for the POSN control command). If this parameter is not specified, *CCQUEUE begins reading at the current tape position (even if that is in the middle of a file). (At postprocessing, the tape will be mounted and positioned appropriately, even to the middle of a file.) *CCQUEUE and the postprocessor use whatever blocking is in effect at the time the queueing request is entered. *CCQUEUE does not rewind the tape after reading it. Pool tapes may not be queued.

Any plot request may include a scale factor. The scale factor will be applied to each point of the plot. This parameter takes the form SCALE=x, where "x" is a positive integer, F-type, or E-type real number.

Users can request delivery of plots to another station by specifying the DELIVERY=station parameter on the queue request, e.g., DELIVERY= NUBS. The DELIVERY parameter is effective for only one queue request. If the DELIVERY parameter is not specified, the setting of the MTS $SET DELIVERY option will be used (which defaults to the Computing Center).

Users can request "quick" plotting by specifying the ASAP parameter on the queue request. This will provide a faster than normal service. Ballpoint pens will be used instead of liquid ink and the appearance of the plot will differ; lines will be thinner and not as dark, and dots may not be drawn very well. The plotter will use only red, blue, and black ballpoint pens, and pen size changes will not be allowed. See the pubic file *PLOTTIMES for the schedule of ASAP plotting service.

ERROR PROCESSING

If *CCQUEUE encounters an error in the user's plot description file, it prints a comment describing the error, together with the file name and line number where the error was discovered. If an error is encountered in batch mode, reading of the current plot description stops, the plot is not queued, and *CCQUEUE begins processing the next request, if any. In conversational mode, the user is asked if he wishes *CCQUEUE to continue. The legal responses are YES (or OK) to continue processing (taking the error-recovery action described below), and NO, to stop processing at this point (this does not abort the plot). The user may also enter MTS or an MTS command preceded by a "$", in which case control will be returned to MTS. (The user may restart using $RES, and will again be prompted for a response.) Error-recovery actions are

Error	Action Taken by *CCQUEUE
Record not a plot record	Record ignored
Actual record length not equal to expected length	Record ignored
Missing PBGN	PBGN (logically) inserted with default normalizing factor
Invalid PSYM record	Record ignored
Invalid PPEN record	Record ignored
Missing PEND	PEND (logically) inserted

If the user indicates that processing should continue after an error is encountered, and then queues the plot, the same error recovery action will be taken by the postprocessor. A line beginning $CONTINUE WITH ... will be treated as an invalid plot line. Implicit concatenation may not be used in plot description files (or tapes).

QUEUEING

After *CCQUEUE finishes reading the plot description (because it has reached the end or the user has told it not to continue after an error), it prints the number of plots (blocks), the plotting time required, the amount of paper required, and the cost. In conversational mode, the user is asked whether the plot should actually be queued. If he replies NO, the plot is aborted and he is prompted for a new plot request (unless the plot request was taken from the PAR field). If he replies YES, or if *CCQUEUE is being executed in batch mode, the plot request is placed in the queue, he is charged for the plot, and a receipt number is printed (this receipt number must be used to obtain the plot at the output window later). In addition, if the plot description was contained in one or more disk files, *CCQUEUE will, if necessary, permit the queued files so that the postprocessor, which is a program with PKEY=*CCQUEUE, may access the plot descriptions. *CCQUEUE will establish READ access for the file to the postprocessor if necessary. (Specifically, the file will be permitted "READ PKEY=*CCQUEUE" unless the postprocessor already has read access. See the description of the $PERMIT command in MTS Volume 1, The Michigan Terminal System.) A message is printed for each file so permitted. If *CCQUEUE cannot determine the permit status of a file, a message is printed, and the user must be sure that the file is permitted before the plot request is postprocessed. See the section below on postprocessing.

The plot will be aborted if it is null, if the user's plotting time limit (local, global, or ID) has been exceeded, if the user indicated abort after an attention interrupt (see below), or if reading stopped on an error while in batch mode.

CANCELATION

A queued plot may be canceled by entering CANCEL nnnnnn where "nnnnnn" is the receipt number, whenever *CCQUEUE is prompting for a plot request. A plot can be canceled only by the signon ID which queued it. If the plot specified is still in the queue (i.e., it has not been postprocessed), it will be canceled and a message to that effect will be printed. Plotting charges will be rebated automatically at a later time.

ATTENTION INTERRUPT PROCESSING

If the user issues an attention interrupt, processing of the current plot request is suspended. The subsequent action taken by *CCQUEUE is as follows: (1) if it was parsing a plot request, the plot request is discarded and the user is prompted to enter a new request; or (2) *CCQUEUE will print "READING:" or "QUEUEING:", depending upon whether it was in the process of reading the plot description or queueing the plot. In either case, if the user enters a null line or an end-of-file, *CCQUEUE will resume processing the current request. If the user enters "MTS" or an MTS command beginning with a "$", *CCQUEUE will return to MTS command mode; if a command was given, it will be immediately executed by MTS. If the user enters anything else, the plot request will be aborted.

If the user issues a second attention interrupt while *CCQUEUE is processing the first interrupt, an immediate return is made to MTS command mode. The user may subsequently reenter *CCQUEUE via the $RESTART command.

POSTPROCESSING

After a plot request has been queued, the user’s plot description must be available for reading by the system postprocessor at the scheduled plotting time. If the plot description is in a file, it must be permitted, and it must not be locked at any level higher than READ. If the file is not permitted, the plot will be canceled; if it is locked, the plot request will remain queued until the next scheduled plot time at which it is accessible. If the plot description is on a tape, the tape must be available for mounting (e.g., not already mounted); otherwise, the plot will remain queued until the tape is available at a scheduled postprocessing time. If the tape ID is incorrect, the plot will be canceled.

EXAMPLES

In the following examples, terminal output appears in uppercase, and user input in lowercase.

In the first example the plots contained in the file PLOTFILE are to be queued, and then the plots contained in two files, PLOTS and MOREPLOTS, are to be queued. First, the two plots in PLOTFILE are queued. At line 2 in MOREPLOTS, *CCQUEUE discovers that a PBGN record seems to be missing, and asks the user if it may assume a PBGN record with a default normalizing factor. The user OKs this, so *CCQUEUE continues reading the plot file. At line 2.5, *CCQUEUE discovers a PPEN record with an invalid color, and asks the user whether it may ignore the line. The user invokes the MTS file editor to examine the line, and then tells *CCQUEUE not to continue. The user then OKs queueing of all plots up to line 2.5 of MOREPLOTS.

```
#$r *ccqueue
 EXECUTION BEGINS
 ENTER PLOT REQUEST:
 plotfile
   2 PLOTS; PLOTTING REQUIRES    94 SEC. AND    19 IN.;     $.31
 PEN WAS UP  34% OF THE TIME.
 OK?
 ok
 PLOT ASSIGNED RECEIPT # 523989.
 ENTER PLOT REQUEST:
 plots+moreplots
 MISSING PBGN RECORD.
 DISCOVERED AT LINE      2.000 IN MOREPLOTS
 OK TO USE DEFAULT?
 y
 INVALID PEN TYPE.
 DISCOVERED AT LINE      2.500 IN MOREPLOTS
 OK TO IGNORE?
 $edit moreplots
#$EDIT MOREPLOTS
:p 2.5
:     2.5    PPEN??GREE
:stop
#$res
 OK TO IGNORE?
 n
 PLOT UP TO LAST ERROR CAN BE QUEUED.
   2 PLOTS; PLOTTING REQUIRES  47 SEC.  AND 9 IN.; $.17
 PEN WAS UP  32% OF THE TIME.
 OK?
 ok
 PLOT ASSIGNED RECEIPT # 516064.
 .
 .
 .
```

In the second example, the user does not have appropriate access to XXXX:PLOTS for *CCQUEUE to determine its permit status. Note that the user must make sure that the file is permitted before the next postprocessing time.

```
   #$r *ccqueue
|   EXECUTION BEGINS
    ENTER PLOT REQUEST:
    xxxx:plots
|     1 PLOTS; PLOTTING REQUIRES 35 SEC.  AND 7 IN.; $.13
|   PEN WAS UP  39% OF THE TIME.
    OK?
    y
    **PERMIT STATUS OF "XXXX:PLOTS      " UNKNOWN.
    PLOT ASSIGNED RECEIPT # 516068.
    .
    .
```

In the third example, the file whose data set name (DSN) is PLOTS and the file following it on tape *I* are to be queued. The user allows *CCQUEUE to position the tape. These two files together contain three plots. *CCQUEUE encounters two errors, and the user tells it to continue after each.

```
   #$r *ccqueue
|   EXECUTION BEGINS
    ENTER PLOT REQUEST:
    *I* 'CALCOMP PLOTS' p=PLOTS files=2
    MISSING PBGN RECORD.
    DISCOVERED AT LINE      14.000 IN *I*
    OK TO USE DEFAULT?
    y
    INVALID PEN TYPE.
    DISCOVERED AT LINE      78.000 IN *I*
    OK TO IGNORE?
    y
|     3 PLOTS; PLOTTING REQUIRES 117 SEC.  AND 27 IN.; $.47
|   PEN WAS UP  35% OF THE TIME.
    OK?
    y
    PLOT ASSIGNED RECEIPT # 516070.
    .
    .
    .
```

PREVIEWING PLOTS ON GRAPHICS TERMINALS

The *PLOTSEE program is a graphics program that displays plots from a plot file on a graphics terminal, providing a simple and fast way to check plot files for accuracy before submitting them to the plot queue. A plot file is used on MTS to hold output for a picture that is to be plotted on the CalComp plotters. The *IG, *TELLAGRAF, and *PLOTSYS graphics packages can all produce plot files.

A plot file contains encoded instructions for drawing one or more plots on the CalComp plotter. These encoded instructions need to be stored so a program can be run which will interpret them and have the CalComp plotter draw the picture or pictures represented in the plot file. The program that submits plot files to be plotted is *CCQUEUE, which is described in this volume.

*PLOTSEE can be used to serve a number of functions, from previewing a plot before spending the time and money to get the actual plot, to manipulating several plots at one time to form a composite plot. *PLOTSEE's enlargement facility allows you to see a detailed view of a section of a plot. *PLOTSEE also allows you to move, scale, rotate and combine plots into a single picture, to which lines, text and other features may be added. Any plot, enlargement, or any picture which is a combination of plots and enlargements that *PLOTSEE is able to produce can be put into a new plot file for later display or plotting. If a remote plotter is attached to a terminal, *PLOTSEE can be told to draw the picture on either the terminal screen or the remote plotter.

The *PLOTSEE program is invoked by the command

```
$RUN *PLOTSEE
```

Commands are then issued to the program to control the previewing of the plots. A summary of the *PLOTSEE command language is given below. The complete description of the *PLOTSEE program is given in Computing Center Memo 456, *PLOTSEE.

ACTIVATE number	Sets the current picture.
AUXPLOT plotter	Plots the screen image on a remote plotter.
BLOWUP	Enlarges a section of the current plot.
BOX	Draws a box around the current picture.
COLOR color	Sets the color of the current picture.

CONTINUE	Continues with the next frame.
DEFAULT string	Changes default command sequence.
DELETE	Deletes the current picture from the screen.
DESTROY	Destroys unnamed pictures on a DELETE.
DISPLAY picture	Displays only the indicated picture.
DOWN amount	See the DY command.
DRAW	Updates the screen.
DUMP	Prints information about the data structure.
DX amount	Displaces the current picture horizontally.
DY amount	Displaces the current picture vertically.
EDIT plotfile	Changes the plot input file to plot file.
ENGLISH	Specifies that units should be inches.
EXPLAIN command	Explains the use of a command.
FONT fontname	Sets the font to be used by the TEXT command.
FTNCMD command	Issues a command to the FORTRAN I/O monitor.
GET name	Makes a picture the current picture.
GLOBAL	Makes the screen image the current picture.
HELP	Assists with the running of the program.
IGCTRL switch	Turns KEEP switch on and off.
INVISIBLE	Makes the current picture invisible.
KEEP	Prevents DELETE from destroying pictures.
LEFT amount	See the DX command.
LINE	Draws a line on the screen.
LIST	Prints a list of the pictures currently defined.
MESSAGES	Prints information about the current picture.
METRIC	Makes all units metric.

MOVE	Moves the current picture to a new position.
MTS	Returns to MTS command mode.
NAME name	Assigns a name to the current picture.
NEXT	Displays the next picture.
NOPROMPT	Turns off prompting.
PEN number	Sets the line type or color.
PICK	Makes a displayed picture current.
PLOT	Puts the screen image on a plotfile.
POLYGON sides	Draws a polygon of "sides" sides.
PREFIX string	Specifies a command string to be executed when a CONTINUE command is executed.
PRINT string	Prints "string" on SPRINT.
PROMPT	Enables prompting.
REDRAW	Redraws the screen.
RIGHT amount	See the DX command.
ROTATE degrees	Rotates the current picture.
SCALE factor	Scales the current picture.
SEGMENT	Picks part of a picture as the current picture.
SELECT	Allows previewing plots in any order.
STOP	Terminates the run.
UNDO	Undoes all the transformations on the current picture.
UP amount	See the DY command.
VIEWPORT picture	Puts the indicated picture in a viewport.
VISIBLE	Makes the current picture visible.
WINDOW	Displays a window of the current picture.
$MTS command	Executes the MTS command.

INTRODUCTION TO IOH

IOH is a collection of subroutines which use a format to convert data, e.g., numbers, from internal to external form or vice versa. As it is used here, IOH does not read or write data, it simply changes its form. This section describes only a few of the capabilities of IOH. A complete description may be found in MTS Volume 14, 360/370 Assemblers in MTS.

Several of the subroutines in the Plot Description System draw numbers. They all do so by using IOH to convert the number to a character string which is then drawn by the PSYM subroutine. To perform this conversion IOH requires a format. Several of the subroutines in the Plot Description System require IOH formats as arguments. These formats determine the appearance of the numbers drawn by the subroutine.

An IOH format is a character string. Therefore to be specified as an argument in a FORTRAN program it must either be contained in a variable or array or enclosed in primes or be specified as a Hollerith literal. An IOH format is very similar to a FORTRAN format. It consists of one or more format terms separated from each other by commas. The last character of a complete IOH format must be an asterisk. A format term consists of three parts. The only part which is required is the control character, which specifies the type of conversion to be performed. One or more modifier characters may precede the control character. Following the control character may be one or more numbers separated by periods which specify the field widths. The remainder of this section describes the most commonly used format terms.

INTEGER NUMBERS

The general form of the integer format term is

Iw

where "w" is the field width. The number is converted and placed right-justified in the next "w" columns of the character string. If "w" is omitted, the field width defaults to 8.

FLOATING-POINT NUMBERS

There are two format specifications for floating-point numbers. F-type specifies integers and fractions. Examples of numbers of this type are 30.12, 3.14159 and .025. The general form of an F-type format term is

Fm.n.w

where "m" is the number of digits before the decimal point, "n" is the number of digits after the decimal point and "w" is the total field width. Note that this is not the same as a FORTRAN format. The F format causes the number to be converted and placed right-justified in the next "w" columns of the character string. If "w" is omitted, it defaults to m+n+2. For example,

F5.3.10

is the same as

F5.3

The number 123.456 is converted as shown below using the indicated format.

123.456	F3.3
123.46	F3.2
123.	F3.0
123.4560	F3.4

The second way to specify floating-point numbers is by using a fraction and an exponent. Examples of numbers of this type are 30.12E+02, .314159E+01, and .025E-01. The format term used to convert E-type floating-point numbers is

Em.n.w

where each of the fields has the same meaning as the fields in the F-type specification. Because of the form of the number, the external field width specification must be large enough to include not only the number of digits before and after the decimal point but also the decimal point, the sign of the number, the E, the sign of the exponent, and the two digits of the exponent. If "w" is not given it defaults to m+n+6. The number 123.456 is converted as shown below with the indicated formats.

123.456E+00	E3.3.11
12.346E+01	E2.3.10
1.235E+02	E1.3.9
.123E+03	E0.3.8
1.235E+02	E1.3

CHARACTER LITERALS

It is often desirable to place text as well as numbers into the output. This can be done by enclosing the text to be transferred literally in either primes or quotation marks. If the literal string is initiated with a prime it must be terminated with a prime. Likewise if the string is initiated with quotation marks it must be terminated with quotation marks. It is recommended that quotation marks be used to enclose literal strings in formats since the entire format is usually enclosed in primes in a FORTRAN program.

MODIFIERS

Modifiers may be used to alter the effect of the conversion. Any number of modifiers may be specified prior to the control character in a format term. A modifier is specified in the form of a letter. A description of the most commonly used modifiers follows.

H The H modifier indicates that the conversion is to take place from a halfword. This is used only with integer format terms when the variable is declared as INTEGER*2.

J The J modifier causes the external field width to be reduced to the smallest number of characters which can accommodate the converted field without using any fill characters. This means that leading and trailing blanks are removed. The J modifier may be used with either integer or floating-point format terms.

L The L modifier indicates that the field is to be left-justified with trailing blanks instead of right-justified with leading blanks which is the default.

M The M modifier is used only with F format terms. It causes suppression of the decimal point. For example, the number 123.456 is converted as shown below with the indicated formats.

```
    123        MF3.0
 123456        MF3.3
```

COMPLETE EXAMPLE PROGRAMS

The following pages contain six complete example programs which use the PDS. Each program is followed by the plot which it produces. The programs contain comments which should make them self-explanatory. The plots produced by these programs had to be reduced in size somewhat to fit on the pages of this volume.

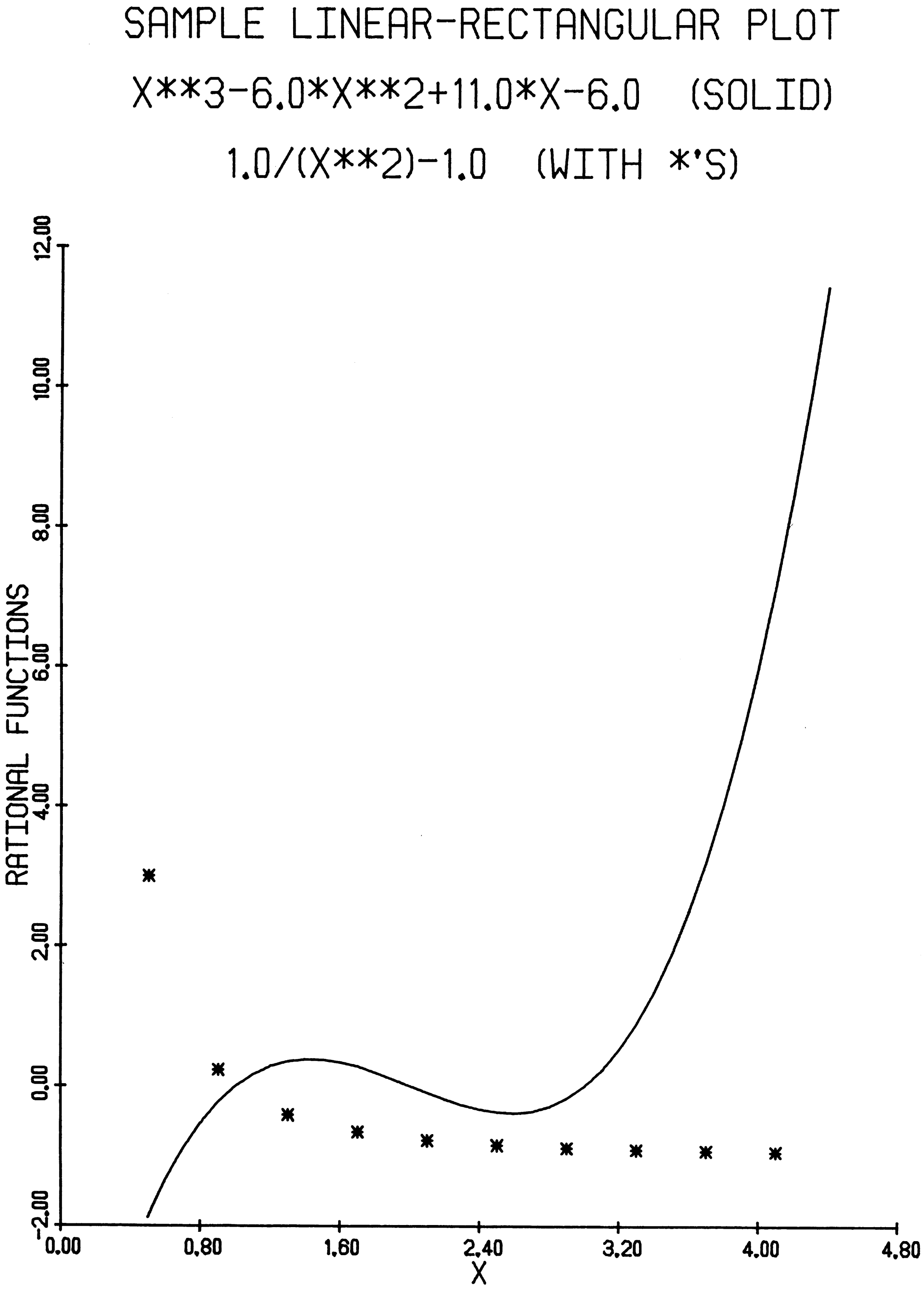
The table below shows the minimum file size to contain the plot description for each of the examples. This should help to give some feel for the file sizes necessary for plot files. Examples 4 and 5 contain circular grids which generate a large number of points.

Example	File Size (pages)
1	1
2	1
3	4
4	13
5	36
6	4

Example 1

```
C===============================================================C
C                                                               C
C                SAMPLE LINEAR-RECTANGULAR PLOT                 C
C                                                               C
C===============================================================C
      DIMENSION X(40),Y(80)
      LOGICAL*1 TL1(30)/’SAMPLE LINEAR-RECTANGULAR PLOT’/
      LOGICAL*1 TL2(33)/’X**3-6.0*X**2+11.0*X-6.0  (SOLID)’/
      LOGICAL*1 TL3(26)/’1.0/(X**2)-1.0  (WITH *’’S)’/

C---------------------------------------------------------------C
C        CALCULATE 40 VALUES FOR EACH OF THE CURVES:            C
C        (1) Y = 1.0/(X**2)-1.0                                 C
C        (2) Y = X**3-6.0*X**2+11.0*X-6.0                       C
C                                                               C
C        THE X VALUES ARE THE SAME FOR BOTH CURVES. THE Y       C
C        VALUES FOR (1) ARE STORED IN Y(1)...Y(40). THE Y       C
C        VALUES FOR (2) ARE STORED IN Y(41)...Y(80).            C
C---------------------------------------------------------------C
      U=0.5
      DO 1 I=1,40
        X(I)=U
        Y(I)=1./(U**2)-1.
        Y(I+40)=U**3-6.*U**2+11.*U-6.0
    1   U=U+.1

C---------------------------------------------------------------C
C        DETERMINE X AND Y SCALE FACTORS AND MINIMUM VALUES     C
C        FOR A 6 INCH X AXIS AND A 7 INCH Y AXIS BOTH HAVING    C
C        TICK MARKS 1 INCH APART.                               C
C---------------------------------------------------------------C
      CALL PSCALE(6.,1.,XMIN,DX,X(1),40,1)
      CALL PSCALE(7.,1.,YMIN,DY,Y(1),40,1,Y(41),40,1)

C---------------------------------------------------------------C
C        DEFINE THE COORDINATE SYSTEM WITH THE ORIGIN AT        C
C        (2.,2.) WITH THE MINIMUM VALUES AND SCALE FACTORS      C
C        DETERMINED BY PSCALE.                                  C
C---------------------------------------------------------------C
      CALL PLTOFS(XMIN,DX,YMIN,DY,2.,2.)

C---------------------------------------------------------------C
C        DRAW THE X AND Y AXES.                                 C
C---------------------------------------------------------------C
      CALL PAXIS(2.,2.,’X’,-1,6.,0.,XMIN,DX,1.)
      CALL PAXIS(2.,2.,’RATIONAL FUNCTIONS’,18,7.,90.,
     * YMIN,DY,1.)
```

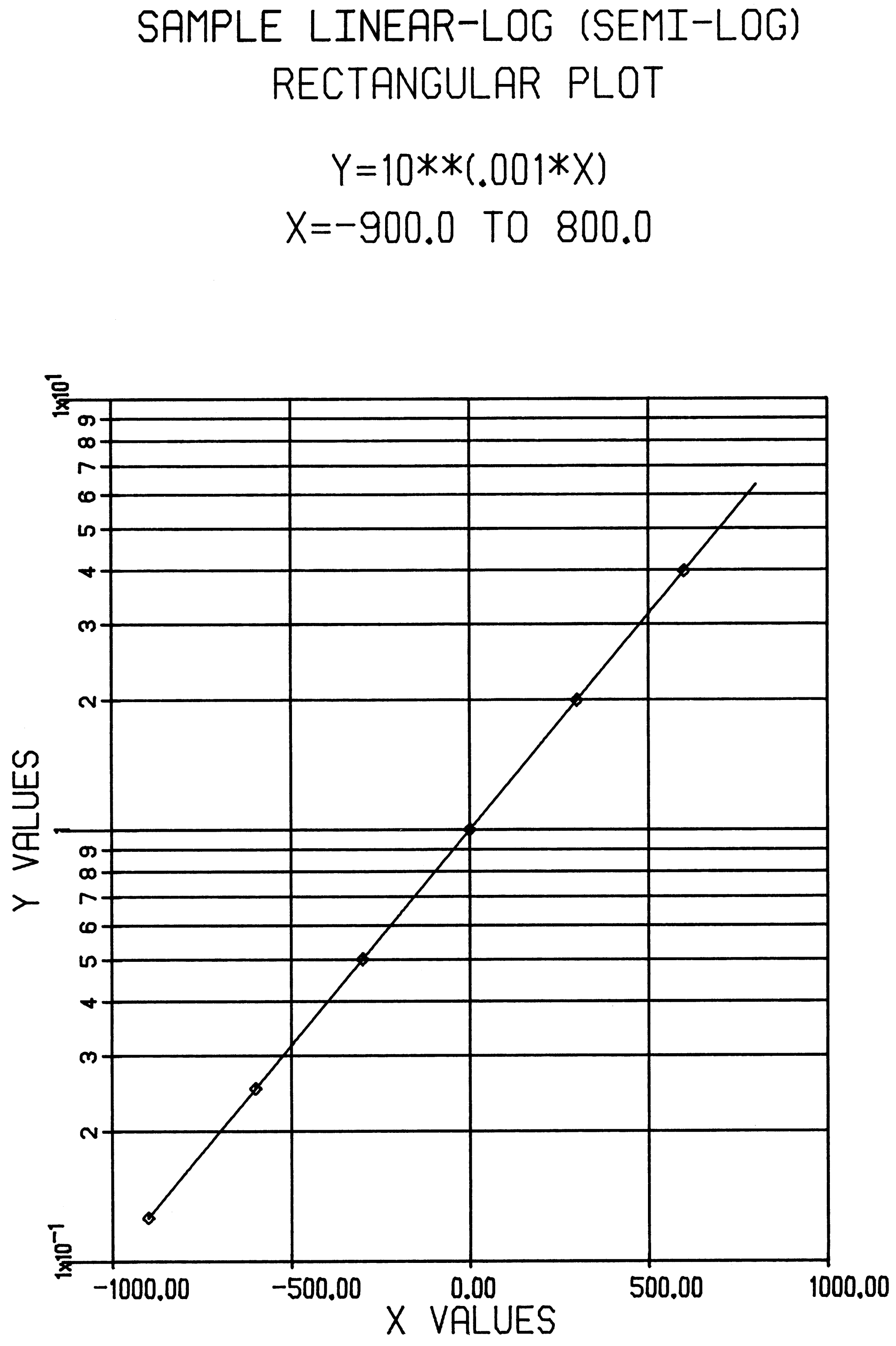
```
C---------------------------------------------------------------C
C        DRAW (2) USING A SOLID LINE.                           C
C---------------------------------------------------------------C
      CALL PLINE(X(1),Y(41),40,1,0,0,1)
C---------------------------------------------------------------C
C        DRAW (1) WITH AN ASTERISK AT EVERY 4TH POINT.          C
C---------------------------------------------------------------C
      CALL PLINE(X(1),Y(1),40,1,-4,11,1)

C---------------------------------------------------------------C
C        ADD THE TITLES, CENTERED ABOVE THE PLOT.               C
C---------------------------------------------------------------C
      CL=PSMLEN(TL1,30,.2)
      CALL PSYM(5.-CL/2.,10.5,.2,TL1,0.,30,0)
      CL=PSMLEN(TL2,33,.2)
      CALL PSYM(5.-CL/2.,10.,.2,TL2,0.,33,0)
      CL=PSMLEN(TL3,26,.2)
      CALL PSYM(5.-CL/2.,9.5,.2,TL3,0.,26,0)

C---------------------------------------------------------------C
C        TERMINATE THE PLOT                                     C
C---------------------------------------------------------------C
      CALL PLTEND
      STOP
      END
```


SAMPLE LINEAR-RECTANGULAR PLOT
X**3-6.0*X**2+11.0*X-6.0 (SOLID)
1.0/(X**2)-1.0 (WITH *'S)
RATIONAL FUNCTIONS
12.00
10.00
8.00
6.00
4.00
2.00
0.00
-2.00
0.00
0.80
1.60
2.40
3.20
4.00
4.80
X

Example 2

```
C================================================================C
C                                                                C
C                     SAMPLE SEMI-LOG PLOT                       C
C                                                                C
C================================================================C
      DIMENSION X(18),Y(18)
      LOGICAL*1 TL1(28)/'SAMPLE LINEAR-LOG (SEMI-LOG)'/
      LOGICAL*1 TL2(16)/'RECTANGULAR PLOT'/
      LOGICAL*1 TL3(14)/'Y=10**(.001*X)'/
      LOGICAL*1 TL4(17)/'X=-900.0 TO 800.0'/

C----------------------------------------------------------------C
C        CALCULATE 18 VALUES FOR THE CURVE                       C
C        Y=10**(.001*X)                                          C
C----------------------------------------------------------------C
      XX=-900.
      DO 1 I=1,18
        X(I)=XX
        Y(I)=10.**(.001*XX)
    1   XX=XX+100.

C----------------------------------------------------------------C
C        SINCE THE RANGE OF THE DATA IS KNOWN, THE SCALING       C
C        SUBROUTINES ARE NOT USED. YEXMIN IS THE NEAREST         C
C        POWER OF 10 LESS THAN OR EQUAL TO THAT OF THE           C
C        SMALLEST Y VALUE. DYEX IS LOG CYCLES PER INCH. TWO      C
C        CYCLES ARE NEEDED ON A 6 INCH AXIS (ARBITRARY).         C
C----------------------------------------------------------------C
      YEXMIN=-1.
      DYEX=2./6.

C----------------------------------------------------------------C
C        CHOOSE XMIN AND DX FOR A 5 INCH X AXIS WITH TICK        C
C        MARKS EVERY 1.25 INCHES. ROUND XMIN AND DX TO GIVE      C
C        "NICE" VALUES.                                          C
C----------------------------------------------------------------C
      XMIN=-1000.
      DX=2000./5.

C----------------------------------------------------------------C
C        DEFINE THE COORDINATE SYSTEM WITH THE ORIGIN AT         C
C        (1.5,1.).                                               C
C----------------------------------------------------------------C
      CALL PLTOFS(XMIN,DX,YEXMIN,DYEX,1.5,1.)

C----------------------------------------------------------------C
C        DRAW THE AXES (LINEAR X, LOG Y).                        C
C----------------------------------------------------------------C
      CALL PAXIS(1.5,1.,'X VALUES',-8,5.,0.,XMIN,DX,1.25)
      CALL PLGAXS(1.5,1.,'Y VALUES',8,6.,90.,YEXMIN,DYEX)
```

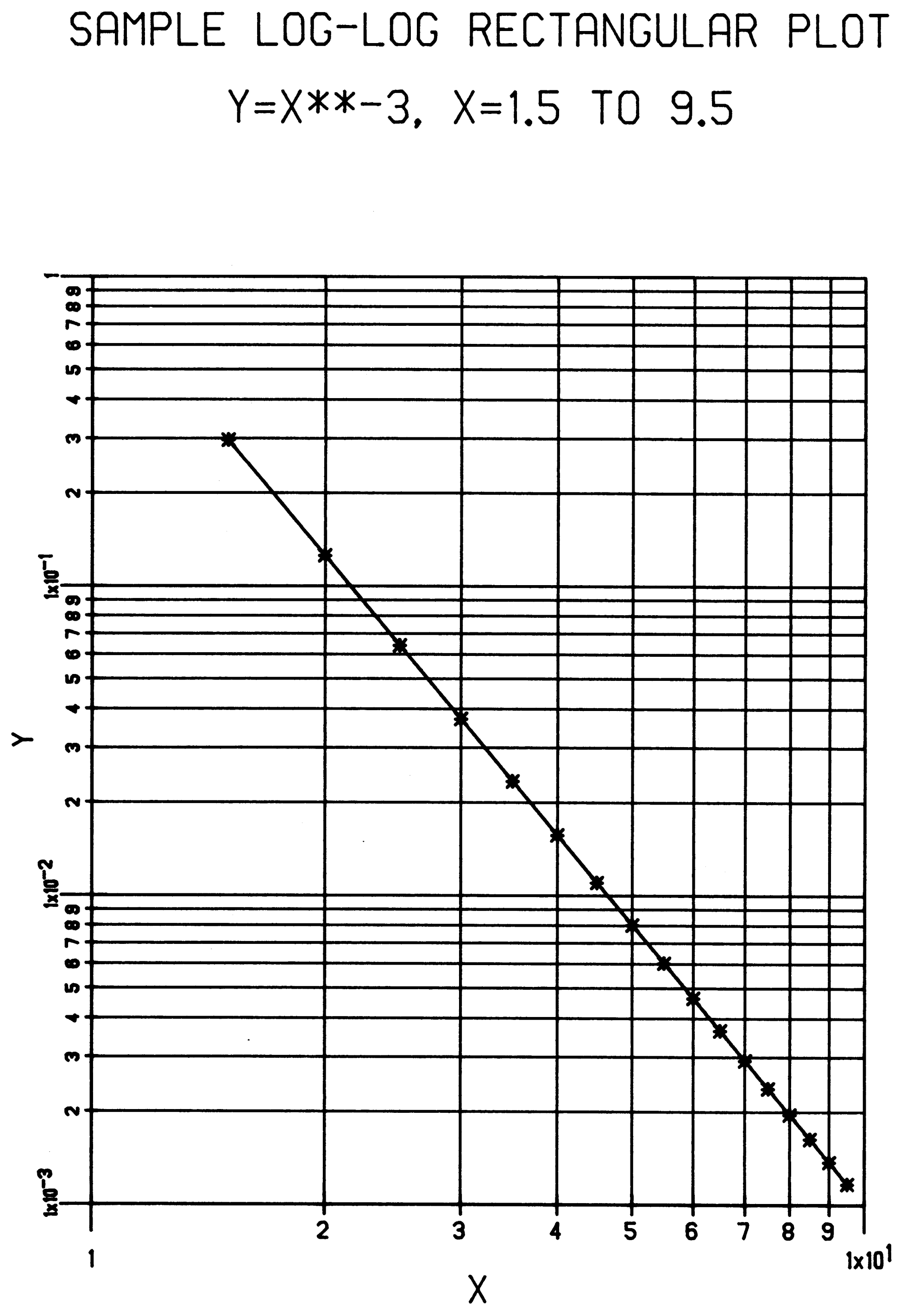
```
  C---------------------------------------------------------------C
| C        DRAW A LINEAR GRID FOR THE X AXIS AND A                C
| C        LOGARITHMIC GRID FOR THE Y AXIS.                       C
  C---------------------------------------------------------------C
        CALL PGRID(1.5,1.,1.25,6.,4,1)
        CALL PLGGRD(1.5,1.,1./DYEX,6.,-5.,90.)
  C---------------------------------------------------------------C
| C        ENTER LOG-LINEAR PLOTTING MODE.                        C
  C---------------------------------------------------------------C
        CALL PLTLOG(1)
  C---------------------------------------------------------------C
| C        DRAW THE CURVE USING A LINE WITH A SYMBOL AT EVERY     C
| C        3RD POINT.                                             C
  C---------------------------------------------------------------C
        CALL PLINE(X,Y,18,1,3,5,1)
  C---------------------------------------------------------------C
| C        RETURN TO LINEAR PLOTTING MODE.                        C
  C---------------------------------------------------------------C
        CALL PLTREC
  C---------------------------------------------------------------C
| C        ADD THE TITLES, CENTERED ABOVE THE PLOT.               C
  C---------------------------------------------------------------C
        CL=PSMLEN(TL1,28,.2)
        CALL PSYM(4.-CL/2.,9.5,.2,TL1,0.,28,0)
        CL=PSMLEN(TL2,16,.2)
        CALL PSYM(4.-CL/2.,9.1,.2,TL2,0.,16,0)
        CL=PSMLEN(TL3,14,.2)
        CALL PSYM(4.-CL/2.,8.5,.2,TL3,0.,14,0)
        CL=PSMLEN(TL4,17,.2)
        CALL PSYM(4.-CL/2.,8.1,.2,TL4,0.,17,0)
  C---------------------------------------------------------------C
| C        TERMINATE THE PLOT                                     C
  C---------------------------------------------------------------C
        CALL PLTEND
        STOP
        END
```


SAMPLE LINEAR-LOG (SEMI-LOG)
RECTANGULAR PLOT
Y=10**(.001*X)
X=-900.0 TO 800.0
1x10^1
9
8
7
6
5
4
3
2
1
9
8
7
6
5
4
3
2
1x10^-1
Y VALUES
-1000.00
-500.00
0.00
500.00
1000.00
X VALUES

Example 3

```
C===============================================================C
C                                                               C
C                     SAMPLE LOG-LOG PLOT                       C
C                                                               C
C===============================================================C
      DIMENSION X(801),Y(801)
      LOGICAL*1 TL1(31)/’SAMPLE LOG-LOG RECTANGULAR PLOT’/
      LOGICAL*1 TL2(21)/’Y=X**-3, X=1.5 TO 9.5’/

C---------------------------------------------------------------C
C        COMPUTE Y=X**-3 FOR X=1.5 TO 9.5 BY 0.01               C
C---------------------------------------------------------------C
      XX=1.5
      DO 1 I=1,801
        X(I)=XX
        Y(I)=XX**(-3)
    1   XX=XX+.01

C---------------------------------------------------------------C
C        FIND THE MINIMUM VALUE AND SCALE FACTOR FOR            C
C        A 6 INCH Y AXIS.                                       C
C---------------------------------------------------------------C
      CALL PLGSCL(6.,YEXMIN,DYEX,Y,801,1)

C---------------------------------------------------------------C
C        CHOOSE VALUES FOR THE X AXIS                           C
C---------------------------------------------------------------C
      XEXMIN=0.
      DXEX=1./5.

C---------------------------------------------------------------C
C        DEFINE THE COORDINATE SYSTEM WITH THE ORIGIN AT        C
C        (1.5,1.5).                                             C
C---------------------------------------------------------------C
      CALL PLTOFS(XEXMIN,DXEX,YEXMIN,DYEX,1.5,1.5)

C---------------------------------------------------------------C
C        DRAW THE AXES.                                         C
C---------------------------------------------------------------C
      CALL PLGAXS(1.5,1.5,’X’,-1,5.,0.,XEXMIN,DXEX)
      CALL PLGAXS(1.5,1.5,’Y’,1,6.,90.,YEXMIN,DYEX)

C---------------------------------------------------------------C
C        DRAW A LOGARITHMIC GRID FOR EACH AXIS.                 C
C---------------------------------------------------------------C
      CALL PLGGRD(1.5,1.5,1./DXEX,5.,6.,0.)
      CALL PLGGRD(1.5,1.5,1./DYEX,6.,-5.,90.)
```

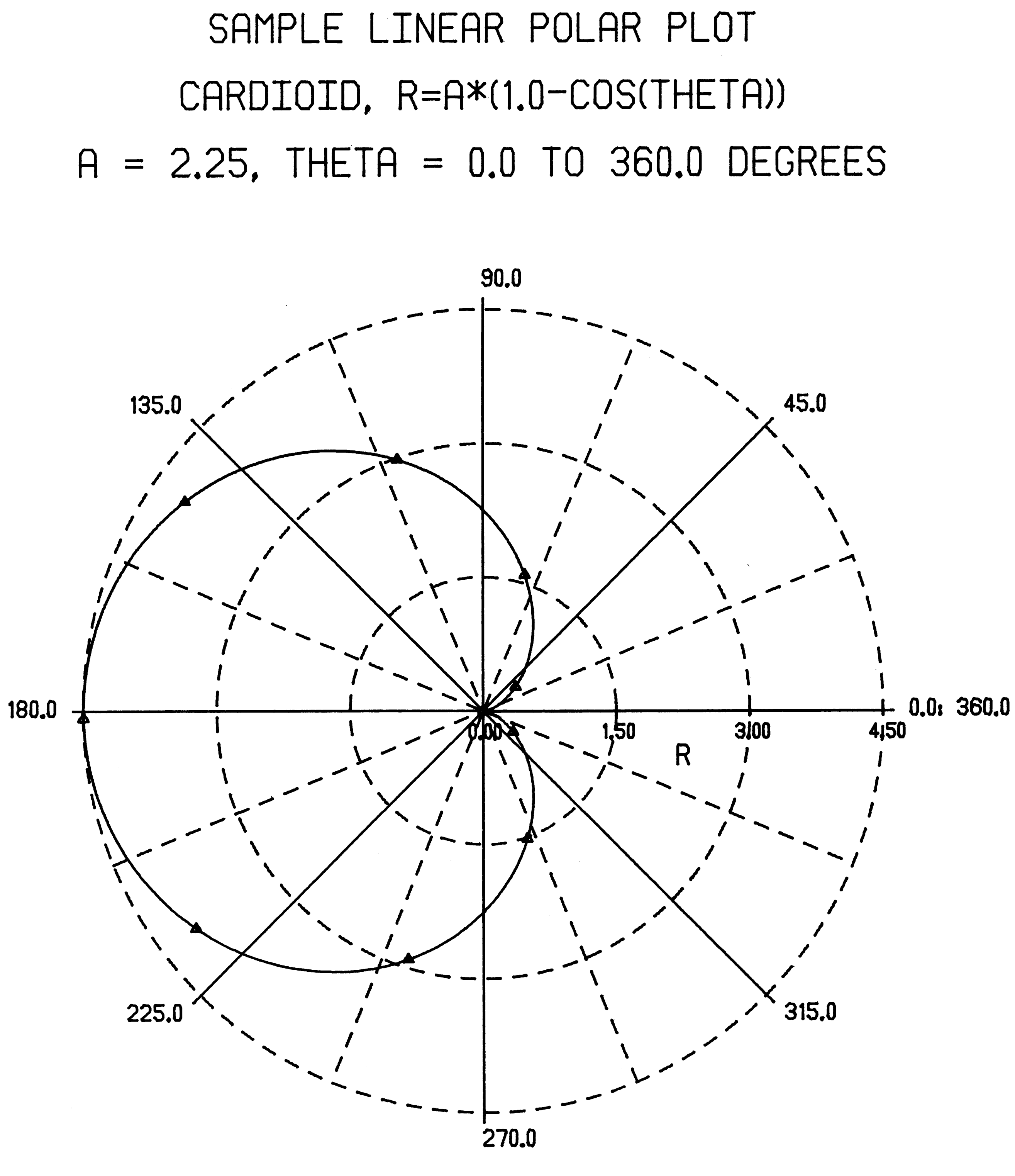
```
  C---------------------------------------------------------------C
  C        ENTER LOG-LOG PLOTTING MODE                            C
  C---------------------------------------------------------------C
        CALL PLTLOG(3)
  C---------------------------------------------------------------C
| C        DRAW THE CURVE WITH AN ASTERISK AT EVERY 50TH POINT.   C
  C---------------------------------------------------------------C
        CALL PLINE(X,Y,801,1,50,11,1)
  C---------------------------------------------------------------C
| C        RETURN TO LINEAR PLOTTING MODE.                        C
  C---------------------------------------------------------------C
        CALL PLTREC
  C---------------------------------------------------------------C
| C        ADD THE TITLES, CENTERED ABOVE THE PLOT.               C
  C---------------------------------------------------------------C
        CL=PSMLEN(TL1,31,.2)
        CALL PSYM(4.-CL/2.,9.,.2,TL1,0.,31,0)
        CL=PSMLEN(TL2,21,.2)
        CALL PSYM(4.-CL/2.,8.5,.2,TL2,0.,21,0)
  C---------------------------------------------------------------C
| C        TERMINATE THE PLOT.                                    C
  C---------------------------------------------------------------C
        CALL PLTEND
        STOP
        END
```


SAMPLE LOG-LOG RECTANGULAR PLOT
Y=X**-3, X=1.5 TO 9.5
Y
1
9
8
7
6
5
4
3
2
1x10^-1
9
8
7
6
5
4
3
2
1x10^-2
9
8
7
6
5
4
3
2
1x10^-3
1
2
3
4
5
6
7
8
9
1x10^1
X

Example 4

```
C===============================================================C
C                                                               C
C                  SAMPLE LINEAR POLAR PLOT                     C
C                                                               C
C===============================================================C
      DIMENSION R(361),THETA(361)
      LOGICAL*1 TL1(24)/'SAMPLE LINEAR POLAR PLOT'/
      LOGICAL*1 TL2(30)/'CARDIOID, R=A*(1.0-COS(THETA))'/
      LOGICAL*1 TL3(38)/'A = 2.25, THETA = 0.0 TO 360.0 DEGREES'/

C---------------------------------------------------------------C
C        GENERATE A CARDIOID USING ANGULAR STEPS OF 1 DEGREE    C
C---------------------------------------------------------------C
      A=2.25
      RADEG=3.1415927/180.
      DO 1 I=1,361
        THETA(I)=I
    1   R(I)=A*(1.-COS(THETA(I)*RADEG))

C---------------------------------------------------------------C
C        PREPARE RADIAL SCALE FACTOR AND MINIMUM VALUE.         C
C        PSCALE COULD BE USED TO COMPUTE THESE, BUT THE         C
C        RANGE IS KNOWN. R VARIES FROM 0.0 TO 4.5               C
C        OVER 3.0 INCHES (THE RADIUS OF THE PLOT.)              C
C---------------------------------------------------------------C
      RMIN=0.
      DR=4.5/3.

C---------------------------------------------------------------C
C        NOW SET UP THE ANGULAR SCALE. THETA VARIES FROM 0.0    C
C        TO 360.0 OVER 360 DEGREES.                             C
C---------------------------------------------------------------C
      THMIN=0.
      DTH=1.

C---------------------------------------------------------------C
C        DEFINE THE LINEAR POLAR COORDINATE SYSTEM WITH THE     C
C        ORIGIN AT (5.,5.).                                     C
C---------------------------------------------------------------C
      CALL PLTOFS(RMIN,DR,THMIN,DTH,5.,5.)

C---------------------------------------------------------------C
C        DRAW A SOLID POLAR AXIS STARTING AT THETA=0.0 WITH     C
C        A RADIUS OF 3.0 INCHES HAVING RAYS EVERY 45.0 DEGREES. C
C---------------------------------------------------------------C
      CALL PTHAXS(5.,5.,3.,0.,45.,360.,0.,45.,0.,0)
```

```
C---------------------------------------------------------------C
C        DRAW A POLAR GRID WITH 0.1 INCH DASHES HAVING RAYS     C
C        EVERY 45 DEGREES BETWEEN THE SOLID RAYS DRAWN BY       C
C        PTHAXS. NOTE THAT -1.0 MEANS DASHED RADIAL LINES AND   C
C        -45.0 MEANS DASHED CIRCULAR ARCS.                      C
C---------------------------------------------------------------C
      CALL POLGRD(5.,5.,-1.,3.,22.5,-45.,382.5,.1)
C---------------------------------------------------------------C
C        DRAW A 3 INCH HORIZONTAL RADIAL AXIS.                  C
C---------------------------------------------------------------C
      CALL PAXIS(5.,5.,’R’,-1,3.,0.,RMIN,DR,1.)

C---------------------------------------------------------------C
C        ENTER POLAR PLOTTING MODE (ANGLES IN DEGREES).         C
C---------------------------------------------------------------C
      CALL PLTPOL(1)

C---------------------------------------------------------------C
C        DRAW THE CURVE WITH A TRIANGLE AT EVERY 36TH POINT.    C
C---------------------------------------------------------------C
      CALL PLINE(R,THETA,361,1,36,2,1)

C---------------------------------------------------------------C
C         RETURN TO LINEAR-RECTANGULAR PLOTTING MODE.           C
C---------------------------------------------------------------C
      CALL PLTREC

C---------------------------------------------------------------C
C         ADD THE TITLES, CENTERED ABOVE THE PLOT.              C
C---------------------------------------------------------------C
      CL=PSMLEN(TL1,24,.2)
      CALL PSYM(5.-CL/2.,10.,.2,TL1,0.,24,0)
      CL=PSMLEN(TL2,30,.2)
      CALL PSYM(5.-CL/2.,9.5,.2,TL2,0.,30,0)
      CL=PSMLEN(TL3,38,.2)
      CALL PSYM(5.-CL/2.,9.,.2,TL3,0.,38,0)

C---------------------------------------------------------------C
C        TERMINATE THE PLOT                                     C
C---------------------------------------------------------------C
      CALL PLTEND
      STOP
      END
```


SAMPLE LINEAR POLAR PLOT
CARDIOID, R=A*(1.0-COS(THETA))
A = 2.25, THETA = 0.0 TO 360.0 DEGREES
90.0
135.0
45.0
180.0
0.0: 360.0
0.00
1.50
3.00
4.50
R
225.0
315.0
270.0

Example 5

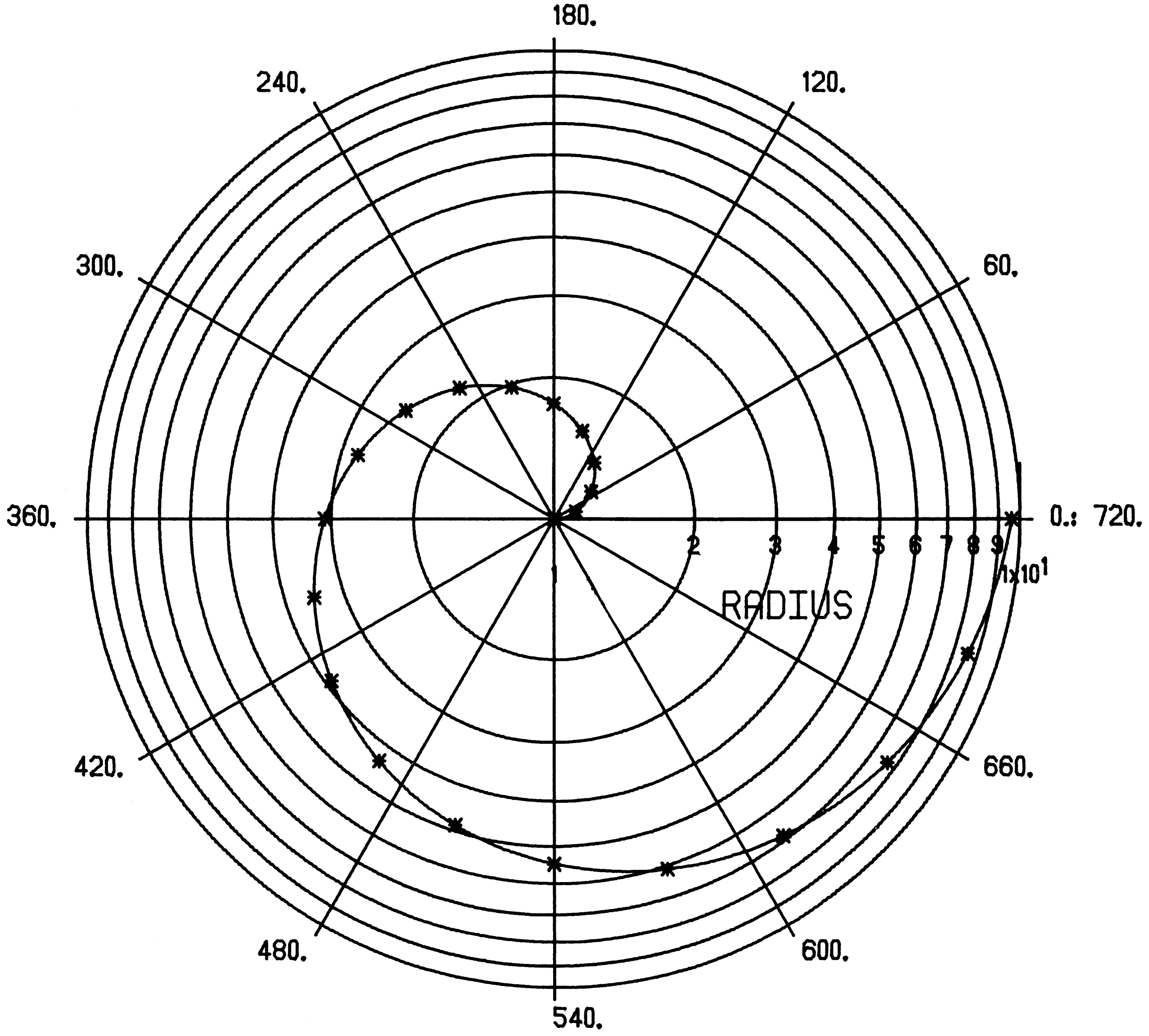
```
C===============================================================C
C                                                               C
C                    SAMPLE LOG-POLAR PLOT                      C
C                                                               C
C===============================================================C
      DIMENSION R(721),THETA(721)
      LOGICAL*1 TL1(21)/’SAMPLE LOG-POLAR PLOT’/
      LOGICAL*1 TL2(34)/’LOGARITHMIC SPIRAL, R=EXP(A*THETA)’/
      LOGICAL*1 TL3(38)/’A =  .18, THETA = 0.0 TO 720.0 DEGREES’/

C---------------------------------------------------------------C
C        GENERATE A LOGARITHMIC SPIRAL USING ANGLES IN RADIANS. C
C---------------------------------------------------------------C
      A=.18
      RADEG=3.1415927/180.
      DO 1 I=1,721
        THETA(I)=(I-1)*RADEG
    1   R(I)=EXP(A*THETA(I))

C---------------------------------------------------------------C
C        OBTAIN RADIAL SCALE FACTOR AND MINIMUM VALUE FOR       C
C        A 3.0 INCH LOG AXIS.                                   C
C---------------------------------------------------------------C
      CALL PLGSCL(3.,RMIN,DR,R,721,1)

C---------------------------------------------------------------C
C        THE RANGE OF THETA IS 0 TO 4PI. THE TWO FULL           C
C        REVOLUTIONS OF THE SPIRAL ARE TO BE MAPPED INTO        C
C        ONE PHYSICAL REVOLUTION.                               C
C---------------------------------------------------------------C
      THMIN=0.
      DTH=2.

C---------------------------------------------------------------C
C        DEFINE THE COORDINATE SYSTEM WITH THE ORIGIN AT        C
C        (4.,4.).                                               C
C---------------------------------------------------------------C
      CALL PLTOFS(RMIN,DR,THMIN,DTH,4.,4.)

C---------------------------------------------------------------C
C        DRAW A SOLID POLAR AXIS WITH A 3.0 INCH RADIUS         C
C        HAVING RAYS EVERY 30.0 DEGREES.                        C
C---------------------------------------------------------------C
      CALL PTHAXS(4.,4.,3.,0.,30.,360.,0.,60.,0.,’F3.0’)

C---------------------------------------------------------------C
C        DRAW A LOG-POLAR GRID WITH SOLID CIRCLES AND NO RAYS   C
C        SINCE THE RAYS FROM PTHAXS ARE SUFFICIENT.             C
C---------------------------------------------------------------C
      CALL POLGRD(4.,4.,1./DR,-3.,0.,0.,360.,0.)
```

```
C---------------------------------------------------------------C
C        DRAW A RADIAL LOG AXIS.                                C
C---------------------------------------------------------------C
      CALL PLGAXS(4.,4.,’RADIUS’,-6,3.,0.,RMIN,DR)
C---------------------------------------------------------------C
C        ENTER LOG-POLAR PLOTTING MODE (ANGLES IN RADIANS).     C
C---------------------------------------------------------------C
      CALL PLGPOL(0)

C---------------------------------------------------------------C
C        DRAW THE CURVE WITH AN ASTERISK AT EVERY 36TH POINT.   C
C---------------------------------------------------------------C
      CALL PLINE(R,THETA,721,1,36,11,1)

C---------------------------------------------------------------C
C        RETURN TO LINEAR-RECTANGULAR PLOTTING MODE.            C
C---------------------------------------------------------------C
      CALL PLTREC

C---------------------------------------------------------------C
C        ADD THE TITLES, CENTERED ABOVE THE PLOT.               C
C---------------------------------------------------------------C
      CL=PSMLEN(TL1,21,.2)
      CALL PSYM(4.-CL/2.,9.,.2,TL1,0.,21,0)
      CL=PSMLEN(TL2,34,.2)
      CALL PSYM(4.-CL/2.,8.5,.2,TL2,0.,34,0)
      CL=PSMLEN(TL3,38,.2)
      CALL PSYM(4.-CL/2.,8.,.2,TL3,0.,38,0)

C---------------------------------------------------------------C
C        TERMINATE THE PLOT.                                    C
C---------------------------------------------------------------C
      CALL PLTEND
      STOP
      END
```

SAMPLE LOG-POLAR PLOT

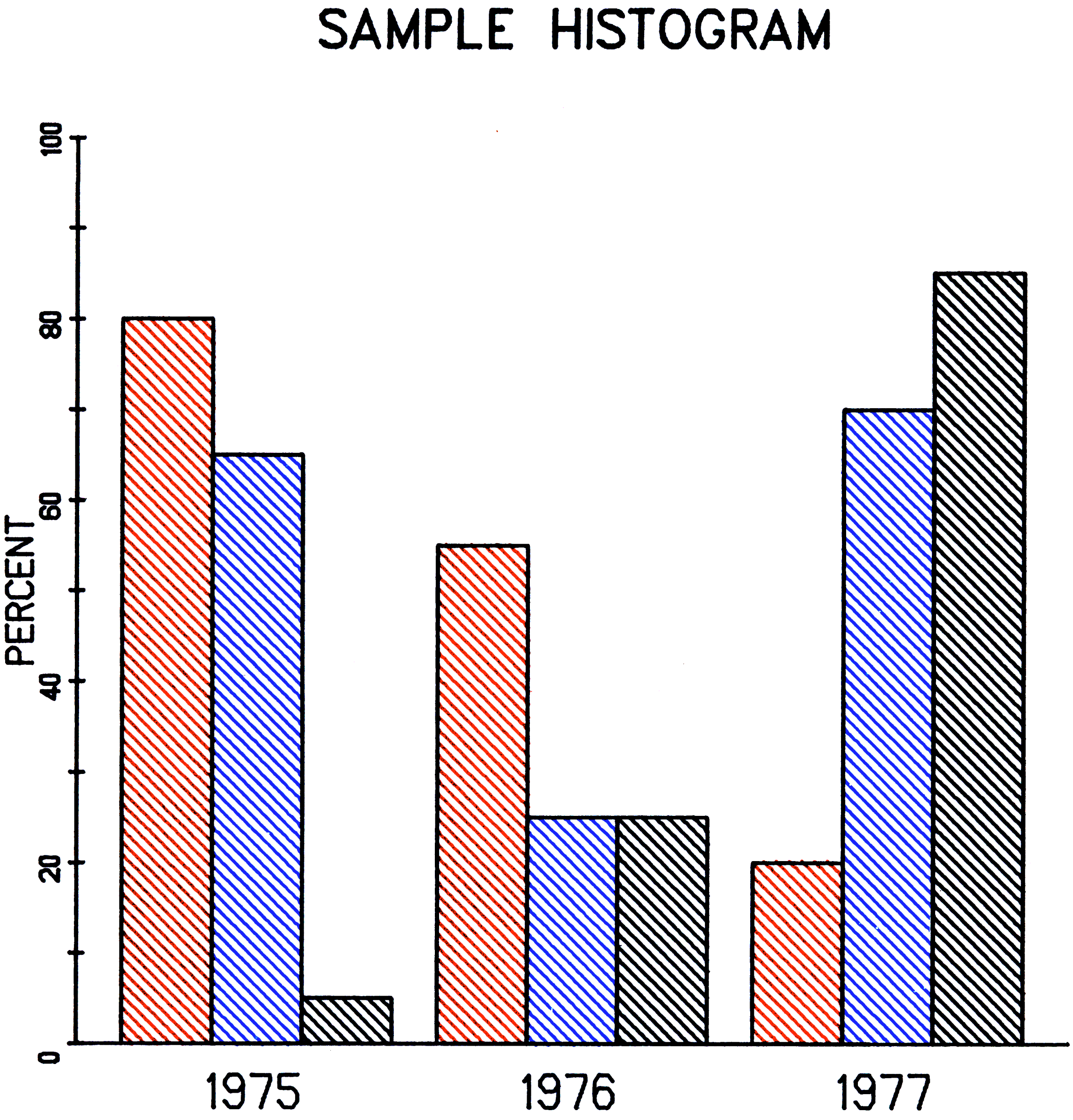
LOGARITHMIC SPIRAL, R=EXP(A*THETA)

A = .18, THETA = 0.0 TO 720.0 DEGREES



Example 6

```
C===============================================================C
C                                                               C
C                      SAMPLE HISTOGRAM                         C
C                                                               C
C===============================================================C
      DIMENSION XB(4),YB(4)
      DATA YB(1)/0./,YB(4)/0./
      REAL Y(3,3)/80.,55.,20.,65.,25.,70.,5.,25.,85./
      INTEGER COLOR(3)/’RED ’,’BLUE’,’BLAC’/
      LOGICAL*1 TL1(16)/’SAMPLE HISTOGRAM’/

C---------------------------------------------------------------C
C         DRAW THE AXES AND DEFINE THE COORDINATE SYSTEM. AN    C
C         ALTERNATE FORMAT IS USED FOR THE AXES SINCE NONE OF   C
C         THE NUMBERS HAVE FRACTIONAL PARTS.                    C
C---------------------------------------------------------------C
      CALL PAXFRM(’MF4.0*’)
      CALL PAXIS(2.,-2.,’PERCENT’,7,5.,90.,0.,20.,.5)
      CALL PAXIS(-2.9,-2.,’ ’,-1,3.5,0.,1975.,1./1.75,1.75)
      CALL PENUP(2.,2.)
      CALL PENDN(7.5,2.)
      CALL PLTOFS(0.,1.,0.,20.,2.,2.)

C---------------------------------------------------------------C
C         LOOP TO OUTLINE EACH RECTANGLE IN BLACK AND           C
C         SHADE IT IN THE APPROPRIATE COLOR.                    C
C---------------------------------------------------------------C
      DO 10 J=1,3
        XL=.25+(J-1)*.5
        DO 20 I=1,3
          XB(1)=XL
          XB(2)=XL
          YB(2)=Y(I,J)
          XB(3)=XL+.5
          YB(3)=Y(I,J)
          XB(4)=XL+.5
          CALL PLINE(XB,YB,4,1,0,0,1)
          CALL PENCHG(COLOR(J))
          CALL PSHADE(XB,YB,4,1,45.,.05,1)
          CALL PENCHG(’BLAC’)
20        XL=XL+1.75
10      CONTINUE

C---------------------------------------------------------------C
C         DRAW THE TITLE CENTERED ABOVE THE PLOT. USE           C
C         BOLD FACE CHARACTERS FOR IT.                          C
C---------------------------------------------------------------C
      CALL PALPHA(’SANSERIF.2 ’,0)
      CL=PSMLEN(TL1,16,.2)
      CALL PSYM(4.75-CL/2.,7.5,.2,TL1,0.,16,0)
      CALL PALPHA('STANDARD ',0)
C---------------------------------------------------------------C
C         TERMINATE THE PLOT                                    C
C---------------------------------------------------------------C
      CALL PLTEND
      STOP
      END
```


SAMPLE HISTOGRAM
PERCENT
0
20
40
60
80
100
1975
1976
1977

PDS SUBROUTINE DESCRIPTIONS

This section contains complete descriptions of each of the subroutines in the PDS. The descriptions are ordered alphabetically.

Unless otherwise noted, all parameter names follow the normal FORTRAN conventions. If the name begins with I-N, the parameter is integer; otherwise it is real. Care must be taken to ensure that each parameter is of the correct type. Numbers of the wrong type cause unpredictable results.

360/370 assembler language terminology is used in the subroutine descriptions. Floating-point is equivalent to REAL*4 (FORTRAN) and to FLOAT(6) (PL/I); integer and fullword integer are equivalent to INTEGER* 4 and FIXED BINARY(31); halfword integer is equivalent to INTEGER*2 and FIXED BINARY(15).

A few of the subroutines use the value -0.0 as a special case. This is a valid value and can be distinguished from a true zero. It is treated by the machine as a true zero in all arithmetic operations. The internal representation of this value is 80000000 in hexadecimal.

Return codes are used by some of the subroutines to indicate error conditions. A return code of zero always indicates successful completion. Nonzero values are always multiples of four. In FORTRAN, the return code may be tested by adding "&stno" (where "stno" is a statement number) to the end of the parameter list in the CALL statement. If the return code is zero, control passes to the following statement. If the return code is four, control passes to the statement labeled "stno". Additional statement number parameters may be used to test for higher return codes. More information on return codes may be found in MTS Volume 3, System Subroutine Descriptions.

All PDS subroutines use OS type I calling conventions, which are described in the section "Calling Conventions" in MTS Volume 3, System Subroutine Descriptions. Thus they may be called from FORTRAN in the usual way. Some of the subroutines may be called either as subroutines or as functions. Some FORTRAN compilers treat the use of a name as both a subroutine and a function as a severe error. For such compilers, it is necessary that a given routine be used consistently either as a subroutine or as a function, even if the function result is not always needed. The FORTRAN compilers in MTS which are accessed via *FTN and *IF are not subject to this restriction.

PALPHA

Subroutine Description

Purpose: PALPHA changes the character set used by PSYM. In addition to the standard character set, several library sets are provided, and user-defined character sets are permitted.

Calling Sequence:

```
CALL PALPHA(name,what,&rc4)
```

Parameters:

name is the name of the character set to be used, if the set is a library set or the standard one. For a user-defined set, the name of the file containing the character-set definition should be given. In both cases, the name must be terminated by a blank. The name of the standard character set is "STANDARD ".

what is a fullword integer (INTEGER*4) which is zero if NAME is the name of a library character set (or the standard one), and nonzero if it is a file name for a user-defined set.

Return Codes:

0 Successful return, character set has been changed.

4 Unsuccessful return, the character set has not been found, or an error occurred while reading it.

Description: All subroutines in the Plot Description System which draw symbols call the PSYM subroutine. Therefore, calling PALPHA causes _all_ symbols to be drawn with the specified character set.

If a library character set is specified, it is read from ETC:PLOTALPHA. If a user-defined set is specified, it is read from the file named in _name_. If the character set is read successfully, succeeding calls to PSYM use the new set.

The library character sets are described in Appendix A. The remainder of this description need be read only by those defining a new character set.

Each character set occupies three lines in a file. The first line contains the control block, which is described below. The second line is a directory for the character information on the third line. The directory consists of 256 halfwords; halfword "m" (0 ≤ m ≤ 255) is the displacement of the information for character "m" in the third line. The third line contains the actual character definitions.

Each character is defined by a sequence of points on an n*n grid. The coordinates of the lower-left corner are (0,0); of the lower-right corner (n-1,0). The character is drawn by drawing straight lines between successive points. The point (15,15) (or (255,255) if n>16) means move to the next point without drawing (a pen up operation). To draw to the point (15,15) when n≤16, two successive (15,15)'s are used. A pen up is not necessary before the first point. If n≤16 each point is represented in the character information by a single byte. The high-order 4 bits of the byte contain the hexadecimal representation of the first coordinate, and the low-order 4 bits contain the second coordinate. If n>16, each point is represented by two bytes, the first containing the first coordinate, and the second containing the second coordinate. The format of the information for each character is:

```
1 byte number of points (k)
1 byte letterspacing (see below)
k (or 2k for n>16) bytes points
```

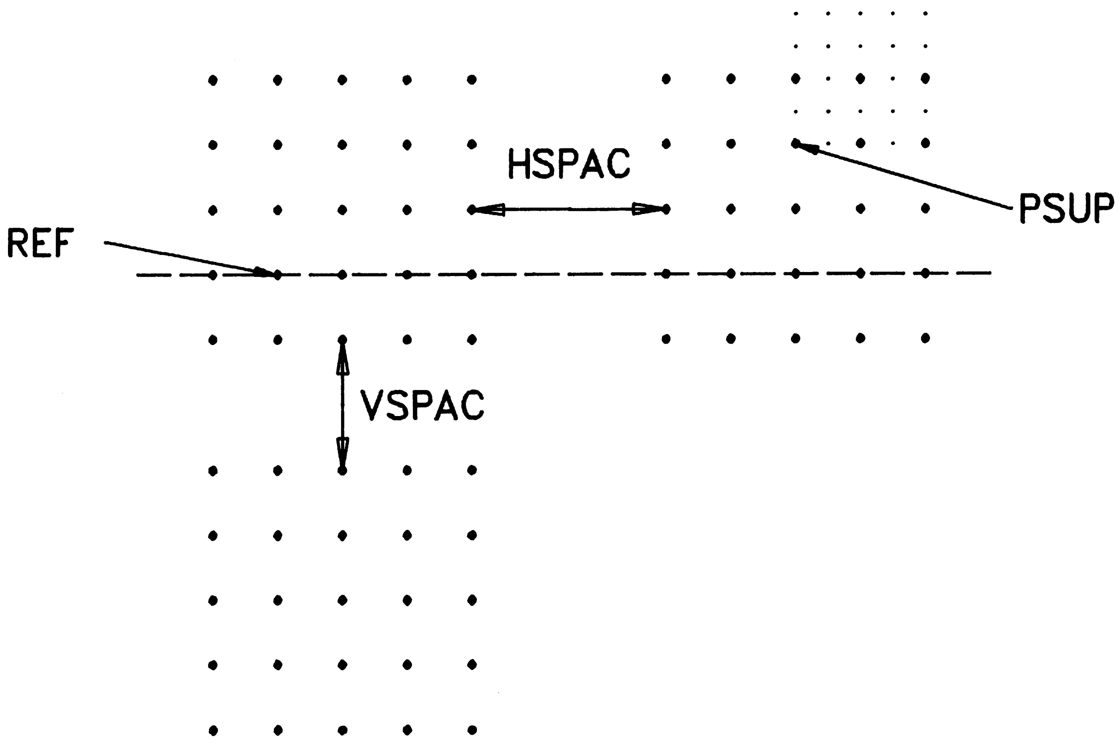
The character definitions may be packed end-to-end in the character information. For example, the information for a lowercase h on a 16x16 grid might be (in hex):

```
0809333FFF3A4B8B9A93
```

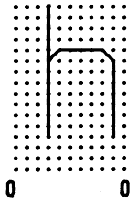
The structure of the control block is shown below. In the following table, H means a halfword integer (INTEGER*2), and E means a fullword real (REAL*4).

Disp.(dec)	Type	Name	Description
0	H	SIZE	is the total length of character information, in bytes.
2	H	GRID	is the grid size (GRID*GRID).
4	H	HSPAC	is the number of grid units between adjacent characters on the same line. It must satisfy HSPAC > -GRID.
6	H	VSPAC	is the number of grid units between lines of characters. It must satisfy VSPAC > -GRID.
8	2H	PSUP	are the coordinates of a grid point to be used as a reference point for superscripts (halfword X coordinate, halfword Y coordinate).
12	2H	PSUB	are the coordinates of a grid point to be used as a reference point for subscripts (halfword X coordinate, halfword Y coordinate).
16	E	SHGT	is (height of subscripts and superscripts) / character height.
20	E	SCAL	are the grid proportions (horizontal/vertical).
24	E	SSCAL	is the height scale. Heights given to PSYM are multiplied by this value.
28	E	ANGL	is the angle of the Y grid axis from the vertical in degrees (<0 corresponds to left-slanting letters). It must satisfy -90.<ANGL<90.
32	2H	REF	are the coordinates of a grid point which is the reference point for characters. For the first character of a string drawn by PSYM, this grid point corresponds to (X,Y).
36	H	SWITCH	Only the high-order bit of SWITCH is currently used. If it is 1, letterspacing is enabled; if 0, letterspacing is disabled.
38	H	LSPAC	is the constant horizontal spacing (similar to HSPAC) used when letterspacing is enabled.
40	CL16	NAME	is the 16-character alphabet name, padded with blanks if necessary.

The figure below illustrates some of the above parameters.



Variable character spacing, or letterspacing, may be used in character sets. In a set with variable spacing, the definition of each character includes the width of the character so that the space allotted that character can be shortened (which will allow the next character to appear closer to it). Specifically, the second byte of the character definition contains (in hex) the coordinate of the column which is considered the end of the character. This figure illustrates this for the character "h" whose definition was given earlier:



PARROW, PARRO2

Subroutine Description

Purpose: PARROW and PARRO2 draw curves which terminate with arrowheads.

Calling Sequence:

```
CALL PARROW(x,y,n,k,arhl,lin,mode)

CALL PARRO2(x,y,n,k,k1,arhl,lin,mode)
```

Parameters:

x is a floating-point (REAL*4) array of X coordinate values. *x* can contain X coordinates for rectangular plotting or radial coordinates for polar plotting.

y is a floating-point (REAL*4) array of Y coordinate values. *y* can contain Y coordinates for rectangular plotting or angular coordinates for polar plotting.

n is a fullword integer (INTEGER*4) whose magnitude is the number of elements to be processed in the *x* and *y* arrays. If *n* is zero, no processing is done.

k is the fullword-integer (INTEGER*4) spacing of the elements in the *x* and *y* arrays. The elements are situated in every *k*th position of the array starting with the first position. *k* should be 1 if the elements are in adjacent positions. If every other value in an array is to be used, *k* should be 2. If a double-precision array is used, *k* must be 2 (or a multiple of 2) to use the most significant part of each element.

k1 is similar in function to *k*, and describes spacing of data elements in the *y* array so that different spacings may be specified for *x* and *y*.

arhl is the floating-point length of the arrowhead in inches. If *arhl* is zero, then 0.3 inches is assumed. The width of the arrowhead is always one-third of its length.

lin If *lin* is nonzero, a line segment is drawn perpendicular to the arrowhead through the *n*th data point. The length of the line segment is twice the length of the arrowhead. This is useful in constructing dimension lines in drafting. If lin is zero, the line segment is not drawn.

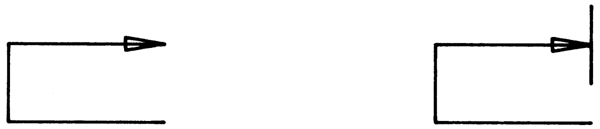
mode is a fullword-integer switch whose value indicates whether x and y are interpreted as absolute or relative coordinates. x and y are interpreted as absolute if mode is zero. For any other value of mode, x and y are treated as relative coordinates.

Description: Given a curve of n data points in corresponding x and y arrays, PARROW connects the points with straight line segments and draws an arrowhead of a specified length oriented along the line segment between the last two data points, pointing toward the last (nth) data point.

For PARROW, successive x and y values are k elements apart. For PARRO2, successive x values are k elements apart, while successive y values are k1 elements apart. This is the only difference between the two.

Example:

```
REAL X1(4)/2.,1.,1.,2./
REAL X2(4)/4.75,3.75,3.75,4.75/
REAL Y(4)/1.5,1.5,2.,2./
CALL PARROW(X1,Y,4,1,.25,0,0)
CALL PARROW(X2,Y,4,1,.25,1,0)
```



PAXFRM

Subroutine Description

Purpose: PAXFRM changes the format used by PAXIS for the numbering on tick marks.

Calling Sequence:

```
CALL PAXFRM(string)
```

Parameter:

string is an IOH format describing a floating-point format for the axis tick-mark numbering produced by PAXIS. It must end with an asterisk. If _string_ is '*', PAXIS reverts to the standard format.

Description: See the section "Introduction to IOH" for information on IOH formats.

| _string_ will be used by PAXIS whenever _y_ in the call to
| PAXIS is negative.

PAXIS

Subroutine Description

Purpose: PAXIS draws a linear axis which may have numbering and labeling (a title).

Calling Sequence:

```
CALL PAXIS(x,y,title,nchar,axlth,angle,xmin,dx,dv)

scalf = PAXIS(x,y,title,nchar,axlth,angle,xmin,dx,dv)
```

Parameters:

x,*y* are floating-point (REAL*4) values whose magnitudes are the absolute linear-rectangular coordinates (in inches) of the axis origin. If *x* is negative, the tick marks are deleted. If *y* is negative and a nonzero format term was given to the subroutine PAXFRM, it is used for the description of the axis numbering. If both coordinates are positive, then the tick marks are produced and a standard format is used.

title is a character string to be used as a centered axis title.

nchar is a fullword integer (INTEGER*4) whose magnitude is the number of characters in the axis title. The title and tick-mark numbering are placed on the clockwise side of the axis if *nchar* < 0 and are otherwise placed on the counterclockwise side of the axis. To suppress plotting of the title, *nchar* should be zero; however, as a *special case*, a *floating-point* -0.0 (hexadecimal 80000000) must be used to position the numbering on the clockwise side of the axis if no title is used.

axlth is a floating-point (REAL*4) value whose magnitude is the axis length in inches. If *axlth* is negative, numbering by the tick marks is deleted.

angle is the floating-point (REAL*4) angular orientation in degrees of the axis with respect to the horizontal (using a positive counterclockwise convention).

xmin is the floating-point (REAL*4) number to be drawn at the first tick mark.

dx is the floating-point (REAL*4) scale factor in axis variable units per inch (for instance seconds per inch).

dv is the floating-point (REAL*4) distance in inches between tick marks (division marks).

Value Returned:

scalf is the floating-point (REAL*4) variable which is set to the exponent that PAXIS may produce beside the axis title. A zero is returned if no exponent is drawn. See the explanation below.

Description: Normally numbering at the tick marks is represented in the form F4.2. (This is an IOH format--not a FORTRAN format.) If the numbers to be drawn are too large or too small to be drawn this way, and the standard numbering format is being used, PAXIS scales the numbers by some power of 10 and draws that power next to the title. (See the example below.) If nchar is zero then no labeling scale factor is drawn. The value of the exponent is always returned as the floating-point value of the function PAXIS.

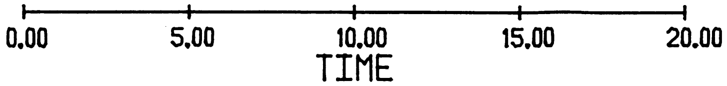
A different format for the numbering along the axis may be provided by calling PAXFRM with a new format. After PAXFRM has been called, the new format is used whenever y < 0, in which case PAXIS does not do the scaling described above.

The size of the characters for the axis title and numbering may be controlled by the PAXTTL or PAXVAL subroutines. If PAXTTL or PAXVAL is not called, PAXIS calculates sizes which depends on the values of axlth and dv.

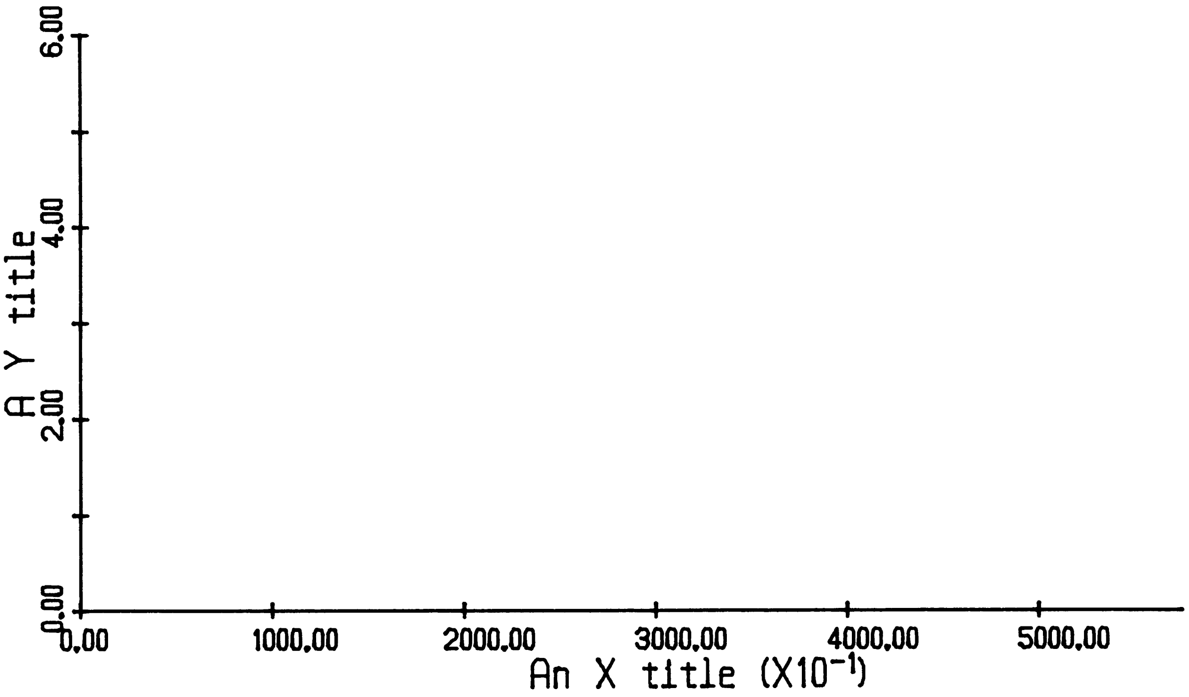
Usually, a value is drawn at every tick mark. However, if the tick marks are less than 0.6 inch apart, a number is drawn by every other tick mark. This may be changed by the PAXTIC subroutine.

The subroutine PSCALE may be used to determine appropriate values for xmin and dx.

Example: CALL PAXIS(1.,2.,’TIME’,-4,4.,0.,0.,5.,1.)



CALL PAXIS(1.,1.,’An X title’,-10,5.75,0.,0.,10000.,1.)
CALL PAXIS(1.,1.,’A Y title’,9,3.,90.,0.,2.,.5)


6.00
4.00
2.00
0.00
A Y title
0.00
1000.00
2000.00
3000.00
4000.00
5000.00
An X title (X10^{-1})

PAXTIC

Subroutine Description

Purpose: PAXTIC controls the distance between tick-mark labels on a linear axis drawn by PAXIS.

Calling Sequence:

```
CALL PAXTIC(inter)
```

Parameter:

inter is a fullword integer (INTEGER*4). If its value is N, a label is drawn by every Nth tick mark. If *inter* is zero or negative, PAXIS resets to use the default.

Description: The new value is used for all axes drawn by PAXIS until PAXTIC is called again.

PAXTTL

Subroutine Description

Purpose: PAXTTL sets the lettering size for the title on a linear axis.

Calling Sequence:

```
CALL PAXTTL(thght)
```

Parameter:

thght is the floating-point (REAL*4) lettering height in inches for an axis title. If *thght* is zero or negative, PAXIS returns to using the default.

Description: The new value for *thght* is used for all axes drawn by PAXIS until PAXTTL is called again.

PAXVAL

Subroutine Description

Purpose: PAXVAL sets the lettering size for the numbering on a linear axis.

Calling Sequence:

```
CALL PAXVAL(vhght)
```

Parameter:

vhght is the floating-point (REAL*4) lettering height in inches for the numbering on an axis tick mark. If *vhght* is zero or negative, PAXIS returns to using the default.

Description: The new value for *vhght* is used for all axes drawn by PAXIS until PAXVAL is called again.

PBOUND

Subroutine Description

Purpose: PBOUND determines if a point is inside the plotting region.

Calling Sequence:

switch = PBOUND(x,y)

Parameters:

x,*y* are the floating-point (REAL*4) coordinates of an absolute, linear-rectangular point to be tested.

Value Returned:

switch The function result is 1.0 if the point (*x*,*y*) is inside the current plotting region and 0.0 if the point is outside. The plotting region is changed by the PLTXMX and PWIND subroutines.

PCIRCL

Subroutine Description

Purpose: PCIRCL draws a circle, a spiral, or portion of either (an arc) using solid or dashed lines.

Calling Sequence:

```
CALL PCIRCL(x,y,thf,thl,rf,rl,dl,mode)
```

Parameters:

x,*y* are the floating-point (REAL*4) coordinates of the center of the circle or spiral or of a point on the circle or spiral. The meaning of the coordinates depends on the sign of *rf* as noted below.

thf is the floating-point (REAL*4) angular orientation in degrees of the initial radius *rf* with respect to the horizontal. Positive angles are measured in a counterclockwise direction.

thl is the floating-point (REAL*4) angular orientation of the final radius *rl*.

rf is a floating-point (REAL*4) number whose magnitude is the initial radius of the circle, spiral, or arc to be drawn. If *rf* is negative, *x* and *y* are the coordinates of the point on the circle or spiral at the end of the initial radius; otherwise, *x* and *y* are coordinates of the center of the circle or spiral.

rl is the floating-point (REAL*4) value of the final radius of the circle, spiral, or arc.

dl is the floating-point (REAL*4) dash length in inches for a dashed arc. If *dl* is zero, a solid arc is drawn.

mode is a fullword-integer switch whose value indicates whether *x*, *y*, *rf*, and *rl* are interpreted as absolute or relative coordinates. *x*, *y*, *rf*, and *rl* are interpreted as absolute if *mode* is zero. For any other value of *mode*, *x*, *y*, *rf*, and *rl* are treated as relative coordinates.

Description: The desired curve is drawn from (*rf*,*thf*) to (*rl*,*thl*) (in polar coordinates). If *thl* is greater than *thf*, the curve is drawn in a counterclockwise direction. If *thf* is greater than *thl*, the curve is drawn in a clockwise direction.

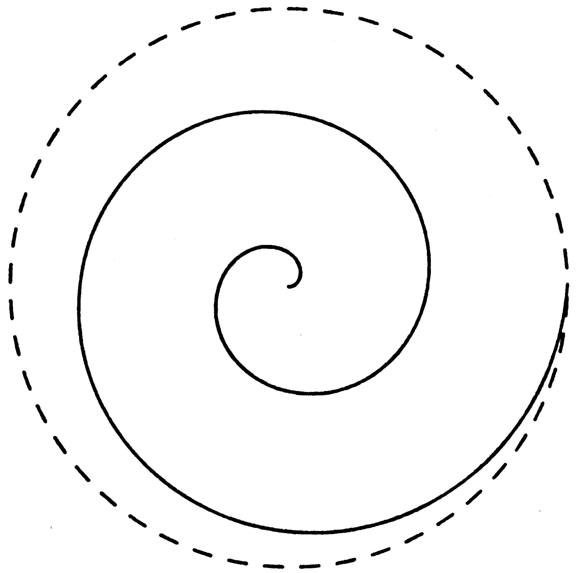
To draw a circle rf and rl should be equal to the radius, and thf and thl should be 0.0 and 360.0, respectively.

The values of x, y, rf, and rl may be either absolute or relative units, depending on the value of mode.

| To draw a spiral or dashed circle, a large number of
| points must be used. In such cases, PCIRCL may be very expensive, both in computation time and plot file space.

Example:

```
CALL PCIRCL(3.,3.,0.,360.,1.5,1.5,.1,0)
CALL PCIRCL(3.,3.,0.,720.,0.,1.5,0.,0)
```



PCTRLN, PCTRL2

Subroutine Description

Purpose: PCTRLN plots data curves with a centerline between adjacent data points contained in X,Y arrays.

Calling Sequence:

CALL PCTRLN(x,y,n,k,mode)

CALL PCTRL2(x,y,n,k,k1,mode)

Parameters:

x is an array of floating-point (REAL*4) X coordinate values. x can contain X coordinates for rectangular plotting or radial coordinates for polar plotting.

y is an array of floating-point (REAL*4) Y coordinate values. y can contain Y coordinates for rectangular plotting or angular coordinates for polar plotting.

n is a fullword integer (INTEGER*4) whose magnitude is the number of elements to be processed in the x and y arrays. If n is zero, no processing is done.

k is the fullword-integer (INTEGER*4) spacing of the elements in the x and y arrays. The elements are situated in every kth position of the array starting with the first position. k should be 1 if the elements are in adjacent positions. If every other value in an array is to be used, k should be 2. If a double-precision array is used, k must be 2 (or a multiple of 2) to use the most significant part of each element.

k1 is similar in function to k, and describes spacing of data elements in the y array so that different spacings may be specified for x and y.

mode is a fullword-integer (INTEGER*4) switch whose value indicates whether x and y are interpreted as absolute or relative coordinates. x and y are interpreted as absolute if mode is zero. For any other value of mode, x and y are treated as relative coordinates.

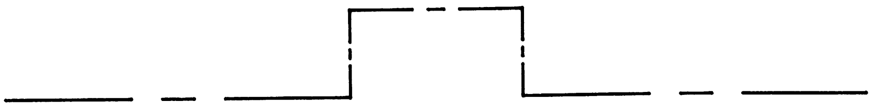
Description: The interpretation of x and y values depends on the current plotting mode. For example, in linear-polar mode, x contains radial data and y contains angular data.

Normally the pen is moved to either the first or last data point, whichever is closer to the current pen position. The rest of the points are processed in sequence. If n is negative, the first data point is processed first.

For PCTRLN, successive x and y values are k elements apart. For PCTRL2, successive x values are k elements apart, while successive y values are k1 elements apart. This is the only difference between the two.

Example:

```
REAL X(6)/2.,4.,4.,5.,5.,7./
REAL Y(6)/2.,2.,2.5,2.5,2.,2./
CALL PCTRLN(X,Y,6,1,0)
```



PDSHLN, PDSHL2

Subroutine Description

Purpose: PDSHLN and PDSHL2 plot a data curve with a sequence of dashed line segments using arrays of coordinate pairs.

Calling Sequence:

```
CALL PDSHLN(x,y,n,k,dshlth,mode)

CALL PDSHL2(x,y,n,k,k1,dshlth,mode)
```

Parameters:

x is an array of floating-point (REAL*4) X coordinate values. *x* can contain X coordinates for rectangular plotting or radial coordinates for polar plotting.

y is an array of floating-point (REAL*4) Y coordinate values. *y* can contain Y coordinates for rectangular plotting or angular coordinates for polar plotting.

n is a fullword integer (INTEGER*4) whose magnitude is the number of elements to be processed in the *x* and *y* arrays. If *n* is zero, no processing is done.

k is the fullword-integer (INTEGER*4) spacing of the elements in the *x* and *y* arrays. The elements are situated in every *k*th position of the array starting with the first position. *k* should be 1 if the elements are in adjacent positions. If every other value in an array is to be used, *k* should be 2. If a double-precision array is used, *k* must be 2 (or a multiple of 2) to use the most significant part of each element.

k1 is similar in function to *k*, and describes spacing of data elements in the *y* array so that different spacings may be specified for *x* and *y*.

dshlth is the floating-point (REAL*4) length in inches of the dash to be used. If *dshlth* is zero, 0.1 inch is assumed.

mode is a fullword-integer switch whose value indicates whether *x* and *y* are interpreted as absolute or relative coordinates. *x* and *y* are interpreted as absolute if *mode* is zero. For any other value of *mode*, *x* and *y* are treated as relative coordinates.

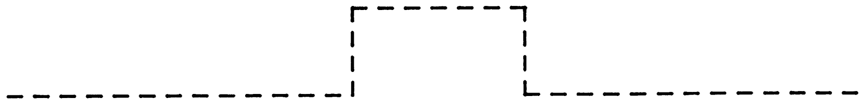
Description: The interpretation of x and y values depends on the current plotting mode. For example, in linear-polar mode, x contains radial data and y contains angular data.

Normally, the pen is moved to either the first or last data point, whichever is closer to the current pen position. The rest of the points are processed in sequence. If n is negative, the first data point is processed first.

For PDSHLN, successive x and y values are k elements apart. For PDSHL2, successive x values are k elements apart, while successive y values are k1 elements apart. This is the only difference between the two.

Example:

```
REAL X(6)/2.,4.,4.,5.,5.,7./
REAL Y(6)/2.,2.,2.5,2.5,2.,2./
CALL PDSHLN(X,Y,6,1,.08,0)
```



PELIPS

Subroutine Description

Purpose: PELIPS draws an ellipse or a portion thereof.

Calling Sequence:

```
CALL PELIPS(x,y,a,b,angle,thf,thl,mode)
```

Parameters:

x,y are floating-point (REAL*4) coordinates of the center of the ellipse if a is non-negative. If a is negative they are the linear-rectangular coordinates of a point on the circumference of the ellipse.

a,b are floating-point (REAL*4) values whose magnitudes are one-half the lengths of the ellipse axes. a is half the length of the axis oriented at the angle angle with respect to the horizontal. b is half the length of the axis normal to the a axis. If a is negative, x and y are coordinates of the point on the ellipse at the end of the initial radius; otherwise, x and y are the coordinates of the center of the ellipse.

angle is the floating-point (REAL*4) angle in degrees describing the orientation of the a axis with respect to the horizontal.

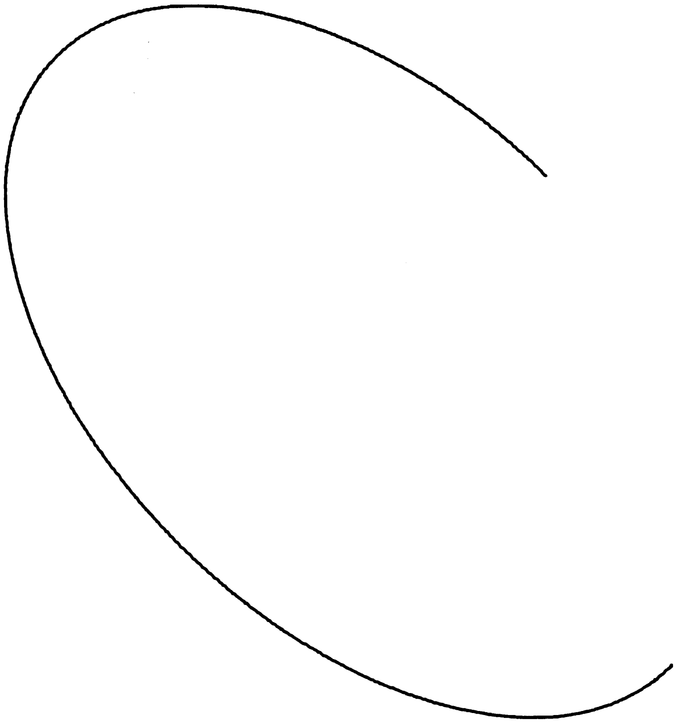
thf,thl are the floating-point (REAL*4) initial and final values in degrees of the angle corresponding to the end points of the arc to be drawn. The angles are measured counterclockwise from the a axis.

mode is a fullword-integer switch whose value indicates whether x, y, a, and b are interpreted as absolute or relative coordinates. x, y, a, and b are interpreted as absolute if mode is zero. For any other value of mode, x, y, a, and b are treated as relative coordinates.

Description: The ellipse is drawn from thf to thl. If thf is greater than thl, the curve is drawn in a counterclockwise direction; otherwise the curve is drawn in a clockwise direction.

x, y, a, and b may be either absolute or relative, depending on the value of mode. If the x and y scale factors given to PLTOFS are unequal, the ellipse is distorted (if mode is not zero).

Example: CALL PELIPS(4.,4.,1.5,2.5,45.,0.,270.,0)



PENABS

Subroutine Description

Purpose: PENABS determines the X and Y absolute linear-rectangular coordinates of the current pen position. It also determines whether the current pen position is inside the plotting region.

Calling Sequence:

switch = PENABS(x,y)

CALL PENABS(x,y)

Values Returned:

x,y are floating-point (REAL*4) variables whose values are set to the current absolute linear-rectangular X and Y pen coordinates.

switch The floating-point (REAL*4) function result is 1.0 if the current pen position is outside of the plotting region. The result is 0.0 otherwise.

PENCHG

Subroutine Description

Purpose: PENCHG changes the pen.

Calling Sequence:

```
CALL PENCHG(color,size)
```

Parameters:

color is a 4-character string containing a color name. If the name is less than four characters, it must be left-justified and padded with blanks. Only the first four characters are used. If *color* is blanks, the default color (black) is assumed.

size (optional) is a 4-character string containing a size name. If the name is less than four characters, it must be left-justified and padded with blanks. Only the first four characters are used. If *size* is blanks or omitted, the default size is assumed.

Description: The pen is changed to the one with the specified color and size. The current pen position is not changed.

The color and size are not checked by PENCHG; they are simply written into the plot description file. The colors and sizes available are listed in Appendix E.

PENDN, PENDNS

Subroutine Description

Purpose: PENDN and PENDNS move with the pen down from its current position to the point (X,Y).

Calling Sequence:

```
CALL PENDN(x,y)

CALL PENDNS(x,y)

switch = PENDN(x,y)

switch = PENDNS(x,y)
```

Parameters:

_x_,_y_ are the floating-point (REAL*4) coordinate values for the new pen position.

Value Returned:

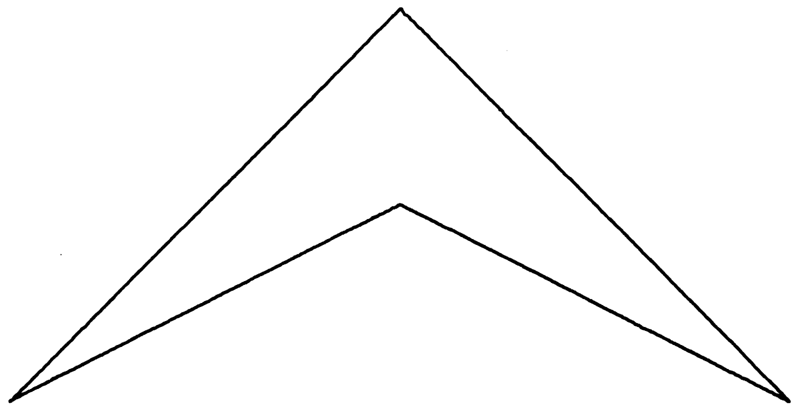
switch The floating-point (REAL*4) function result is 1.0 if the new pen position is outside the plotting region; otherwise, the result is 0.0.

Description: PENDN moves with the pen down to the absolute coordinates (_x_,_y_). PENDNS moves with the pen down to the relative coordinates (_x_,_y_).

Example:

```
CALL PENUP(2.0,2.0)
CALL PENDN(4.0,4.0)
CALL PENDN(6.0,2.0)
CALL PENDN(4.0,3.0)
CALL PENDN(2.0,2.0)
```

The resulting plot appears on the following page.



PENOPT

Subroutine Description

Purpose: PENOPT determines which of two points is nearer the current pen position.

Calling Sequence:

```
ptno = PENOPT(x1,y1,x2,y2,mode)
```

Parameters:

x1,y1,x2,y2 are the floating-point (REAL*4) coordinates of the points in question. The interpretation of the coordinates depends on the current plotting mode.

mode is a switch whose value indicates whether coordinates are interpreted as absolute or relative. x1, y1, x2, and y2 are interpreted as absolute if mode is zero. For any other value of mode, x1, y1, x2, and y2 are treated as relative coordinates.

Value Returned:

ptno The floating-point (REAL*4) function result is 1.0 if point (x1,y1) is nearest the current pen position and 2.0 otherwise.

PENREL

Subroutine Description

Purpose: PENREL determines the X and Y relative linear-rectangular coordinates of the current pen position. It also determines whether the current pen position is inside the plotting region.

Calling Sequence:

```
switch = PENREL(x,y)

CALL PENREL(x,y)
```

Parameters:

_x_,_y_ are floating-point (REAL*4) variables whose values are set to the current relative linear-rectangular X and Y pen coordinates.

Value Returned:

switch The floating-point (REAL*4) function result is 1.0 if the current pen position is outside the plotting region. The result is 0.0 otherwise.

PENUP, PENUPS

Subroutine Description

Purpose: PENUP and PENUPS move with the pen up from its current position to the point (X,Y).

Calling Sequence:

```
CALL PENUP(x,y)

CALL PENUPS(x,y)

switch = PENUP(x,y)

switch = PENUPS(x,y)
```

Parameters:

x,y are the floating-point (REAL*4) coordinate values for the new pen position.

Value Returned:

switch The floating-point function result is 1.0 if the new pen position is outside the plotting region. The result is 0.0 otherwise.

Description: PENUP moves with the pen up to the absolute coordinates (x,y). PENUPS moves with the pen up to the relative coordinates (x,y).

See the description of the PENDN subroutine for an example of the use of PENUP and PENDN.

PFDUB

Subroutine Description

Purpose: PFDUB obtains the MTS FDUB pointer or logical I/O unit number for the current PDS output stream.

Calling Sequence:

CALL PFDUB(ifdub)

Value Returned:

ifdub is a fullword-integer (INTEGER*4) variable into which the FDUB pointer or logical I/O unit number is stored.

PFLINE, PFLIN2

Subroutine Description

Purpose: PFLINE and PFLIN2 plot a data curve by drawing a smooth line through the coordinate points.

Calling Sequence:

```
CALL PFLINE(x,y,n,k,mode)

CALL PFLIN2(x,y,n,k,k1,mode)
```

Parameters:

x is a floating-point (REAL*4) array of X coordinate values. *x* can contain X coordinates for rectangular plotting or radial coordinates for polar plotting.

y is a floating-point (REAL*4) array of Y coordinate values. *y* can contain Y coordinates for rectangular plotting or angular coordinates for polar plotting.

n is an fullword integer (INTEGER*4) whose magnitude is the number of elements to be processed in the *x* and *y* arrays. If *n* is zero, no processing is done.

k is the fullword-integer (INTEGER*4) spacing of the elements in the *x* and *y* arrays. The elements are situated in every *k*th position of the array starting with the first position. *k* should be 1 if the elements are in adjacent positions. If every other value in an array is to be used, *k* should be 2. If a double-precision array is used, *k* must be 2 (or a multiple of 2) to use the most significant part of each element.

k1 is similar in function to *k*, and describes spacing of data elements in the *y* array so that different spacings may be specified for *x* and *y*.

mode is a fullword-integer switch whose value indicates whether *x* and *y* are interpreted as absolute or relative coordinates. *x* and *y* are interpreted as absolute if *mode* is zero. For any other value of *mode*, *x* and *y* are treated as relative coordinates.

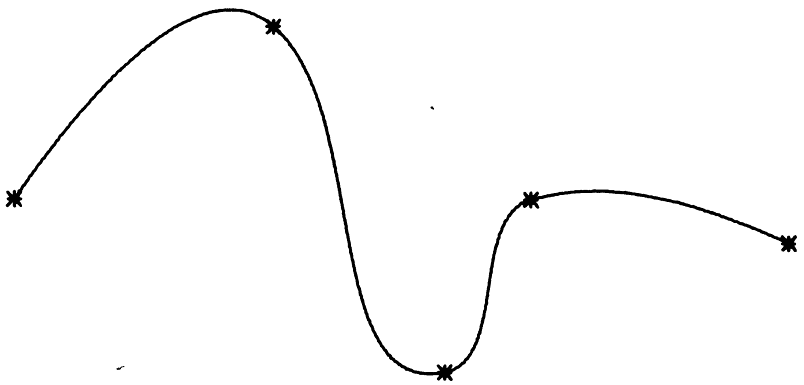
Description: This routine is a curve-smoothing routine, not a curve-fitting routine; that is, a smooth curve is drawn through every point given. The curve may be either open or closed. A closed curve is produced by making the first point equal to the last point. In either case there must be at least three unique points.

The interpretation of x and y values depends on the current plotting mode. For example, in the linear-polar mode the x and y arrays are treated as linear-radial and angular data, respectively.

For PFLINE, successive x and y values are k elements apart. For PFLIN2, successive x values are k elements apart, while successive y values are k1 elements apart. This is the only difference between the two.

Example:

```
REAL X(5)/1.,2.5,3.5,4.,5.5/
REAL Y(5)/8.,9.,7.,8.,7.75/
CALL PLINE(X,Y,5,1,-1,11,0)
CALL PFLINE(X,Y,5,1,0)
```



PFLNAM

Subroutine Description

Purpose: PFLNAM defines an output file for the PDS.

Calling Sequence:

```
CALL PFLNAM(fdname,&rc4)

ret = PFLNAM(fdname)
```

Parameter:

fdname is a character string containing a file or device name terminated by a trailing blank.

Value Returned:

ret The floating-point function result indicates whether or not the new name was accepted. The result values are:

0.0 _fdname_ is valid.
2.0 Unsuccessful - _fdname_ is illegal.

Return Codes:

0 Done.
4 Illegal _fdname_.

Description: This subroutine allows the output file to be changed from within a program instead of (or in addition to) defining it by a logical unit number on the $RUN command.

Example:

```
RET=PFLNAM('PFILE(LAST+1) ')

IF(PFLNAM('*P* ').NE.0.0) GO TO 99
```

PFMXLN

Subroutine Description

Purpose: PFMXLN defines the maximum length for lines written by the PDS.

Calling Sequence:

```
CALL PFMXLN(len)
```

Parameter:

len is a fullword-integer (INTEGER*4) length in bytes of the maximum line length for the PDS to write. *len* must be at least 255.

Description: The maximum length of a line written by the PDS into the plot description file is the smaller of the value passed to PFMXLN (if any) and the maximum length for the file or device. In no case may the length used be less than 255.

If PFMXLN is called during generation of a plot, care must be taken to ensure that no partial lines being stored by the PDS are longer than the new length. This can be done by calling PLTOUT before calling PFMXLN.

Example:

```
C LIMIT THE LINE LENGTH TO 255
      CALL PFMXLN(255)
```

PGNHDR

Subroutine Description

Purpose: PGNHDR deletes the plot-description generation message normally produced by the PDS upon generation of the first line of the plot description.

Calling Sequence:

```
CALL PGNHDR
```

Description: The PDS normally prints the message "PLOT DESCRIPTION GENERATION BEGINS" when the first line is written to the plot description file. If PGNHDR is called before the first output line is generated, this message is not printed.

PGRID

Subroutine Description

Purpose: PGRID draws a linear grid with a specified number of horizontal and vertical grid lines.

Calling Sequence:

```
CALL PGRID(x,y,xs,ys,nx,ny)
```

Parameters:

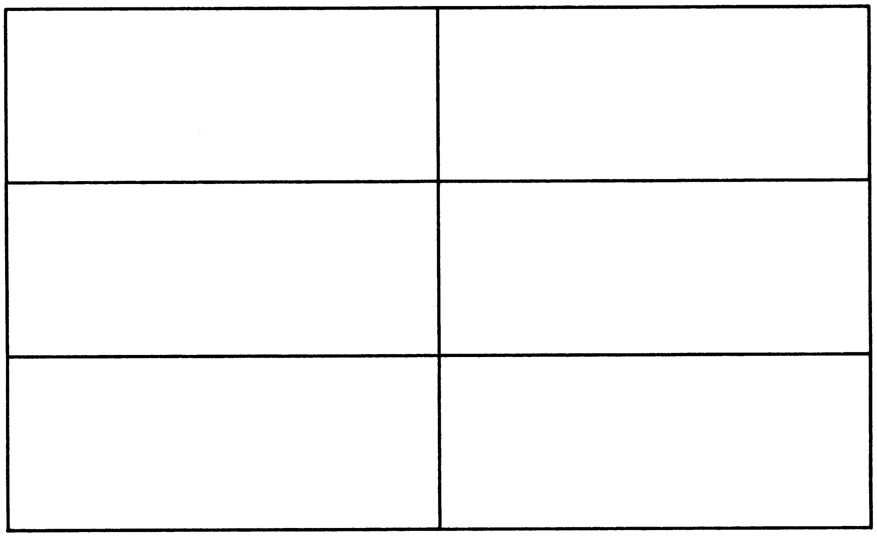
x,y are the floating-point (REAL*4) absolute linear-rectangular coordinates of the lower-left corner of the grid.

xs,ys are the floating-point (REAL*4) values of the spacings between X and Y grid lines, respectively, in inches.

nx,ny are the fullword-integer (INTEGER*4) number of grid divisions in the X and Y directions, respectively. A grid division is defined as a pair of sequential grid lines, for instance, 4 vertical grid lines constitute 3 grid divisions.

Example:

```
CALL PGRID(1.,2.,2.5,1.,2,3)
```



PINFO

Subroutine Description

Purpose: PINFO allows a program to put special plotting information into the plot description file.

Calling Sequence:

```
CALL PINFO(pfxhdr,ncnt,info,&rc4,&rc8)

lineno = PINFO(pfxhdr,ncnt,info)
```

Parameters:

pfxhdr is a four-character prefix to begin the plot-description line.
ncnt is the fullword-integer (INTEGER*4) number of bytes in *info*.
info is a sequence of bytes (for example, characters) which contain the information to be put in the line.

Value Returned:

lineno is the MTS internal line number (external line number times 1000) of the line in the file where the information was written. For tapes and sequential files, this number is a positional counter relative to the first record written.

Return Codes:

0 Done.
4 Line too long.
8 Invalid argument.

Description: The output of a call to PINFO is a line consisting of prefix *pfxhdr* followed by a halfword count of *ncnt* followed by the *ncnt* bytes of supplied information. The line is written into the current plot description file. Any partial lines generated by other PDS subroutines are written before the line generated by PINFO.

If *ncnt* is zero, then only the prefix is written. If the line exceeds the maximum length allowed by the PDS, a return code of four is given and the line is not written.

See Appendix D for more information about the format of a plot description file.

PLDNO

Subroutine Description

Purpose: PLDNO changes the MTS logical unit number to which plot-description lines are directed (default is 9 if PLDNO is not called).

Calling Sequence:

```
CALL PLDNO(num,&rc4)
```

Parameter:

num is a FDUB pointer or a fullword integer (INTEGER*4) between 0 and 19 inclusive designating the logical unit number to which plot description lines are to be directed.

Return Codes:

0 Done.
4 Illegal num.

Description: Any logical I/O unit used in a call to PLDNO must already be assigned to a file or tape.

PLGAXS

Subroutine Description

Purpose: PLGAXS plots a logarithmic axis which may have tick marks, numbering, and/or a title.

Calling Sequence:

```
CALL PLGAXS(x,y,title,nchar,axlth,angle,expmin,expfct)
```

Parameters:

x,*y* are floating-point (REAL*4) values whose magnitudes are the absolute linear-rectangular coordinates of the origin of the axis. If *x* is negative, the intermediate tick marks are deleted. If *y* is negative, the intermediate numbering is deleted. Intermediate marks and numbering are normally produced if these quantities are nonnegative.

title is a character string to be used as an axis title.

nchar is the fullword-integer (INTEGER*4) number of characters in *title*. The title and tick-mark numbering are placed on the clockwise side of the axis if the sign of *nchar* is negative. Otherwise, they are placed on the counterclockwise side of the axis. To suppress plotting of the title, *nchar* should be zero; however, as a *special case*, a *floating-point* -0.0 (hexadecimal 80000000) must be used to position the numbering on the clockwise side of the axis.

axlth is a floating-point (REAL*4) value whose magnitude is the length of the axis in inches. If *axlth* is negative then the cycle numbering (1 times 10 to some power) is deleted; otherwise, the numbering is plotted.

angle is the floating-point (REAL*4) angular orientation in degrees of the axis with respect to the horizontal (using a positive counterclockwise convention).

expmin is the floating-point (REAL*4) value of the exponent at the beginning of the first log cycle of the axis. It must be in the range of -999.0 to 9999.0 inclusive.

expfct is a floating-point (REAL*4) value whose magnitude is the axis scale factor in exponents per inch (number of cycles/inch).

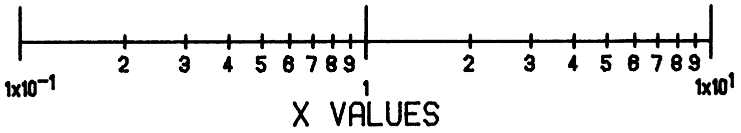
Description: The axis is divided into log cycles. At each cycle division is a number for the corresponding power of ten. The multipliers (2 through 9) of each power of ten are placed at intermediate tick marks logarithmically spaced in each cycle.

As much labeling as possible is drawn while maintaining legibility. If the cycle length is less than one inch, all intermediate tick marks and labels are deleted. If the cycle length is between one and two inches, all tick marks are drawn, but only the multiplier 5 is drawn.

The subroutine PLGSCL may be used to determine appropriate values for *expmin* and *expfct*.

Example:

```
CALL PLGAXS(1.5,1.5,'X VALUES',-8,4.,0.,-1.,.5)
```



PLGGRD

Subroutine Description

Purpose: PLGGRD plots one-half of a logarithmic grid.

Calling Sequence:

```
CALL PLGGRD(x,y,dcycle,axlth,hght,angle)
```

Parameters:

x,*y* Considering the grid before its rotation through the angle *angle*, *x* and *y* are, in magnitude, the floating-point (REAL*4) absolute linear-rectangular coordinates of the lower-left corner of the grid if *hght* is positive. If *hght* is negative, they are the coordinates of the upper-left corner of the grid. This is useful for producing a log grid at 90 degrees for log plotting along the Y axis.

dcycle is the floating-point (REAL*4) length in inches of a log cycle. This is normally the reciprocal of the log scale factor as determined by PLGSCL or by the user. Only as many log cycles as fit entirely within the log grid axis are plotted.

axlth is the floating-point (REAL*4) length in inches of the log grid axis.

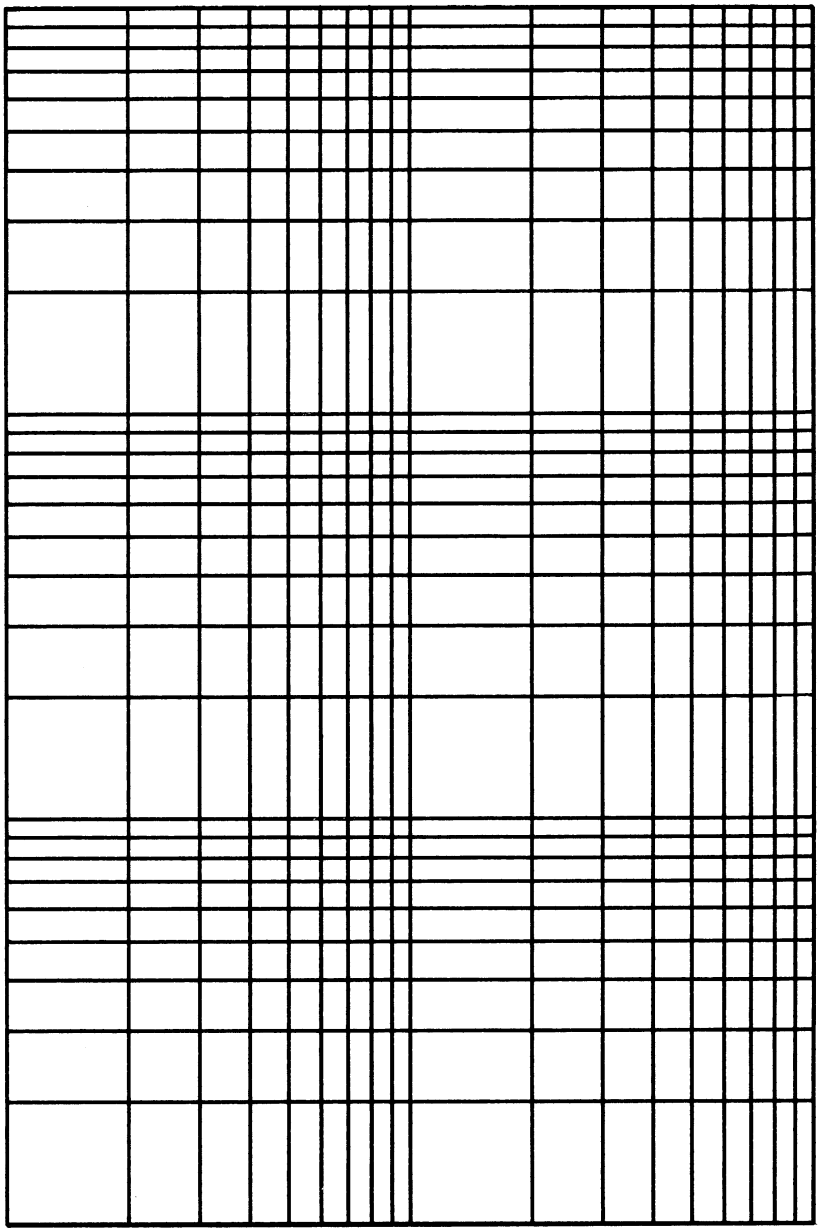
hght in magnitude, is the floating-point (REAL*4) height of the grid in inches. The sign of *hght* affects the interpretation of (*x*,*y*) as described above.

angle is the floating-point (REAL*4) angular orientation in degrees of the log grid axis with respect to the horizontal (using a positive counterclockwise convention).

Example:

```
CALL PLGGRD(1.5,1.5,2.,6.,-4.,90.)
CALL PLGGRD(1.5,1.5,2.,4.,6.,0.)
```

The first call to PLGGRD produces the horizontal lines, and the second call the vertical ones.



PLGPOL

Subroutine Description

Purpose: PLGPOL enters log-polar plotting mode.

Calling Sequence:

```
CALL PLGPOL(switch)
```

Parameter:

<u>switch</u> is a fullword integer. If <u>switch</u> is zero, the angles for plotting are in radians. If <u>switch</u> is not zero, the angles are in degrees.

Description: After PLGPOL is called, all (X,Y) coordinates given to subroutines such as PLINE, PENUPS, PDSHLN, and so on, are transformed as if the X values are radial values and the Y values are angles (in radians or degrees as indicated by the value of <u>switch</u>).

PLGSCL

Subroutine Description

Purpose: PLGSCL determines a scale factor and a minimum exponent value for one or more arrays of positive data so that the data may be plotted using a logarithmic axis.

Calling Sequence:

```
CALL PLGSCL(axlth,xmin,dx,x1,n1,k1,...,xm,nm,km)
```

Parameters:

axlth is the floating-point (REAL*4) length of the axis in inches.

xi is the first element of a floating-point (REAL*4) array of positive data values for which a log scale factor and minimum exponent value are desired. (In the above calling sequences there are "m" arrays, where "m" is a positive integer.)

ni is an fullword-integer (INTEGER*4) number of elements to be processed in the xi array.

ki is the fullword-integer (INTEGER*4) spacing of the elements in the xi array. These ni elements are situated in every kith position of the array starting with the first position. ki should be 1 if the elements are in adjacent positions. For double-precision arrays, ki must be 2 (or a multiple of 2) to use the most significant part of each element.

Values Returned:

xmin is a floating-point (REAL*4) variable whose value is set to the largest integer less than or equal to the base ten logarithm of the smallest data value of all the xi arrays. This is the exponent to be assigned to the origin of the axis. This value may be used as the xmin or ymin parameter for PLTOFS or the expmin parameter for PLGAXS.

dx is a floating-point (REAL*4) variable whose value is set to a positive scale factor in exponents per inch. If XMAX is the smallest integer greater than or equal to the base ten logarithm of the largest data value in all the xi arrays, then dx is given by (XMAX-xmin)/axlth. This value may be used as the xfact or *yfact* parameter for PLTOFS or the *expfct* parameter for PLGAXS.

Description: PLGSCL determines the minimum and maximum of all the *xi* values and then determines the scale factor and adjusted minimum value so that the data fits within the bounds of the axis. If all data values are equal, *dx* is set so that the axis consists of one log cycle. *axlth***dx* is the number of log cycles and 1.0/*dx* is the length of one cycle.

The values in the *xi* arrays must be positive. A non-positive value causes the program to terminate with an error comment.

See the section "Complete Example Programs" for examples of the use of PLGSCL.

PLINE, PLIN2

Subroutine Description

Purpose: PLINE and PLIN2 draw a data curve using arrays of coordinate pairs.

Calling Sequence:

```
CALL PLINE(x,y,n,k,ityp,isym,mode)

CALL PLIN2(x,y,n,k,k1,ityp,isym,mode)
```

Parameters:

x is an array of floating-point (REAL*4) X coordinate values. It can contain X coordinates for rectangular plotting or radial coordinates for polar plotting.

y is an array of floating-point (REAL*4) Y coordinate values. It can contain Y coordinates for rectangular plotting or angular coordinates for polar plotting.

n is a fullword integer (INTEGER*4) whose magnitude is the number of elements to be processed in the x and y arrays. If n is zero, no processing is done.

k is the fullword-integer (INTEGER*4) spacing of the elements in the x and y arrays. The elements are situated in every kth position of the array starting with the first position. k should be 1 if the elements are in adjacent positions. If every other value in an array is to be used, k should be 2. If a double-precision array is used, k must be 2 (or a multiple of 2) to use the most significant part of each element.

k1 is similar in function to k, and describes spacing of data elements in the y array so that different spacings may be specified for x and y.

ityp is a fullword integer (INTEGER*4) which specifies a plot type.

ityp = 0 draws a line plot with no symbols. PLINE connects the points with a sequence of straight line segments. If ityp is zero, the value of isym is ignored.

ityp < 0 draws a point plot with a symbol at every |ityp|th point. Straight line

segments are not drawn between points.
ityp > 0 draws a line plot with a symbol at every itypth point.

isym is a fullword integer (INTEGER*4) which specifies a symbol to be centered about the given data points. The symbols available are given below.

mode is a fullword-integer (INTEGER*4) switch whose value indicates whether x and y are interpreted as absolute or relative coordinates. x and y are interpreted as absolute if mode is zero. For any other value of mode, x and y are treated as relative coordinates.

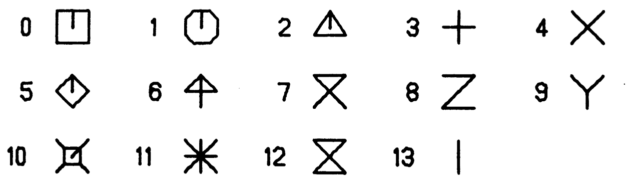
Description: The interpretation of x and y values depends on the current plotting mode. For example, in linear-polar mode x contains radial data and y contains angular data.

Normally, the pen is moved to either the first or last data point, whichever is closer to the current pen position. The rest of the points are processed in sequence. If n is negative, the first data point is processed first.

For PLINE, successive x and y values are k elements apart. For PLIN2, successive x values are k elements apart, while successive y values are k1 elements apart. This is the only difference between the two.

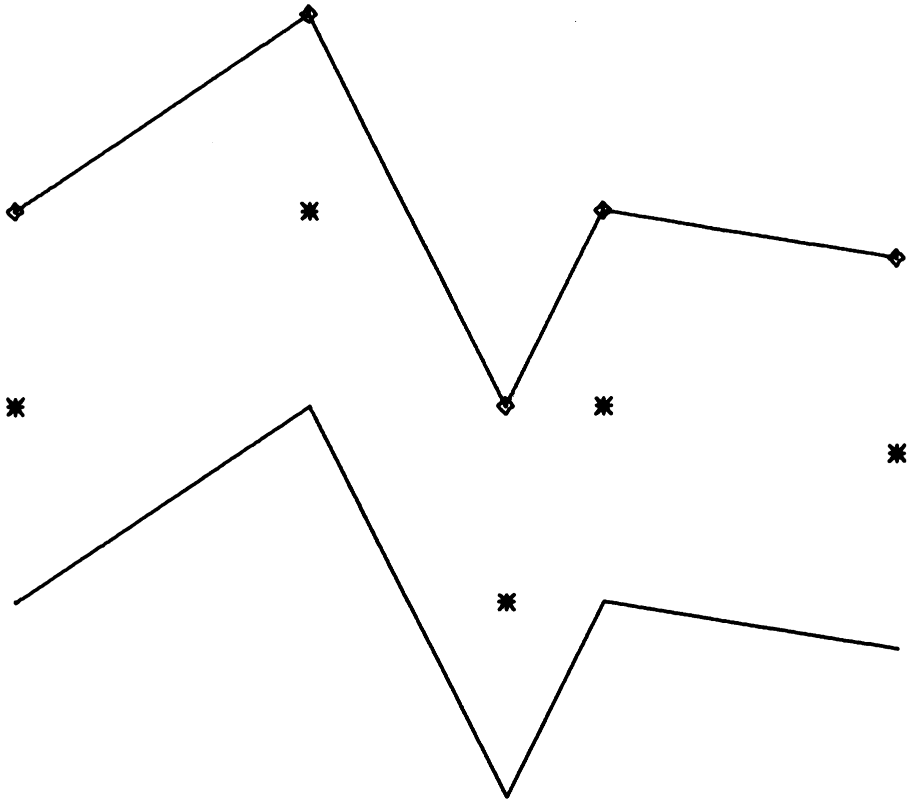
The default height of the special symbol is 0.08 inches. A different symbol size may be defined by calling the PLNSYM subroutine.

The table below shows the symbols available for use with PLINE. The number drawn to the left of each symbol is the corresponding value for isym.



Example:

```
REAL X1(5)/1.,2.5,3.5,4.,5.5/
REAL Y1(5)/8.,9.,7.,8.,7.75/
REAL Y2(5)/9.,10.,8.,9.,8.75/
REAL Y3(5)/10.,11.,9.,10.,9.75/
CALL PLINE(X1,Y1,5,1,0,0,0)
CALL PLINE(X1,Y2,5,1,-1,11,0)
CALL PLINE(X1,Y3,5,1,1,5,0)
```



PLNSYM

Subroutine Description

Purpose: PLNSYM changes the size of the symbols drawn by PLINE.

Calling Sequence:

```
CALL PLNSYM(symh)
```

Parameter:

symh is the floating-point (REAL*4) size in inches of the symbols to be drawn by PLINE or PLIN2. If symh is zero, the size is reset to the default (0.08 inches).

PLOOK

Subroutine Description

Purpose: PLOOK determines the absolute linear-rectangular coordinates corresponding to coordinates in any absolute or relative system.

Calling Sequence:

```
CALL PLOOK(x,y,u,v,mode)
```

Parameters:

x,*y* are floating-point (REAL*4) absolute or relative, linear or log, rectangular or polar coordinates of a point whose absolute linear-rectangular coordinates are desired. The values of *x* and *y* are interpreted according to the current plotting mode.

mode is a fullword-integer (INTEGER*4) switch whose value indicates whether *x* and *y* are interpreted as absolute or relative coordinates. *x* and *y* are interpreted as absolute if *mode* is zero. For any other value of *mode*, *x* and *y* are treated as relative coordinates.

Values Returned:

u,*v* are floating-point (REAL*4) variables whose values are to be set to the desired absolute linear-rectangular coordinates.

Description: PLOOK performs the coordinate transformation necessary to map (*x*,*y*) to (*u*,*v*) but does not move the pen.

PLOTCC

Subroutine Description

Purpose: PLOTCC moves the pen from its current position to the point (X,Y).

Calling Sequence:

```
switch = PLOTCC(x,y,ic)

CALL PLOTCC(x,y,ic)
```

Parameters:

x,*y* are floating-point (REAL*4) coordinate values of the new pen position. *x* and *y* may be either absolute or relative, depending on the value of *ic*.

ic is a fullword integer (INTEGER*4) specifying the type of movement desired. The valid values for *ic* are:

2 Draw (move with the pen down) to the absolute coordinate (*x*,*y*).

3 Move (with the pen up) to the absolute coordinate (*x*,*y*).

12 Draw (move with the pen down) to the relative coordinate (*x*,*y*).

13 Move (with the pen up) to the relative coordinate (*x*,*y*).

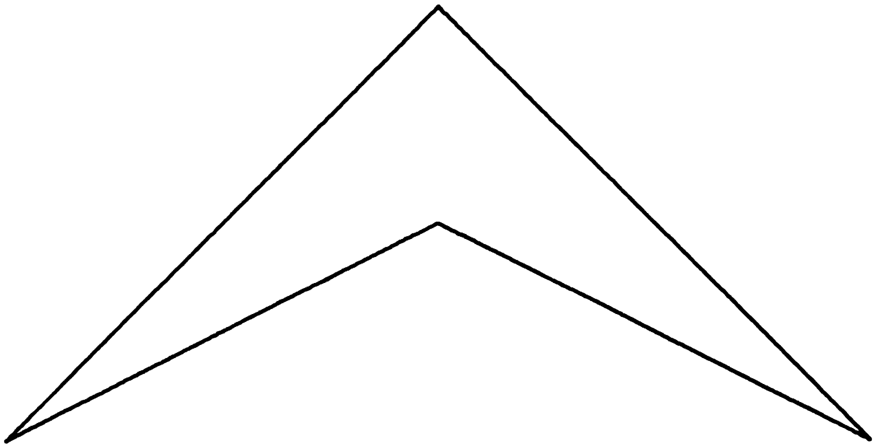
Value Returned:

switch The floating-point (REAL*4) function result is 1.0 if the new pen position is outside of the current plotting region. The result is 0.0 otherwise.

Description: The interpretation of the values of *x* and *y* depends on the current plotting mode. For example, in linear-polar mode *x* contains the radius and *y* contains the angle.

Example:

```
CALL PLOTCC(2.0,2.0,3)
CALL PLOTCC(4.0,4.0,2)
CALL PLOTCC(6.0,2.0,2)
CALL PLOTCC(4.0,3.0,2)
CALL PLOTCC(2.0,2.0,2)
```



PLSTYP

Subroutine Description

Purpose: PLSTYP sets the current plotting mode.

Calling Sequence:

```
CALL PLSTYP(itype)
```

Parameter:

itype is a fullword integer (INTEGER*4) which indicates the current plotting mode.

Description: PLSTYP sets the plotting mode according to the following table.

itype	*Plotting Mode*
0	Linear-rectangular
1	Linear-log rectangular
2	Log-linear rectangular
3	Log-log rectangular
4	Linear-polar (angles in radians)
-4	Linear-polar (angles in degrees)
5	Log-polar (angles in radians)
-5	Log-polar (angles in degrees)

The PLTREC, PLTLOG, PLTPOL, and PLGPOL subroutines may be used instead of the PLSTYP subroutine.

PLTBET

Subroutine Description

Purpose: PLTBET determines whether a plot is in progress.

Calling Sequence:

```
switch = PLTBET(0)
```

Value Returned:

switch The floating-point (REAL*4) function value is 1.0 between plots and is 0.0 during generation of a plot.

Description: This function may be used to determine whether or not a call to PLTEND needs to be given (a decision is sometimes needed for complicated programs).

PLTEND

Subroutine Description

Purpose: PLTEND ends the current plot.

Calling Sequence:

```
CALL PLTEND(&rc4)
```

Return Codes:

0 Done.
4 Not in a plot--call ignored.

Description: This subroutine terminates the current plot. It *must* be called at the end of each plot. The pen color is reset to the default and the pen is moved to the absolute coordinate (0.0,0.0) of a new plotting region. Nothing else is reset. If PLTEND is not called at the end of plotting, the plot may be incomplete.

PLTLOG

Subroutine Description

Purpose: PLTLOG enters semi-log or log-log rectangular plotting mode.

Calling Sequence:

```
CALL PLTLOG(i)
```

Parameter:

i is a positive fullword integer (INTEGER*4) whose value specifies the desired plotting mode.

i	X	Y
1	Linear	Log
2	Log	Linear
3	Log	Log

Description: PLTLOG defines the transformation mode for future use. When subroutines such as PLINE or PDSHLN are used, the (X,Y) coordinates given to them are transformed as indicated by the current plotting mode.

PLTOFS

Subroutine Description

Purpose: PLTOFS defines the parameters of a relative coordinate system to be used for mapping data into the plotting region.

Calling Sequence:

```
CALL PLTOFS(xmin,xfact,ymin,yfact,xorg,yorg)
```

Parameters:

xmin is the floating-point (REAL*4) X value in the relative system to be assigned to the absolute linear-rectangular coordinate *xorg*.

xfact is the floating-point (REAL*4) X scale factor (linear or logarithmic).

ymin is the floating-point (REAL*4) Y value in the relative system to be assigned to the absolute linear-rectangular coordinate *yorg*.

yfact is the floating-point (REAL*4) Y scale factor (linear, logarithmic, or angular).

xorg,*yorg* are floating-point (REAL*4) absolute linear-rectangular X and Y coordinate values in inches defining the position of the coordinate system origin.

Description: *xmin*, *ymin*, *xfact*, and *yfact* may be determined by the user, or calculated by the PDS. *xmin* and *ymin* correspond to the *xmin* parameter for PSCALE or PLGSCL, while *xfact* and *yfact* correspond to the *dx* parameter of those routines. Once established, the coordinate system parameter definition is carried over from plot to plot until the next call to PLTOFS. The default plotting mode is linear-rectangular. Therefore, the data passed to PLTOFS is assumed to define a linear-rectangular transformation. If a logarithmic or polar transformation is desired, one of the subroutines PLTLOG, PLTPOL, or PLGPOL must *also* be called to define the type of transformation desired. The parameters are *not* checked for consistency.

The section "Relative Coordinates" contains a more detailed description of the use of PLTOFS. Appendix F contains the transformation equations used for each of the plotting modes.

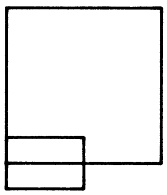
For logarithmic coordinates, *xmin*, *ymin*, *xfact*, and *yfact* are the base 10 logarithms of the minimum and the scale

| factor desired. See the section "Logarithmic Coordinates" for a detailed example.

| xfact and yfact do not affect characters drawn by PSYM.

Example:

```
REAL XBOX(5)/1.,2.,2.,1.,1./
REAL YBOX(5)/1.,1.,2.,2.,1./
CALL PLINE(XBOX,YBOX,5,1,0,0,0)
CALL PLTOFS(0.,2.,0.,3.,.5,.5)
CALL PLINE(XBOX,YBOX,5,1,0,0,1)
```



PLTOUT

Subroutine Description

Purpose: PLTOUT clears the internal PDS storage buffer.

Calling Sequence:

```
CALL PLTOUT
```

Description: Normally, the PDS collects data (coordinate pairs) passed to it and stores them in an internal buffer. This is done to minimize the number of lines written. PLTOUT writes any data currently being stored. Normally, this is done automatically when necessary, such as at the end of a plot.

PLTPOL

Subroutine Description

Purpose: PLTPOL enters linear-polar plotting mode.

Calling Sequence:

```
CALL PLTPOL(switch)
```

Parameter:

switch is a fullword integer (INTEGER*4). If switch is zero, the angles for polar plotting are in radians. If switch is nonzero, the angles are in degrees.

Description: After PLTPOL is called, all (X,Y) coordinates are transformed as if the X values are radial values and the Y values are angles (in radians or degrees as indicated by the value of switch).

PLTREC

Subroutine Description

Purpose: PLTREC enters linear-rectangular plotting mode.

Calling Sequence:

```
CALL PLTREC
```

Description: After calling PLTREC, all coordinates are treated as linear-rectangular. Since this is the default mode, PLTREC need not be called unless one of the subroutines PLTLOG, PLTPOL, or PLGPOL has been called to change the mode.

PLTSIZ

Subroutine Description

Purpose: PLTSIZ scales down an entire plot or series of plots.

Calling Sequence:

```
CALL PLTSIZ(rfact)
```

Parameter:

rfact is a floating-point (REAL*4) multiplicative scaling factor which is greater than zero and less than or equal to one. If *rfact* is outside of this range a factor of one is used.

Description: Everything within the plotting region, as well as the size of the paper used, is scaled down by *rfact*. The program should be written as if a full-sized plot is produced.

The scale factor passed to PLTSIZ remains in effect until PLTSIZ is called again. Calls to PLTSIZ are effective only between plots. A call at any other time is ignored. Overall plot scaling may also be specified when the plot is submitted.

PLTTYP

Subroutine Description

Purpose: PLTTYP determines the current plotting mode.

Calling Sequence:

```
INTEGER PLTTYP
itype = PLTTYP(0)
```

Value Returned:

itype The value of the function is a fullword integer (INTEGER*4) which indicates the current plotting mode.

Description: PLTTYP is an integer function which returns an integer value indicating the current plotting mode. It does not change the plotting mode. The table below describes the value returned.

itype	*Plotting Mode*
0	Linear-rectangular
1	Linear-log rectangular
2	Log-linear rectangular
3	Log-log rectangular
4	Linear-polar (angles in radians)
-4	Linear-polar (angles in degrees)
5	Log-polar (angles in radians)
-5	Log-polar (angles in degrees)

The subroutines PLTREC, PLTLOG, PLTPOL, PLGPOL, and PLSTYP may be used to change the plotting mode.

PLTXMX

Subroutine Description

Purpose: PLTXMX defines the maximum plot length.

Calling Sequence:

CALL PLTXMX(width)

Parameter:

width is the floating-point (REAL*4) value in inches of the maximum length in the X direction of the plotting region to be used for the next plot (or set of plots).

Description: The PDS assumes a default maximum paper width of 36.0 inches per plot. If this is not sufficient, PLTXMX must be called to change the length before the plot is generated. The maximum length allowed is 360.0 inches.

PNUMBR

Subroutine Description

Purpose: PNUMBR plots a number in a form specified by an output format (of IOH type).

Calling Sequence:

```
CALL PNUMBR(x,y,hght,numb,angle,fmt,mode,string)

nc = PNUMBR(x,y,hght,numb,angle,fmt,mode,string)
```

Parameters:

x,y are the floating-point (REAL*4) coordinates of the lower-left corner of the first character to be plotted. If either coordinate is -0.0 (hexadecimal 80000000), the string is drawn following the last character drawn by PNUMBR or PSYM.

hght is a floating-point (REAL*4) value whose magnitude is the height in inches of the characters.

numb is the number to be plotted according to the format *fmt*. It may be integer or floating-point, but must be consistent with the format.

angle is the floating-point (REAL*4) angle in degrees at which the line is drawn. *angle* is measured in a counterclockwise direction with respect to the horizontal.

fmt is an IOH format which is used to convert the number to a character string. This is *not* a FORTRAN format. For information on writing IOH formats, see the section "Introduction to IOH."

mode is a fullword-integer switch whose value indicates whether *x* and *y* are interpreted as absolute or relative coordinates. *x* and *y* are interpreted as absolute if *mode* is zero. For any other value of *mode*, *x* and *y* are treated as relative coordinates. If *mode* is negative, the number is not drawn.

Values Returned:

string (optional) is a variable or array into which the converted character string is placed. The length of this string is returned as the function value. If string is omitted, the converted string is not returned. If string is present, the converted string is returned, whether or not it is drawn. string must be large enough to contain the converted string.

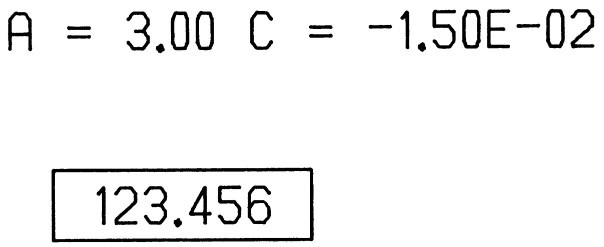
nc The function result is the fullword-integer (INTEGER*4) number of characters in the string. The string may or may not be drawn, depending on the value of mode.

Description: PNUMBR uses the IOH format passed in fmt to convert the number in numb to a character string. The resulting character string is then drawn (if mode is not negative).

The resulting character string is returned in string, if string is present, and the length of the string is returned as the function value. This allows calculation of the length of the string using PSMLEN. This may be useful for centering a number.

Example: The first two calls to PNUMBR produce the first line of output. The remainder of the program draws a rectangle and centers the number 123.456 in it. The PNUMBR subroutine is used (as a function) to convert the number to a character string. PSMLEN is used to find the length of it and PSYM is used to draw it. PNUMBR could be called again to draw the number instead of PSYM.

```
DIMENSION STR(10)
REAL XBOX(5)/2.25,3.75,3.75,2.25,2.25/
REAL YBOX(5)/.9,.9,1.3,1.3,.9/
CALL PNUMBR(2.,2.,.2,3.,0.,’"A = ",JF2.2*’,0)
CALL PNUMBR(-0.,-0.,.2,-.015,0.,’" C = ",WE1.2*’,0)
CALL PLINE(XBOX,YBOX,5,1,0,0,0)
NC = PNUMBR(3.,1.,.2,123.456,0.,’JF4.3*’,-1,STR)
CL = PSMLEN(STR,NC,.2)
CALL PSYM(3.-CL/2.,1.,.2,STR,0.,NC,0)
```



POFSAV

Subroutine Description

Purpose: POFSAV obtains the currently defined PLTOFS parameters.

Calling Sequence:

CALL POFSAV(ofpars)

Value Returned:

ofpars is a floating-point (REAL*4) array into which the currently defined PLTOFS parameters are placed. The array must be dimensioned to at least 6.

Description: The current coordinate transformation parameters are placed in the array *ofpars*. These are the most recent parameters passed to the PLTOFS subroutine. They are stored in *ofpars* in the order *xmin*, *xfact*, *ymin*, *yfact*, *xorg*, *yorg*.

POFWIN

Subroutine Description

Purpose: POFWIN disables use of the window defined by PWIND.

Calling Sequence:

```
CALL POFWIN
```

Description: The use of the current window (plotting region) defined by the PWIND subroutine is disabled. The pen may move anywhere in the entire plotting region.

See the descriptions of the PWIND and PONWIN subroutines for more information.

POLGRD

Subroutine Description

Purpose: POLGRD plots a solid or dashed linear or logarithmic polar grid.

Calling Sequence:

```
CALL POLGRD(x,y,dr,rl,thpf,dthp,thpl,dash)
```

Parameters:

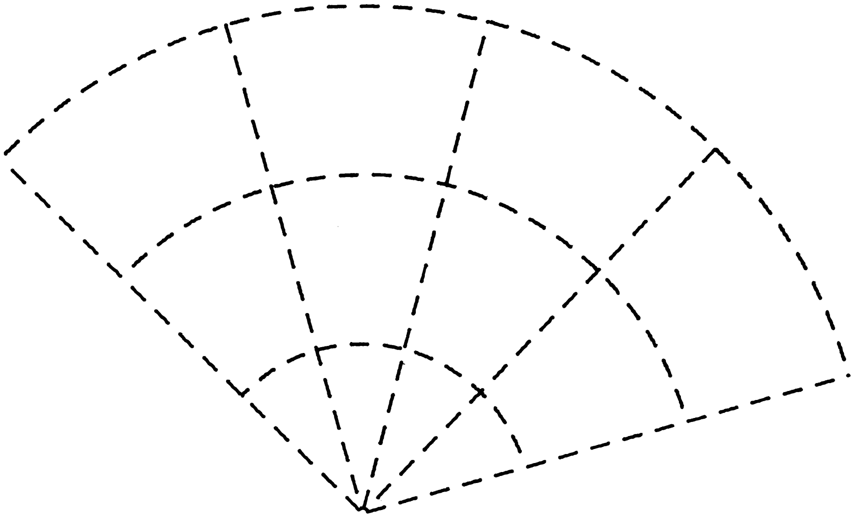
- _x_,_y_ are the floating-point (REAL*4) absolute linear-rectangular coordinates of the grid origin (the center of the circles whose arcs are used to construct the polar grid).
- _dr_ is a floating-point (REAL*4) value whose magnitude is equal to the radial spacing between arcs in inches. The radii are solid unless _dr_ is negative, in which case they are dashed. A zero _dr_ causes no grid to be produced.
- _rl_ is a floating-point (REAL*4) value whose magnitude is the radius in inches of the outermost arc. (This is usually the same magnitude as the argument _r_ for PTHAXS.) A linear grid is produced unless _rl_ is negative, in which case the magnitude of _dr_ is interpreted as the length of a log cycle and intermediate arcs are logarithmically distributed over each cycle. If _rl_ is zero, no grid is drawn.
- _thpf_ is the floating-point (REAL*4) angular orientation in degrees with respect to the horizontal of the first radial line of a sequence of radial grid lines to be drawn.
- _dthp_ is a floating-point (REAL*4) value whose magnitude is equal to the angular spacing in degrees between successive radial lines. If _dthp_ is negative, the circular arcs are dashed. If _dthp_ is positive, the arcs are solid. If _dthp_ is zero, no radial lines are drawn.
- _thpl_ is the floating-point (REAL*4) angular orientation in degrees with respect to the horizontal of the last radial line of the sequence. _thpl_ must be greater than or equal to _thpf_. If they are equal, no grid is drawn.
- _dash_ is the floating-point (REAL*4) length in inches of the dashes for the dashed lined portions of the grid. If _dash_ is zero, a dash length of 0.1 inches is used.

Description: A polar grid is a set of concentric circles about the point (x,y), and a set of radial lines extending from (x,y).

The circles are either evenly spaced or logarithmically spaced, depending on the sign of rl. The arcs are either solid or dashed, depending on the sign of dthp. If rl is not a multiple of dr, as many circular arcs as possible are drawn. Only the portion of the arcs between the initial and final radial lines are drawn.

The radial lines are evenly spaced. They may be either dashed or solid, depending on the sign of dr. If (thpl - thpf) is not a multiple of dthp, as many radial lines as possible are drawn.

Example: CALL POLGRD(3.,3.,-1.,3.,15.,-30.,135.,.1)



PONWIN

Subroutine Description

Purpose: PONWIN enables use of the window defined by PWIND.

Calling Sequence:

```
CALL PONWIN
```

Description: The use of the current window (plotting region) defined by the PWIND subroutine is enabled. The pen is confined to movement within this region.

See the descriptions of the PWIND and POFWIN subroutines for more information.

PPOLY

Subroutine Description

Purpose: PPOLY draws a polygon.

Calling Sequence:

```
CALL PPOLY(x,y,r,n,th,mode)
```

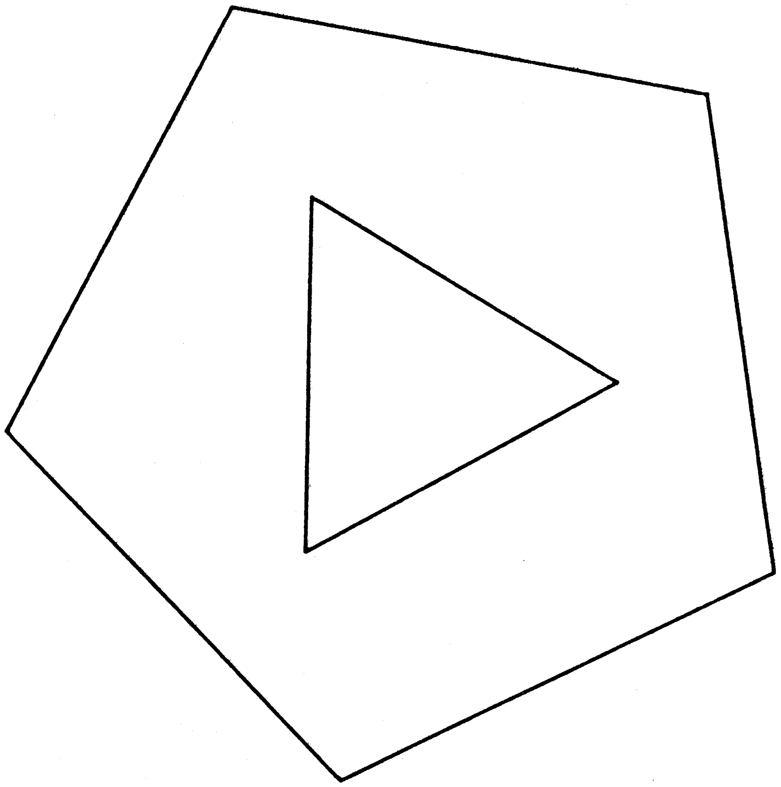
Parameters:

- *x*,*y* are the floating-point (REAL*4) coordinates of the center of the polygon.
- *r* is the floating-point (REAL*4) radius of the polygon (the radius of the circle that circumscribes the polygon).
- *n* is the integer number of sides of the polygon.
- *th* is the floating-point (REAL*4) angular orientation (in degrees) of the initial vertex with respect to the horizontal. Positive angles are measured in the counterclockwise direction.
- *mode* is a fullword integer switch whose value indicates whether *x*, *y*, and *r* are to be interpreted as absolute or relative coordinates. If *mode* is zero, the coordinates are absolute; otherwise, the coordinates are relative.

Description: A polygon of *n* sides will be drawn with the center at *x*,*y*, a radius of *r*, and one vertex at an angle of *th* from the horizontal. Linear-rectangular mode will be used.

Example: The example below draws a 2-inch pentagon with a 1-inch triangle inside of it.

```
CALL PPOLY(4.,4.,2.,5,45.,0)
CALL PPOLY(4.,4.,1.,3,0,0)
```



PSALPH

Subroutine Description

Purpose: PSALPH changes the values of parameters for the current character set.

Calling Sequence:

```
CALL PSALPH(what,value,oldval,&rc4)
```

Parameters:

what is a 4-character name or a fullword-integer number of the parameter to be changed.

value is the new value to be assigned to that parameter. It is either one or two fullword integers (INTEGER*4) or a single floating-point (REAL*4) number, as appropriate for what.

Value Returned:

oldval is set on return to the previous value of the parameter. It should have the same size and type as value.

Return Codes:

0 Successful return.

4 Unsuccessful. what was not a parameter, value was invalid for what, or space could not be obtained for copying the standard alphabet.

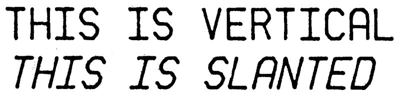
Description: The parameter specified by what has its value changed to value, and its previous value is returned in oldval. The table below describes the values for what. In the "Type" column, F means a fullword integer (INTEGER*4) and E means fullword floating-point (REAL*4).

	what		
Name	Number	Type	Parameter Changed
HSPC	1	F	HSPAC
VSPC	2	F	VSPAC
SUPS	3	2F	PSUP
SUBS	4	2F	PSUB
SHGT	5	E	SHGT
SCAL	6	E	SCAL
HSCL	7	E	SSCAL
ANGL	8	E	ANGL
REFP	9	2F	REF
LTRS	10	F	SWITCH =0 letterspacing off ≠0 letterspacing on
LSPC	11	F	LSPAC

See the description of PALPHA for an explanation of the parameters. The value restrictions listed there are enforced by PSALPH.

Example:

```
CALL PSYM(2.,2.,.15,'THIS IS VERTICAL',0.,16,0)
CALL PSALPH('ANGL',15.,OLDANG)
CALL PSYM(2.,1.75,.15,'THIS IS SLANTED',0.,15,0)
CALL PSALPH('ANGL',OLDANG,DUMMY)
```



PSCALE

Subroutine Description

Purpose: PSCALE determines a scale factor and a minimum value for one or more arrays of data so that the data may be plotted using a linear axis.

Calling Sequence:

```
CALL PSCALE(axlth,dv,xmin,dx,x1,n1,k1,...,xm,nm,km)
```

Parameters:

axlth is the floating-point (REAL*4) length of the axis in inches.

dv is the floating-point (REAL*4) distance in inches between tick marks (division marks) on the axis.

xi is the ith floating-point (REAL*4) array of data for which a scale factor and adjusted minimum are desired. In the above calling sequence there are "m" arrays, where "m" is at least one.

ni is the fullword-integer (INTEGER*4) number of elements to be processed in the xi array.

ki is the fullword-integer (INTEGER*4) spacing of the elements in the xi array. These ni elements are situated in every kith position of the array starting with the first position. ki should be 1 if the elements are in adjacent positions. For double-precision arrays, ki must be 2 (or a multiple of 2) to use the most significant part of each element.

Values Returned:

xmin is a floating-point (REAL*4) variable whose value is set to an adjusted minimum value of all of the xi arrays. The units of xmin are the units of the xi arrays and the value of xmin is the value of the axis variable at the first tick mark. This value may be used as the xmin or ymin parameter for PLTOFS or the xmin parameter for PAXIS.

dx is a floating-point (REAL*4) variable whose value is set to a scale factor. dx is such that the increment in the axis variable per division is either 1, 2, 4, 5, or 8 times 10 to some integer power. The units of dx are the units of the _xi_ arrays per inch. This value may be used as the _xfact_ or _yfact_ parameter for PLTOFS or the _dx_ parameter for PAXIS.

Description: PSCALE determines the minimum and maximum of all the _xi_ values and then determines the scale factor and adjusted minimum value so that the data fits within the bounds of the axis. If all of the data values are equal, _xmin_ and _dx_ are set by PSCALE such that if coordinates equal to the data values are given to the PDS, they are mapped to the origin of their respective axes.

See the section "Complete Example Programs" for examples using PSCALE.

PSHADE, PSHAD2

Subroutine Description

Purpose: PSHADE and PSHAD2 shade an arbitrary polygon.

Calling Sequence:

```
CALL PSHADE(x,y,n,k,angle,dist,mode)

CALL PSHAD2(x,y,n,k,k1,angle,dist,mode)
```

Parameters:

x is a floating-point (REAL*4) array of X coordinate values defining the vertices of an arbitrary polygon. *x* can contain X coordinates for rectangular plotting or radial coordinates for polar plotting.

y is a floating-point (REAL*4) array of Y coordinate values. *y* can contain Y coordinates for rectangular plotting or angular coordinates for polar plotting.

n is the fullword-integer (INTEGER*4) number of elements in the *x* and *y* arrays.

k is the fullword-integer (INTEGER*4) spacing of the elements in the *x* and *y* arrays. The elements are situated in every *k*th position of the array starting with the first position. *k* should be 1 if the elements are in adjacent positions. If every other value in an array is to be used, *k* should be 2. If a double-precision array is used, *k* must be 2 (or a multiple of 2) to use the most significant part of each element.

k1 is similar in function to *k*, and describes spacing of data elements in the Y array so that different spacings may be specified for *x* and *y*.

angle is the floating-point (REAL*4) angle in degrees at which the shading lines are drawn using a positive counterclockwise convention.

dist is the floating-point (REAL*4) distance between shading lines in inches.

mode is a fullword-integer (INTEGER*4) switch whose value indicates whether *x* and *y* are interpreted as absolute or relative coordinates. *x* and *y* are interpreted as absolute if *mode* is zero. For any other value of *mode*, *x* and *y* are treated as relative coordinates.

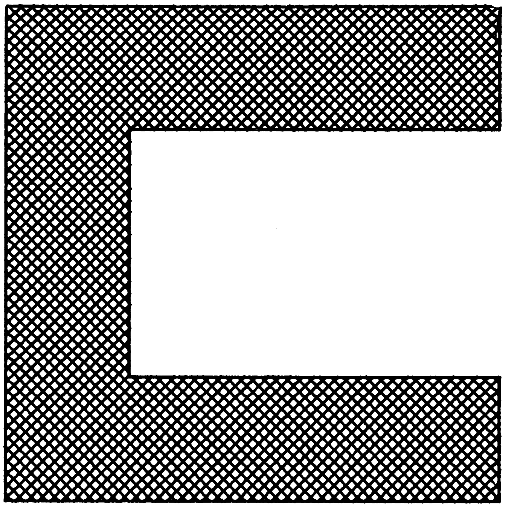
Description: PSHADE shades an arbitrary polygon by drawing a set of parallel lines at the angle angle and dist inches apart.

The polygon may be of any shape. The sides of the polygon may cross each other. The first vertex is assumed to be connected to the last vertex by a single straight line (if they are not the same). There must be at least three distinct vertices.

For PSHADE, successive x and y values are k elements apart. For PSHAD2, successive x values are k elements apart, while successive y values are k1 elements apart. This is the only difference between the two.

Example:

```
REAL X(9)/2.,5.,5.,2.75,2.75,5.,5.,2.,2./
REAL Y(9)/2.,2.,2.75,2.75,4.25,4.25,5.,5.,2./
CALL PLINE(X,Y,9,1,0,0,0)
CALL PSHADE(X,Y,9,1,45.,.05,0)
CALL PSHADE(X,Y,9,1,-45.,.05,0)
```



PSMGEN

Subroutine Description

Purpose: PSMGEN enables or disables generation of PSYM records.

Calling Sequence:

```
CALL PSMGEN(switch)
```

Parameter:

switch is a fullword integer (INTEGER*4). If *switch* is zero, PSYM lines are not generated. If it is nonzero, PSYM lines are generated.

Description: A PSYM line is an abbreviated form of the code for a character string in a plot description file. In a plot with a large amount of text, the use of PSYM lines can save a considerable amount of file space. If generation of PSYM lines is on, PSYM produces PSYM lines instead of expanding character strings whenever the standard character set is in use. The default for PSYM line generation is on.

PSMLEN

Subroutine Description

Purpose: PSMLEN determines the length in inches that a character string would be if plotted by PSYM.

Calling Sequence:

```
size = PSMLEN(string,length,height)
```

Parameters:

string is the character string whose length is to be determined.

length is the fullword-integer (INTEGER*4) number of characters in the string.

height is the floating-point (REAL*4) height in inches for *string*.

Value Returned:

size The function result is the length of the string in inches.

Description: *string* has the same format as a character string given to PSYM. (The same string could be passed to PSYM.) The length of *string* as it would be plotted by PSYM is determined, with the exception that carriage returns in *string* are ignored.

PSYM

Subroutine Description

Purpose: PSYM produces text on a plot. This routine is used for all character sets. See the description of PALPHA for information on alternate character sets.

Calling Sequence:

```
CALL PSYM(x,y,height,string,angle,length,mode,&rc4)
```

Parameters:

x,y	are the floating-point (REAL*4) coordinates of the first character to be drawn. For most character sets, including the standard one, this is the lower-left corner of the first character. If either coordinate is -0.0 (hexadecimal 80000000), PSYM continues from the end of the last character string drawn.
height	is the floating-point (REAL*4) height in inches at which the string is drawn.
string	is the character string to be drawn.
angle	is the floating-point (REAL*4) angle in degrees of the character string (using a positive counterclockwise convention).
length	is the fullword-integer (INTEGER*4) number of characters in string.
mode	is a fullword-integer (INTEGER*4) switch whose value indicates whether x and y are interpreted as absolute or relative coordinates. x and y are interpreted as absolute if mode is zero. For any other value of mode, x and y are treated as relative coordinates.

Return Codes:

0	Successful return.
4	Unsuccessful return; string not drawn. Either the parameters are in error (for instance, calling PSYM for the first time with x or y equal to -0.0), or there is an error in a user-defined character set.

Description: PSYM draws string with the height specified, beginning at (x,y), with the angle angle.

There are four special characters which produce carriage returns, backspaces, subscripts, and superscripts:

X’15’: Carriage return. This causes an immediate carriage return as on a typewriter, where the margin is defined by the last previous carriage return, or the last explicit (not -0.0) X,Y pair.

X’16’: Backspace. This produces a one-character backspace. If the alphabet currently being used has letterspacing on, this may be farther than one character back. See the description of PALPHA for details.

X’09’: Up. If encountered while drawing text normally, succeeding text is drawn as superscripts. If encountered while drawing superscripts, text is drawn at the next higher superscript level. If encountered while drawing subscripts, drawing continues at the next higher level of subscripts (or back to normal if there was only one level of subscripts).

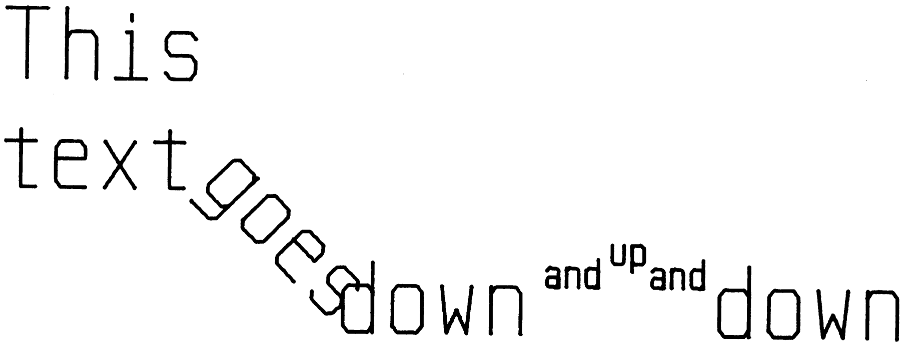
X’38’: Down. If encountered while drawing text normally, succeeding text is drawn as subscripts. If encountered while drawing subscripts, text is drawn at the next lower subscript level. If encountered while drawing superscripts, succeeding text is drawn at the next lower level of superscripts (or back to normal if there was only one level of superscripts).

The exact location and height of subscripts and superscripts are determined by the particular alphabet. Multiple levels of subscripts and superscripts are permitted; all levels are drawn with the same height. This height is some fraction of height. See the description of PALPHA for details.

The scale factors set by a call to PLTOFS do not affect characters drawn by PSYM.

Example:

```
INTEGER FIRST(3)/’This’,Z15A385A7,ZA3404040/
INTEGER THIRD(5)/’down’,Z09819584,Z09A49738,Z81958438,
                 ’down’/
CALL PSYM(1.,6.,.4,FIRST,0.,9,0)
CALL PSYM(-0.,-0.,.4,’goes’,-45.,4,0)
CALL PSYM(-0.,-0.,.4,THIRD,0.,20,0)
```


This
text
goes
down
and
up
and
down

PSYMLC

Subroutine Description

Purpose: PSYMLC obtains the coordinates of the end of the last character string drawn by PSYM or PNUMBR.

Calling Sequence:

```
CALL PSYMLC(x,y)
```

Values Returned:

x,*y* are floating-point (REAL*4) variables set to the absolute coordinates of the point at which PSYM would begin lettering if called with coordinates (-0.0, -0.0). -0.0 (hexadecimal 80000000) is returned if PSYM has never been called.

PTHAXS

Subroutine Description

Purpose: PTHAXS plots a solid or dashed polar axis with angular labeling.

Calling Sequence:

```
CALL PTHAXS(x,y,r,thpf,dthp,thpl,thstr,delth,dash,fmt)
```

Parameters:

_x_,_y_ are, in magnitude, the floating-point (REAL*4) absolute linear-rectangular coordinates in inches of the axis origin. If _x_ is negative, _fmt_ is assumed to be 8 characters long (2 fullwords). The magnitude of _x_ is used for the X coordinate of the axis origin.

_r_ is a floating-point (REAL*4) value whose magnitude is equal to the radius in inches of the angular axis lines to be drawn. This is usually the same in magnitude as the argument _rl_ for POLGRD. If _r_ is zero, no axis is drawn. If _r_ is negative, the radial lines of length _r_ are not drawn (in which case the magnitude of _r_ is used only as a reference for placement of numbering). A positive value of _r_ causes the radial lines to be drawn.

thpf is the floating-point (REAL*4) angular orientation in degrees with respect to the horizontal of the first radial line of a sequence of radial lines to be produced.

dthp is the floating-point (REAL*4) angular spacing in degrees between successive radial lines. If _dthp_ is zero, no axis is drawn.

thpl is the floating-point (REAL*4) angular orientation in degrees with respect to the horizontal of the last radial line of the sequence. _thpl_ must be greater than or equal to _thpf_. If they are equal, then only a label plus possibly one radial line is drawn.

thstr is the floating-point (REAL*4) value in degrees or radians of the angle variable corresponding to _thpf_.

delth is a floating-point (REAL*4) increment in the angle variable corresponding to the increment of _dthp_. The units of _delth_ must be consistent with those of _thstr_.

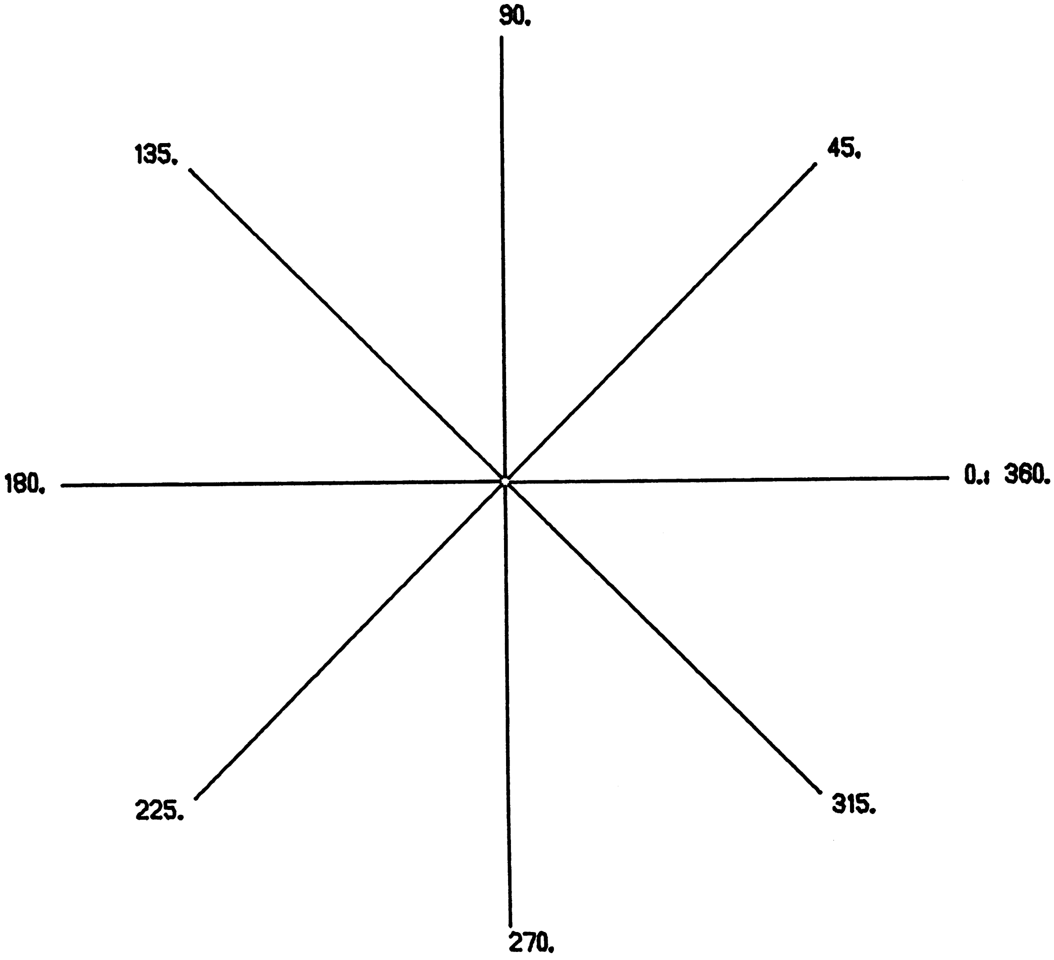
dash is the floating-point (REAL*4) length of dashes to be drawn for dashed radial lines. If dash is zero, solid lines are drawn.

fmt is a fullword IOH floating-point format term describing the form of the axis labeling. If the value of fmt is zero, then F3.1 is used. PTHAXS automatically supplies the IOH W and J modifiers for fullword data and minimization of field width and the terminating asterisk. If x is negative, fmt is assumed to be an 8-character format term that is fullword aligned. All 8 characters must be specified. For information on IOH formats, see the section "Introduction to IOH."

Description: The values for thpf, dthp, and thpl describe physical angles on the paper. A polar axis is a set of radial lines extending from the axis origin (x,y). The first line is at angle thpf and successive radial lines are spaced dthp degrees apart. As many lines are drawn as can fit in the range (thpl - thpf). At the end of each line a number corresponding to an angle variable is drawn. The value of thstr is drawn at the end of the line oriented at the angle thpf. The label at each successive radial line is incremented by delth. If (thpl - thpf) is 360.0, a full polar axis is drawn with the radial line at thpf having both thstr and the final value of the angle variable drawn at its end.

Example:

```
CALL PTHAXS(5.,5.,2.5,0.,45.,360.,0.,45.,0.,’F3.0’)
```


90.
135.
45.
180.
0.; 360.
225.
315.
270.

PWIND

Subroutine Description

Purpose: PWIND defines a window (a rectangular subregion of the plotting region) which may be used to confine the movement of the pen. This may be used to ignore errant data points that fall beyond the bounds of X or Y axes.

Calling Sequence:

```
CALL PWIND(x,y,delx,dely)
```

Parameters:

x,y are the floating-point (REAL*4) absolute linear-rectangular X and Y coordinates of the lower-left corner of the window.

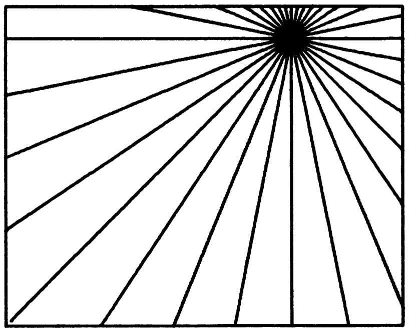
delx is the floating-point (REAL*4) window length in inches.

dely is the floating-point (REAL*4) window height in inches.

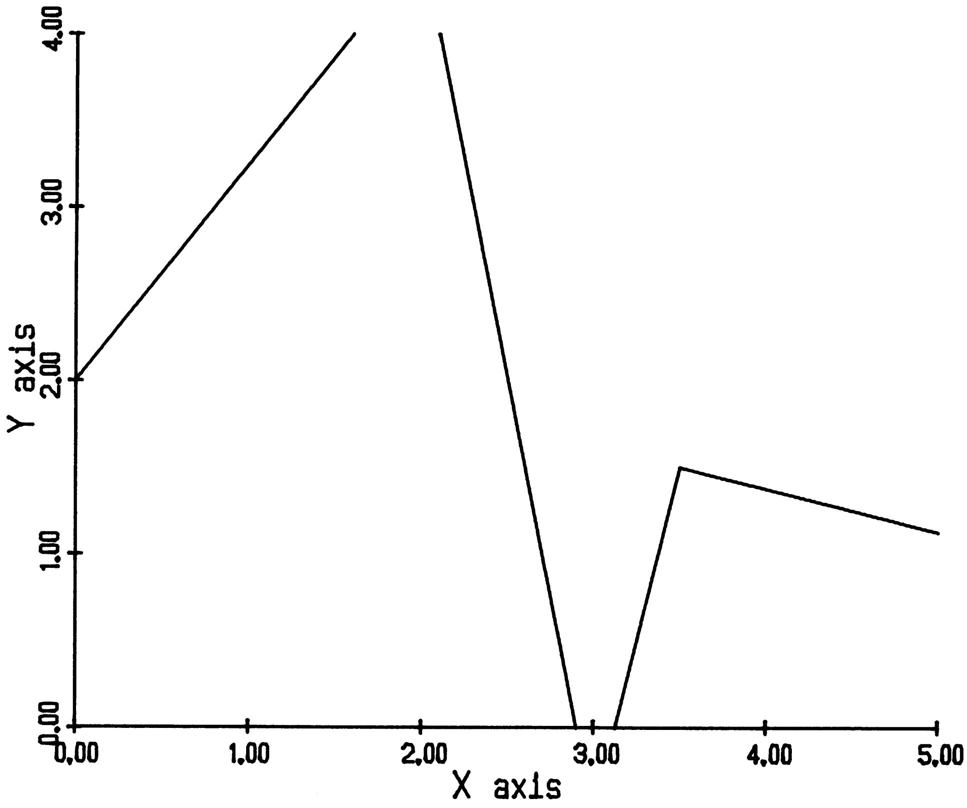
Description: PWIND is used to define the window. When the window is enabled, pen movements are confined to the window region. When the window is disabled, the pen is free to move anywhere within the entire plotting region. If a window is enabled and a new one is defined, it replaces the current one and is automatically enabled. The defined window is not used unless PONWIN is called. POFWIN disables use of the window.

Examples:

```
  CALL PENUP(.8,3.5)
  CALL PENDN(3.3,3.5)
  CALL PENDN(3.3,5.5)
  CALL PENDN(.8,5.5)
  CALL PENDN(.8,3.5)
  CALL PLTOFS(0.,1.,0.,1.,2.6,5.3)
  CALL PWIND(.8,3.5,2.5,2.)
  CALL PONWIN
  TH = 0.
1 T=TH*.0174533
  CALL PENUPS(2.5*COS(T),2.5*SIN(T))
  CALL PENDNS(2.5*COS(T+3.141593),2.5*SIN(T+3.141593))
  TH=TH+11.25
  IF(TH.LT.180.) GO TO 1
  CALL POFWIN
```



```
REAL XR(5)/0.,2.,3.,3.5,5.5/
REAL YR(5)/2.,4.5,-.5,1.5,1./
CALL PAXIS(1.,1.,'X axis',-6,5.,0.,0.,1.,1.)
CALL PAXIS(1.,1.,'Y axis',6,4.,90.,0.,1.,1.)
CALL PWIND(1.,1.,5.,4.)
CALL PONWIN
CALL PLTOFS(0.,1.,0.,1.,1.,1.)
CALL PLINE(XR,YR,5,1,0,0,1)
CALL POFWIN
```


4.00
3.00
2.00
1.00
0.00
Y axis
0.00
1.00
2.00
3.00
4.00
5.00
X axis

PXMARG

Subroutine Description

Purpose: PXMARG defines a right margin.

Calling Sequence:

```
CALL PXMARG(xmarg)
```

Parameter:

xmarg is a floating-point (REAL*4) right margin value in inches which is to be added to the maximum X excursion of the pen to produce the right boundary of the plot.

Description: *xmarg* must be between 0.0 and 10.0. If *xmarg* is outside this range, a value of 2.0 is used. The default margin is two inches.

The margin set by PXMARG remains in effect until PXMARG is called again.

APPENDIX A: CHARACTER SET DEFINITIONS

This appendix contains tables defining the character sets available in the PDS. Character strings may be drawn using any of these sets. The PALPHA subroutine selects the set for use by the PSYM subroutine.

Table 1 defines the Standard character set. This is the set used by PSYM if PALPHA is not called to select an alternate set. All of the characters on the TN print train are in this set. Table 2 defines a Sanserif.1 character set that produces better appearing characters with a minimal increase in plotting time. The use of this character set is recommended.

For the Standard and Sanserif.1 character sets, a grid layout is included at the end of this appendix showing exactly how each character is drawn (see the PALPHA subroutine for a complete explanation of the grid). The small number next to each character is the hexadecimal representation of that character.

The remainder of the tables each define a character set. The values given below each table describe the alphabet parameters. These parameters are described in the PALPHA subroutine description. Most of them may be changed by the PSALPH subroutine. Wherever possible, the characters in each set have the obvious correspondence to a keypunch or terminal keyboard. For example, the character "A" is in the same position in all character sets which have an "A". In some cases, such as Japanese, this is impossible. In these cases, it may be necessary to use the hexadecimal representation to obtain the characters. This representation is found by concatenating the row number and column number from the table. For instance, the hexadecimal for the letter "A" from the Standard character set is C1.

The Japanese alphabet is divided into seven tables, Hiragana, Katakana, and Kanji1 to Kanji5 since the set contains more symbols than fit into a single table. In the Kanji tables, the symbols are grouped by ascending number of strokes in the radical. Within each group, the symbols are ordered by increasing complexity.

For all character sets, if the character exists on the keyboard of a keypunch or terminal, it may be used directly. Otherwise, if the language used to call the subroutines does not support hexadecimal representation, then direct hexadecimal entry from the terminal must be used. See MTS Volume 4, Terminals and Tapes, for information on entering characters in hexadecimal.

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4											¢	.	<	(	+	|
5	ℇ										!	$	*	)	;	¬
6	-	/										,	%	_	>	?
7											:	#	@	'	=	"
8		a	b	c	d	e	f	g	h	i		{	≤	(	+	+
9		j	k	l	m	n	o	p	q	r		}	⌑	)	±	■
A	-	°	s	t	u	v	w	x	y	z		└	┌	[	≥	●
B	0	1	2	3	4	5	6	7	8	9		┘	┐	]	≠	−
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

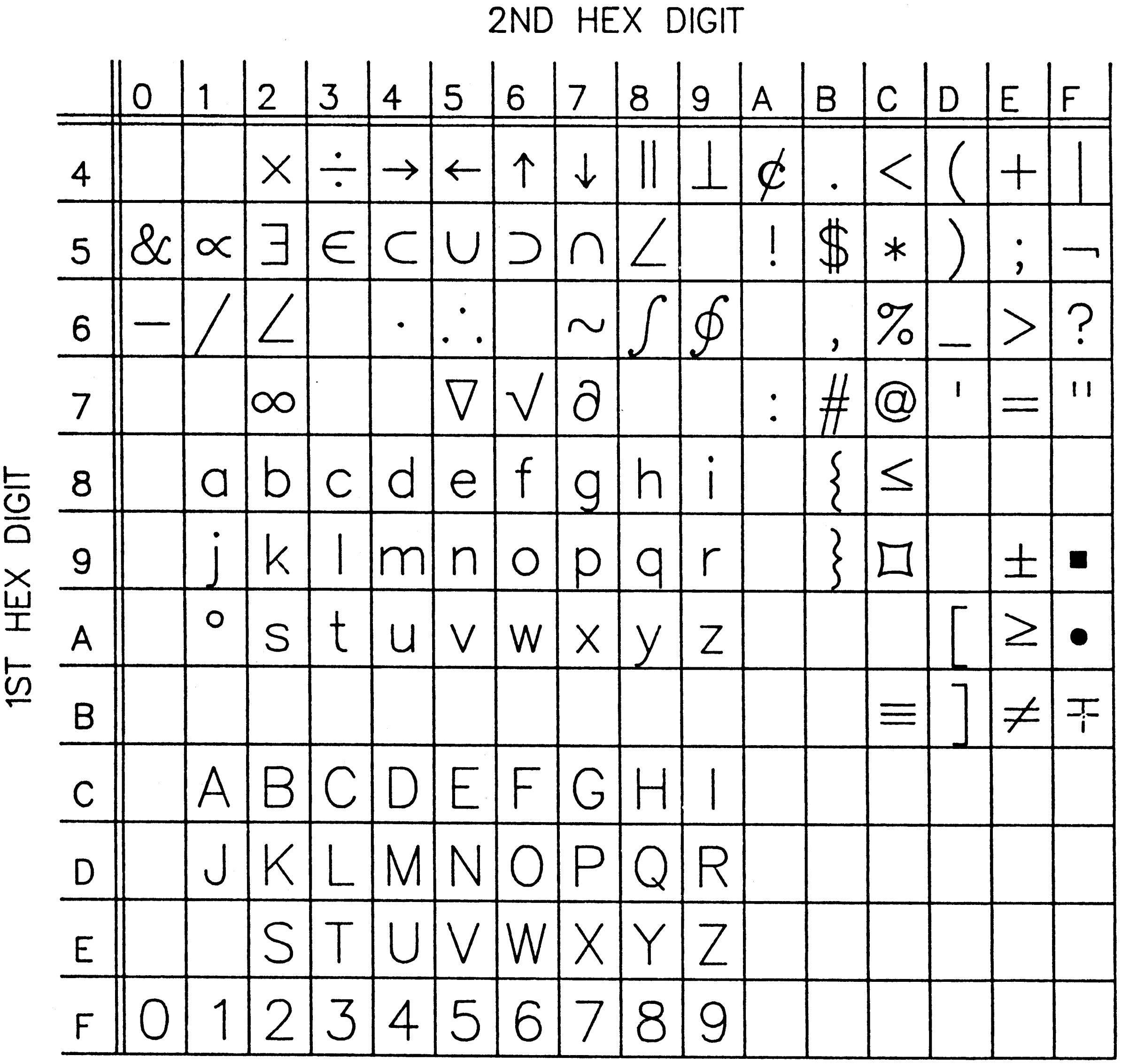
STANDARD

VSPAC = 4
PSUP = (0,10)
SCAL = 0.86
GRID = 16

HSPAC = -3
PSUB = (0,0)
SSCAL = 1.25
REF = (3,3)

LSPAC = 1
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

Table 1: Standard Character Set



2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4			×	÷	→	←	↑	↓	‖	⊥	¢	.	<	(	+	|
5	&	∝	∃	∈	⊂	∪	⊃	∩	∠		!	$	*	)	;	¬
6	—	/	∠		·	∴		~	∫	∮		,	%	_	>	?
7			∞			∇	√	∂			:	#	@	'	=	"
8		a	b	c	d	e	f	g	h	i		{	≤			
9		j	k	l	m	n	o	p	q	r		}	□		±	■
A		°	s	t	u	v	w	x	y	z				[	≥	●
B													≡	]	≠	∓
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

SANSERIF.1

VSPAC = 5	HSPAC = –5	LSPAC = 3
PSUP = (0,23)	PSUB = (C,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.52	ANGL = 0.00
GRID = 33	REF = (4,7)	LETTERSPACING ON

| Table 2: Sanserif.1 Character Set

2ND HEX DIGIT

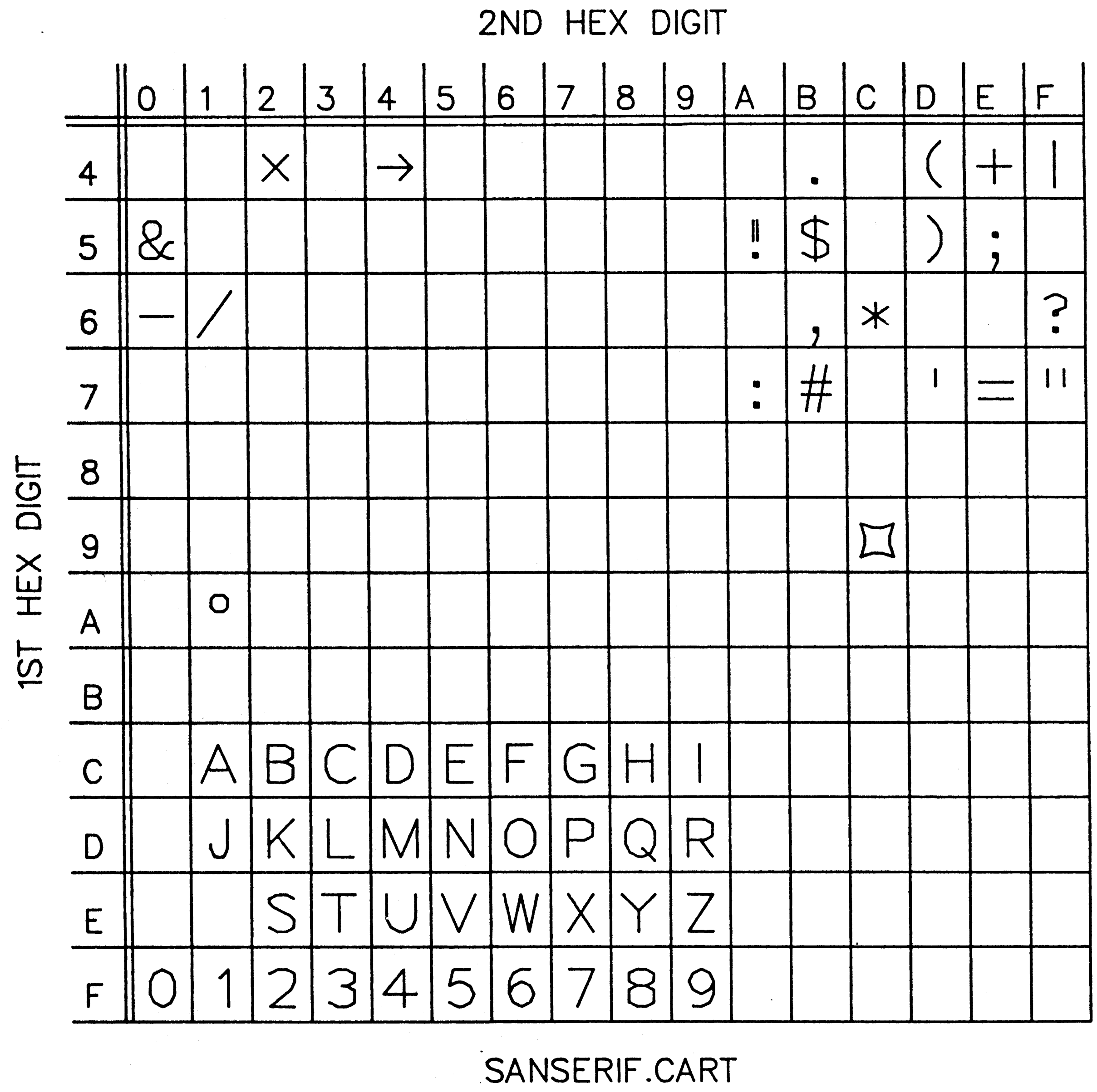
1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.		(	+	
5	&										!	$	*	)	;	¬
6	-	/										,		_		?
7											:			'	=	"
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A		°	s	t	u	v	w	x	y	z						
B						’	‘									
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

SANSERIF.2

VSPAC = 5	HSPAC = -5	LSPAC = 3
PSUP = (0,23)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.52	ANGL = 0.00
GRID = 33	REF = (3,7)	LETTERSPACING ON

| Table 3: Sanserif.2 Character Set



2ND HEX DIGIT

1ST HEX DIGIT	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4			×		→							.		(	+	|
5	&										!	$		)	;	
6	−	/										,	*			?
7											:	#		'	=	"
8																
9													⌑			
A		°														
B																
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

SANSERIF.CART

VSPAC = 3
PSUP = (0,10)
SCAL = 1.00
GRID = 17

HSPAC = −5
PSUB = (0,0)
SSCAL = 1.78
REF = (1,3)

LSPAC = 2
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 4: Sanserif.Cart Character Set

2ND HEX DIGIT

1ST HEX DIGIT

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

HELVETICA.1

VSPAC = 0
PSUP = (3,10)
SCAL = 1.00
GRID = 16

HSPAC = 0
PSUB = (3,0)
SSCAL = 1.36
REF = (3,3)

LSPAC = 1
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 5: Helvetica.1 Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4											¢	.	<	(	+	|
5	&										!	$	*	)	;	¬
6	−	/			·			~	∫	∮		,	%	_	>	?
7		ℵ	∞			∇	√	∂			:	#	@	'	=	"
8		a	b	c	d	e	f	g	h	i	§	{	≤			
9		j	k	l	m	n	o	p	q	r	‡	}	□		±	■
A		°	s	t	u	v	w	x	y	z	†			[	≥	●
B		^	´	`	˘	’	‘	‘	’				≡	]	≠	∓
C		A	B	C	D	E	F	G	H	I		ff	fi	fl	ffi	ffl
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

ROMAN.2

VSPAC = 2
PSUP = (0,23)
SCAL = 1.00
GRID = 41

HSPAC = −6
PSUB = (0,0)
SSCAL = 1.90
REF = (3,7)

LSPAC = 3
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 6: Roman.2 Character Set

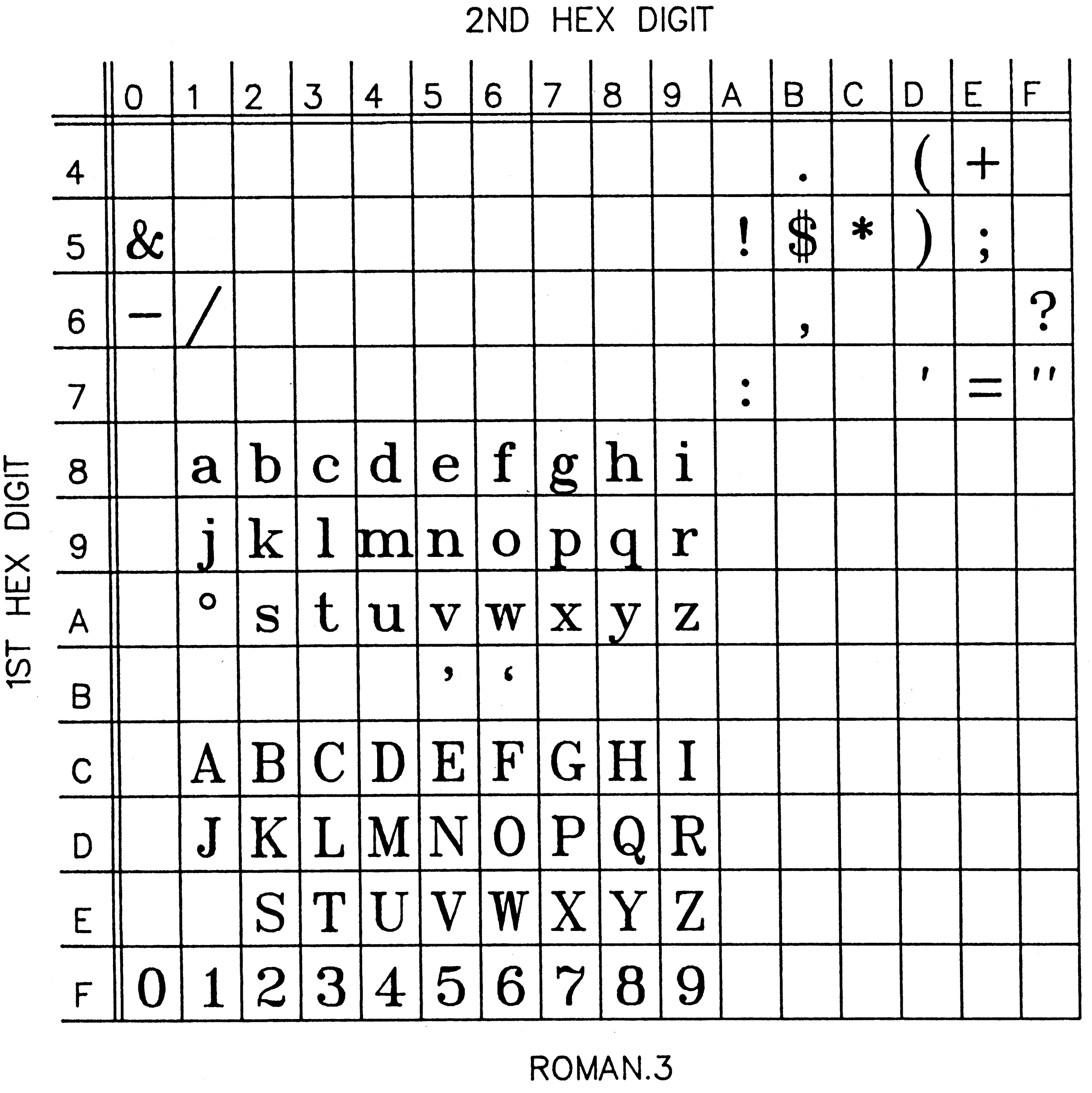
2ND HEX DIGIT

1ST HEX DIGIT	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.	<	(	+	|
5	&										!	$	*	)	;	
6	-	/										,	%		>	?
7											:		@	'	=	"
8		a	b	c	d	e	f	g	h	i		{				
9		j	k	l	m	n	o	p	q	r		}		±		
A		°	s	t	u	v	w	x	y	z				[		
B														]	≠	∓
C		A	B	C	D	E	F	G	H	I		ff	fi	fl	ffi	ffl
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

ROMAN.2A

VSPAC = 5	HSPAC = -5	LSPAC = 3
PSUP = (0,17)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.77	ANGL = 0.00
GRID = 24	REF = (3,5)	LETTERSPACING ON

| Table 7: Roman.2A Character Set



2ND HEX DIGIT

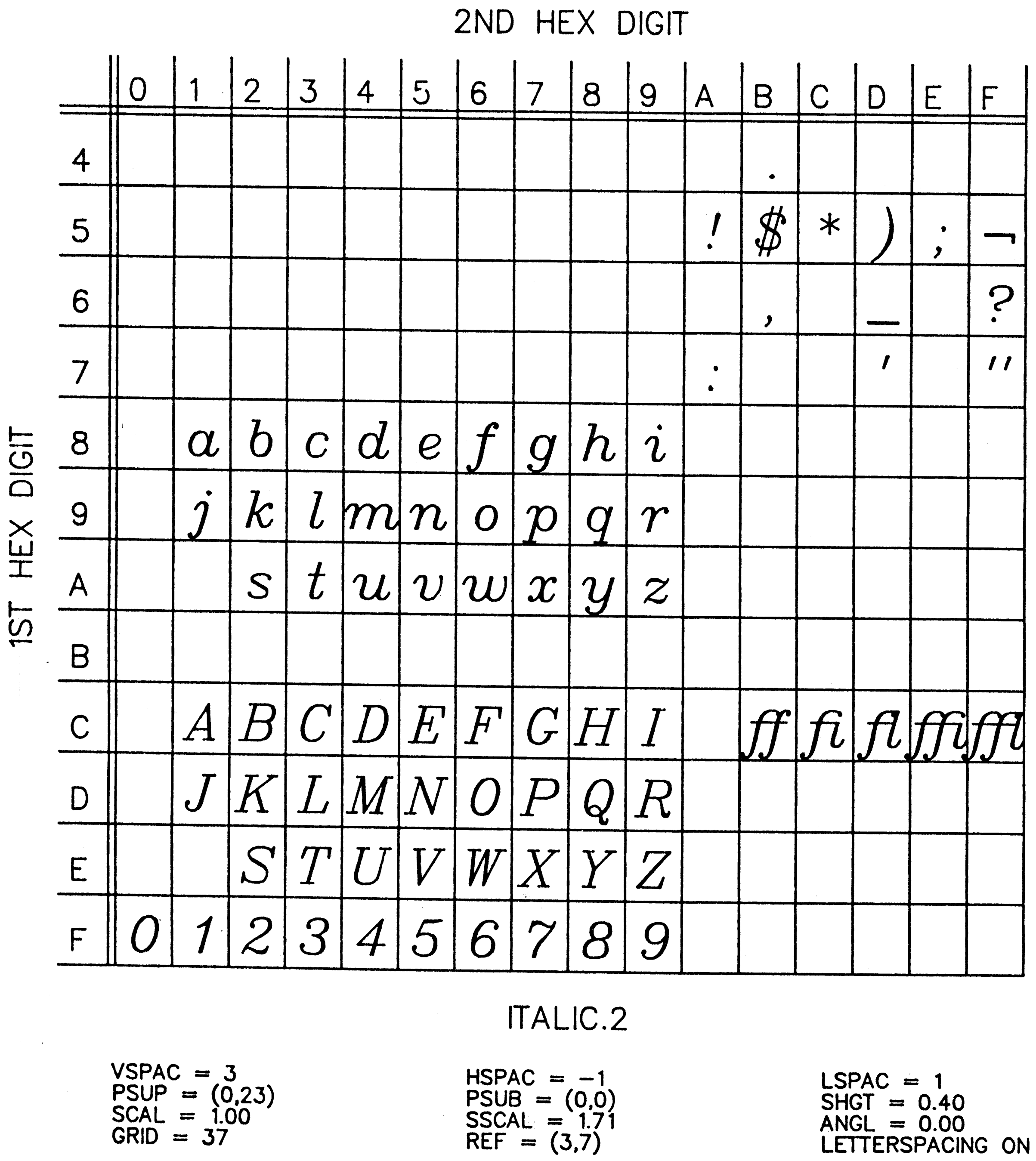
1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.		(	+	
5	&										!	$	*	)	;	
6	–	/										,				?
7											:			'	=	"
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A		°	s	t	u	v	w	x	y	z						
B						’	‘									
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

ROMAN.3

VSPAC = 5	HSPAC = –4	LSPAC = 3
PSUP = (0,23)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.52	ANGL = 0.00
GRID = 33	REF = (2,7)	LETTERSPACING ON

| Table 8: Roman.3 Character Set


2ND HEX DIGIT
1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5											!	$	*	)	;	¬
6												,		_		?
7											:			'		"
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A			s	t	u	v	w	x	y	z						
B																
C		A	B	C	D	E	F	G	H	I		ff	fi	fl	ffi	ffl
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

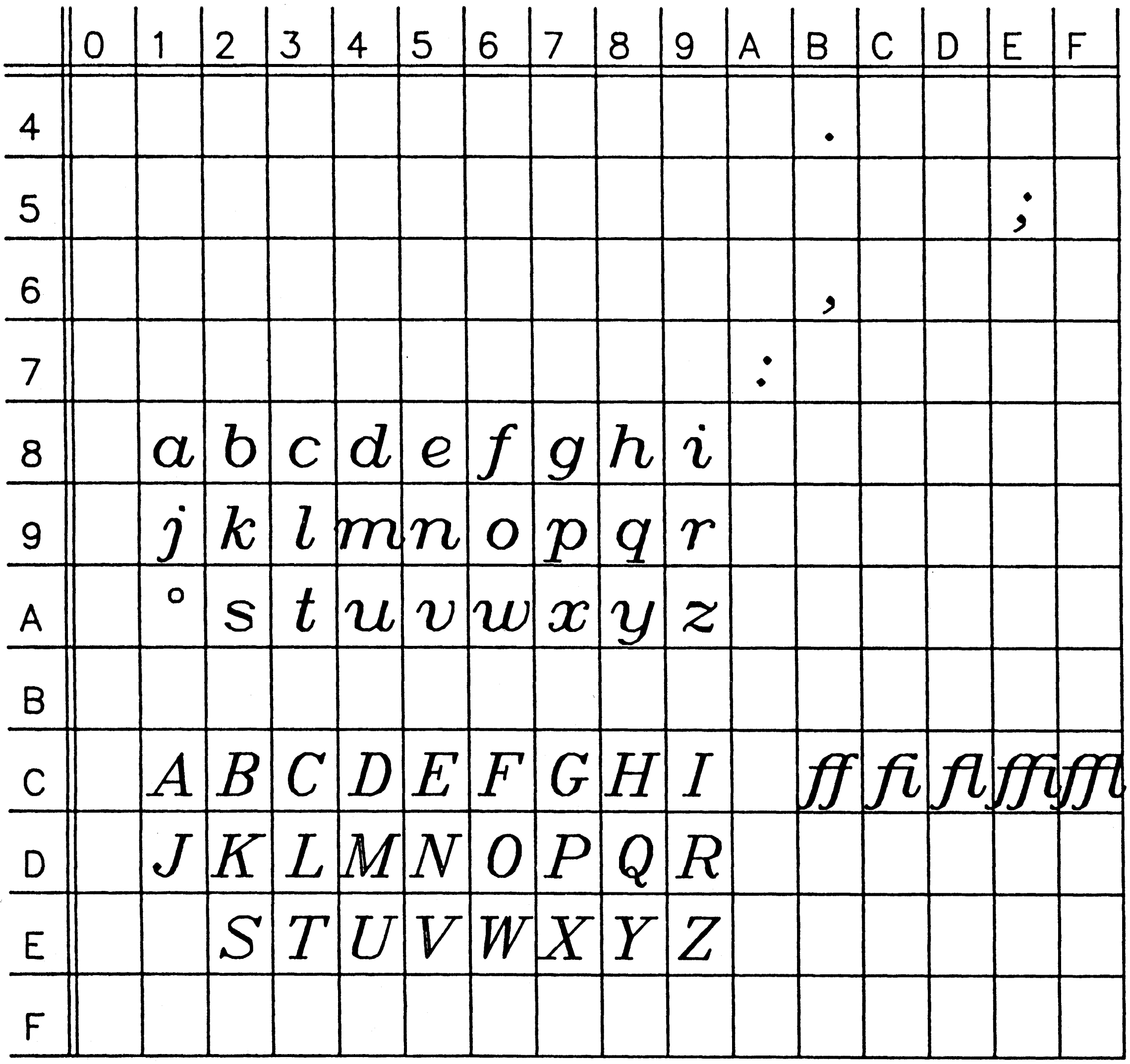
ITALIC.2

VSPAC = 3	HSPAC = −1	LSPAC = 1
PSUP = (0,23)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.71	ANGL = 0.00
GRID = 37	REF = (3,7)	LETTERSPACING ON

| Table 9: Italic.2 Character Set

2ND HEX DIGIT

1ST HEX DIGIT



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5															;	
6												,				
7											:					
8		*a*	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*						
9		*j*	*k*	*l*	*m*	*n*	*o*	*p*	*q*	*r*						
A		°	*s*	*t*	*u*	*v*	*w*	*x*	*y*	*z*						
B																
C		*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*I*		*ff*	*fi*	*fl*	*ffi*	*ffl*
D		*J*	*K*	*L*	*M*	*N*	*O*	*P*	*Q*	*R*						
E			*S*	*T*	*U*	*V*	*W*	*X*	*Y*	*Z*						
F																

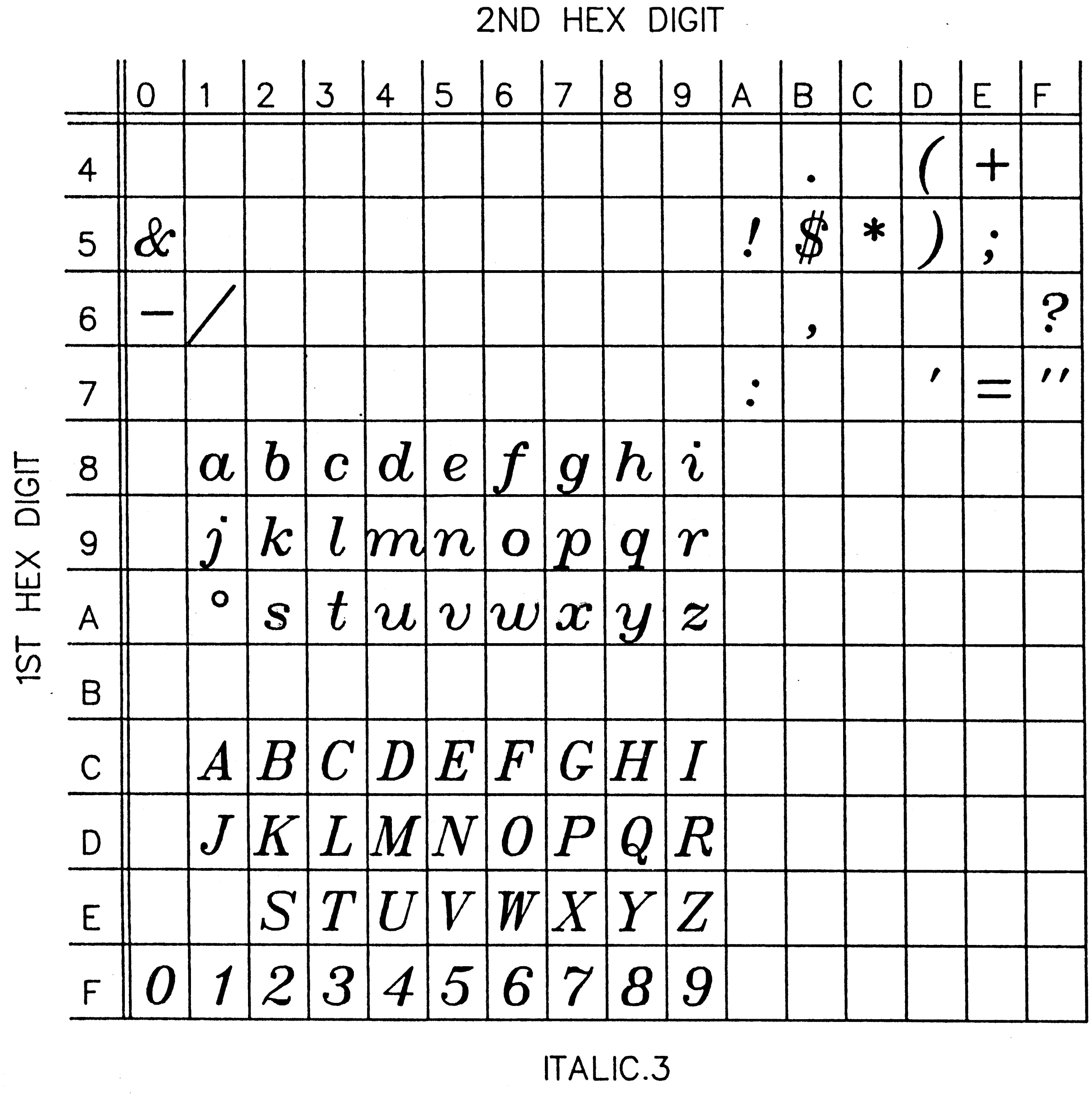
ITALIC.2A

VSPAC = 2
PSUP = (0,17)
SCAL = 1.00
GRID = 26

HSPAC = 0
PSUB = (0,0)
SSCAL = 1.92
REF = (3,4)

LSPAC = −1
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 10: Italic.2A Character Set



2ND HEX DIGIT

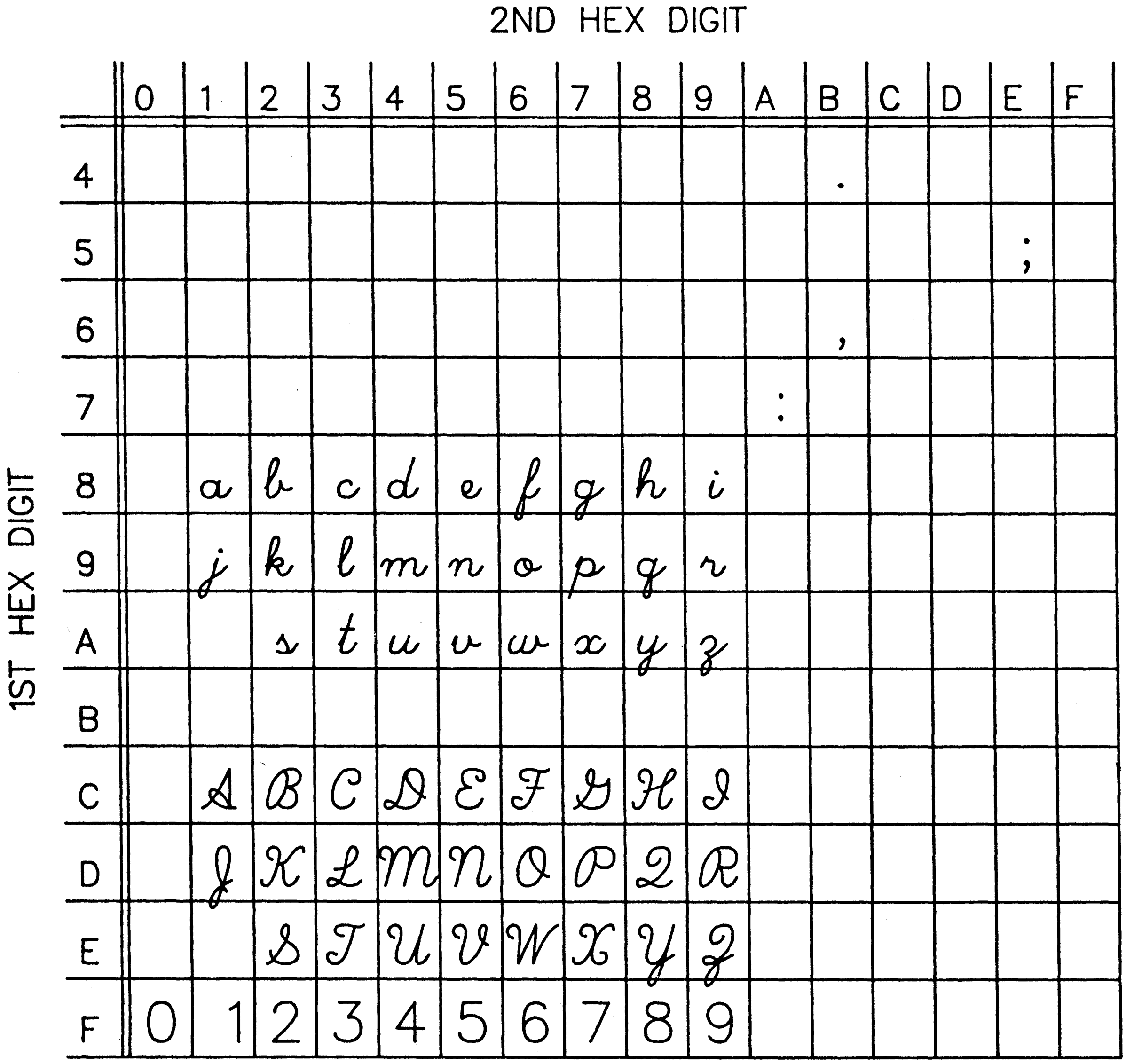
1ST HEX DIGIT

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

ITALIC.3

VSPAC = 2	HSPAC = −2	LSPAC = 1
PSUP = (0,23)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.71	ANGL = 0.00
GRID = 37	REF = (4,7)	LETTERSPACING ON

| Table 11: Italic.3 Character Set



2ND HEX DIGIT

1ST HEX DIGIT	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5															;	
6												,				
7											:					
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A			s	t	u	v	w	x	y	z						
B																
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

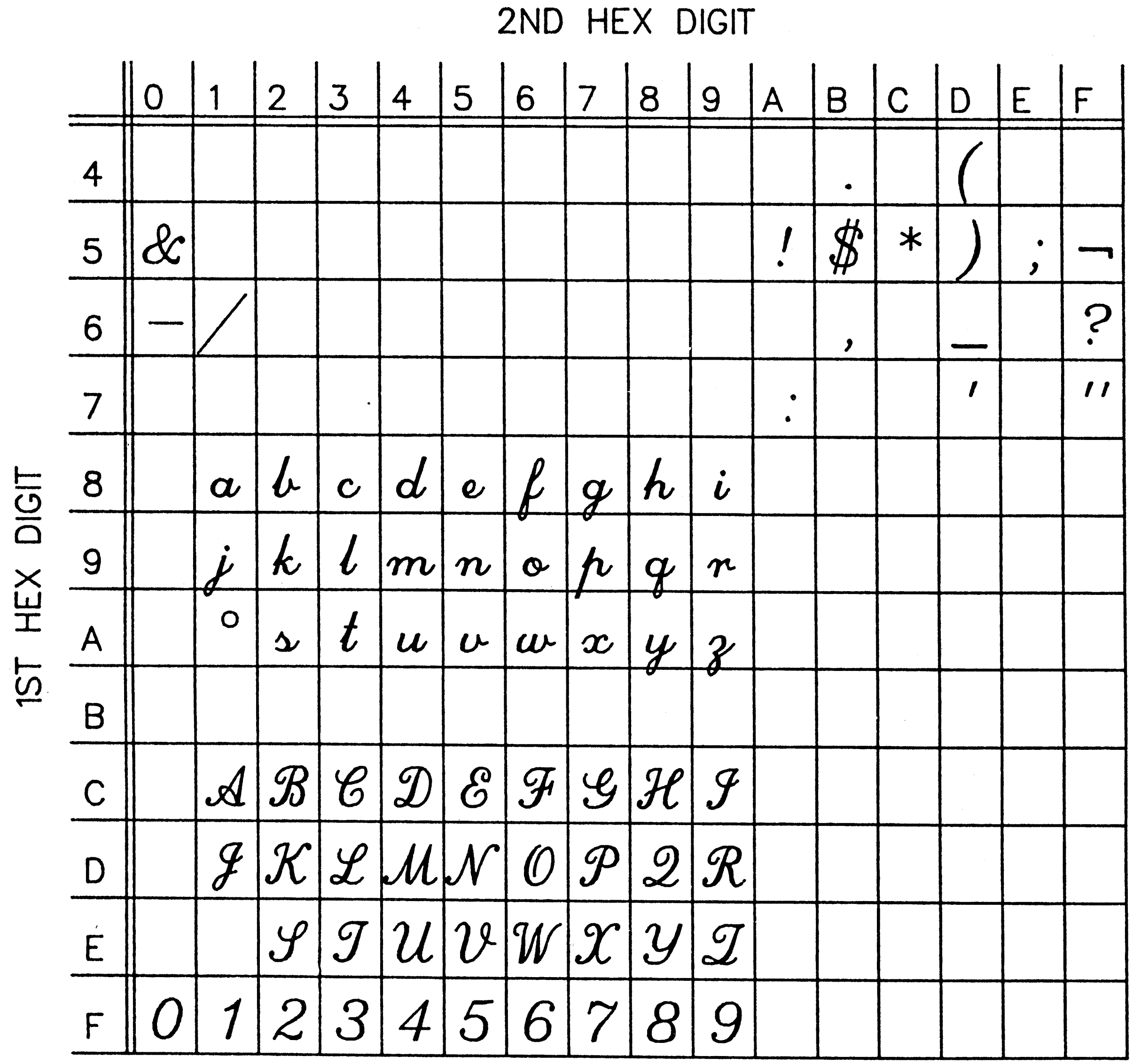
SCRIPT.1

VSPAC = 2
PSUP = (0,23)
SCAL = 1.00
GRID = 43

HSPAC = −5
PSUB = (0,4)
SSCAL = 2.00
REF = (11,12)

LSPAC = −8
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 12: Script.1 Character Set



2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.		(		
5	&										!	$	*	)	;	¬
6	—	/										,		_		?
7											:			'		"
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A		°	s	t	u	v	w	x	y	z						
B																
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

SCRIPT.2

VSPAC = 3
PSUP = (0,27)
SCAL = 1.00
GRID = 39

HSPAC = −4
PSUB = (0,0)
SSCAL = 1.81
REF = (8,12)

LSPAC = −7
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 13: Script.2 Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5	&										!				;	
6												,				?
7											:			'		
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A			s	t	u	v	w	x	y	z						
B																
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

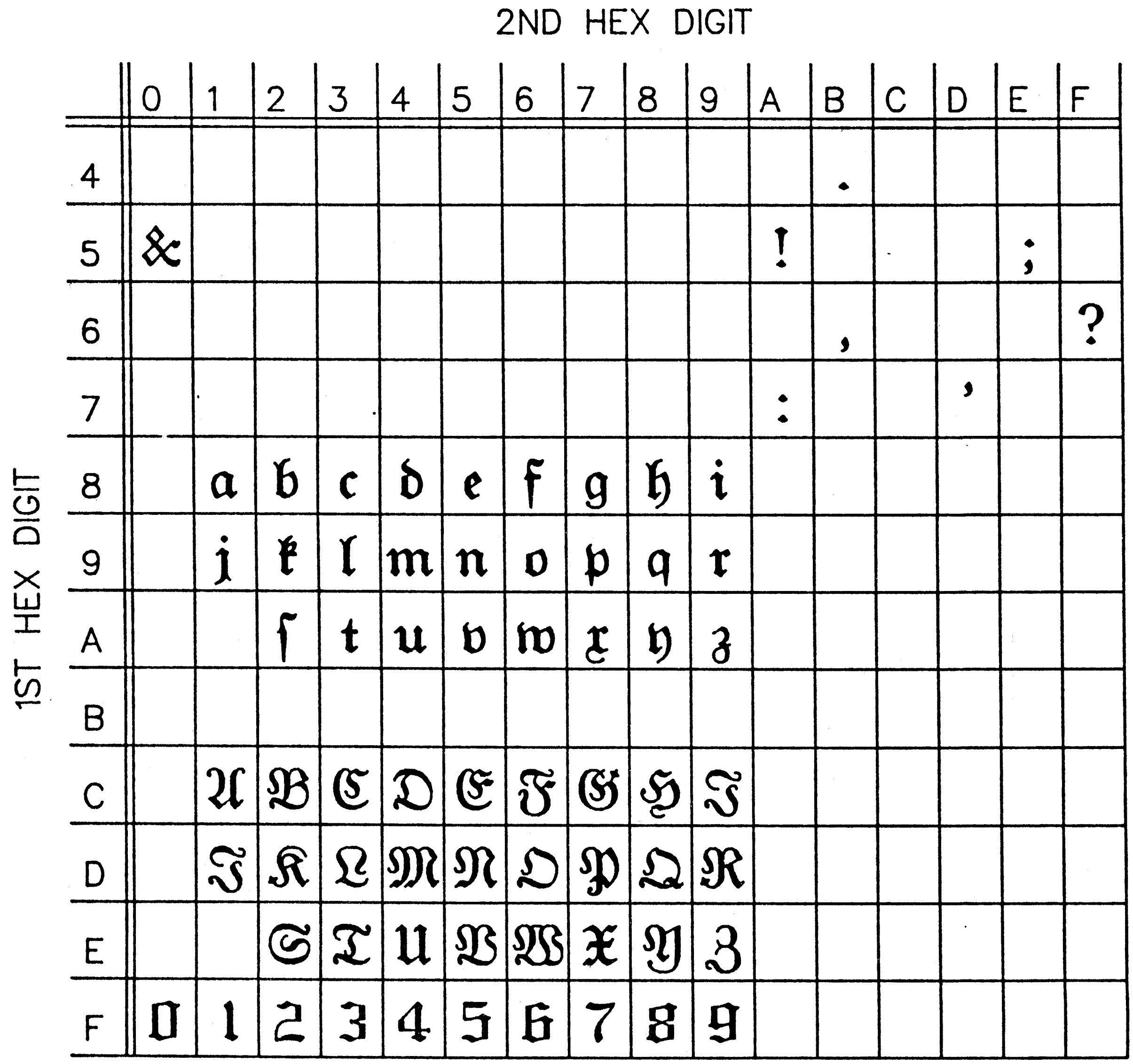
GOTHIC.ENGLISH

VSPAC = 6
PSUP = (0,23)
SCAL = 1.00
GRID = 29

HSPAC = −1
PSUB = (0,0)
SSCAL = 1.33
REF = (3,7)

LSPAC = 3
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 14: Gothic.English Character Set



2ND HEX DIGIT

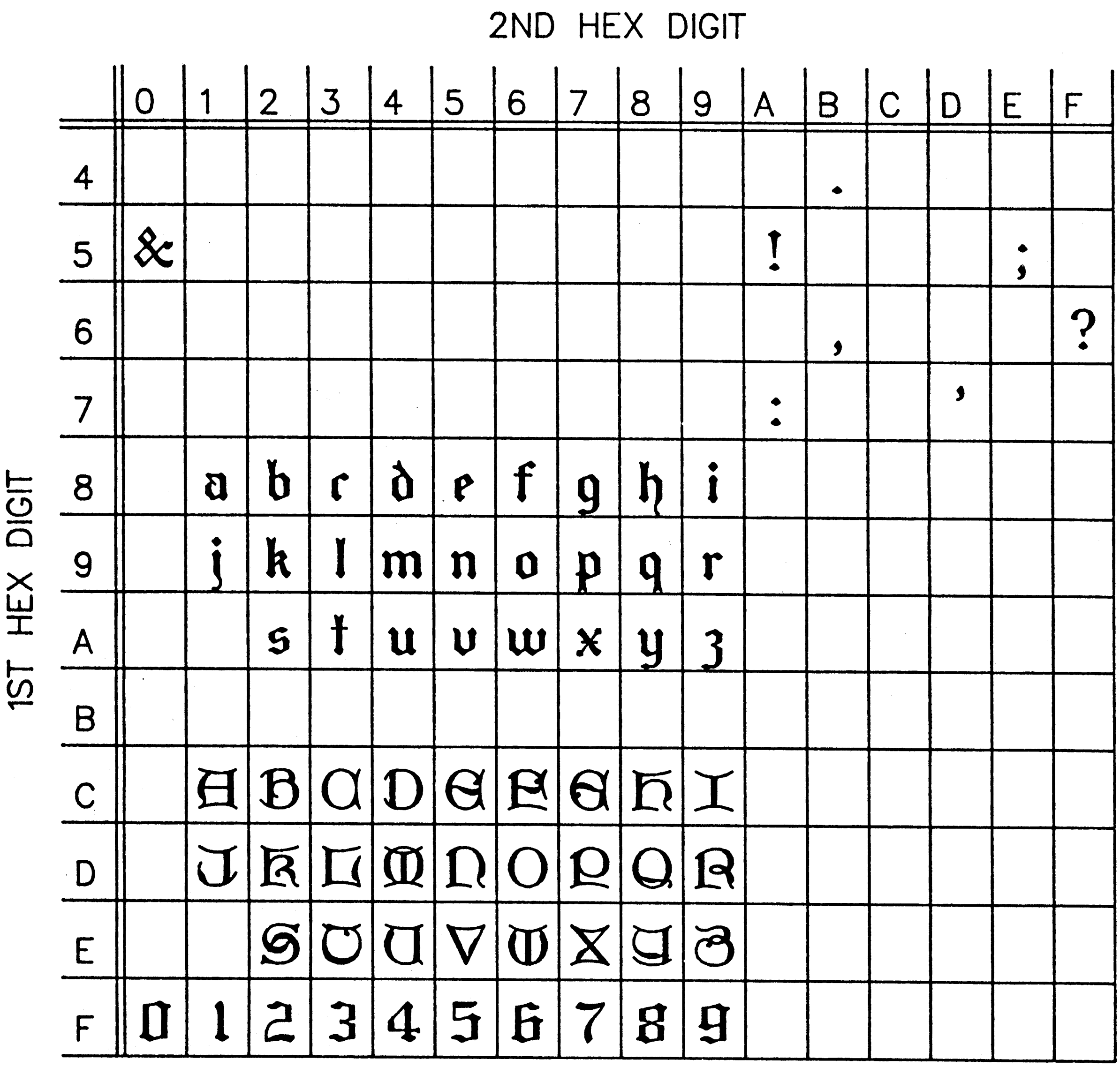
1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5	&										!				;	
6												,				?
7											:			'		
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A			ſ	t	u	v	w	x	y	z						
B																
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

GOTHIC.FRAKTUR

VSPAC = 5	HSPAC = −5	LSFAC = 3
PSUP = (0,21)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.43	ANGL = 0.00
GRID = 31	REF = (3,5)	LETTERSPACING ON

| Table 15: Gothic.Fraktur Character Set



2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5	&										!				;	
6												,				?
7											:			'		
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A			s	t	u	v	w	x	y	z						
B																
C		A	B	C	D	E	F	G	H	I						
D		J	K	L	M	N	O	P	Q	R						
E			S	T	U	V	W	X	Y	Z						
F	0	1	2	3	4	5	6	7	8	9						

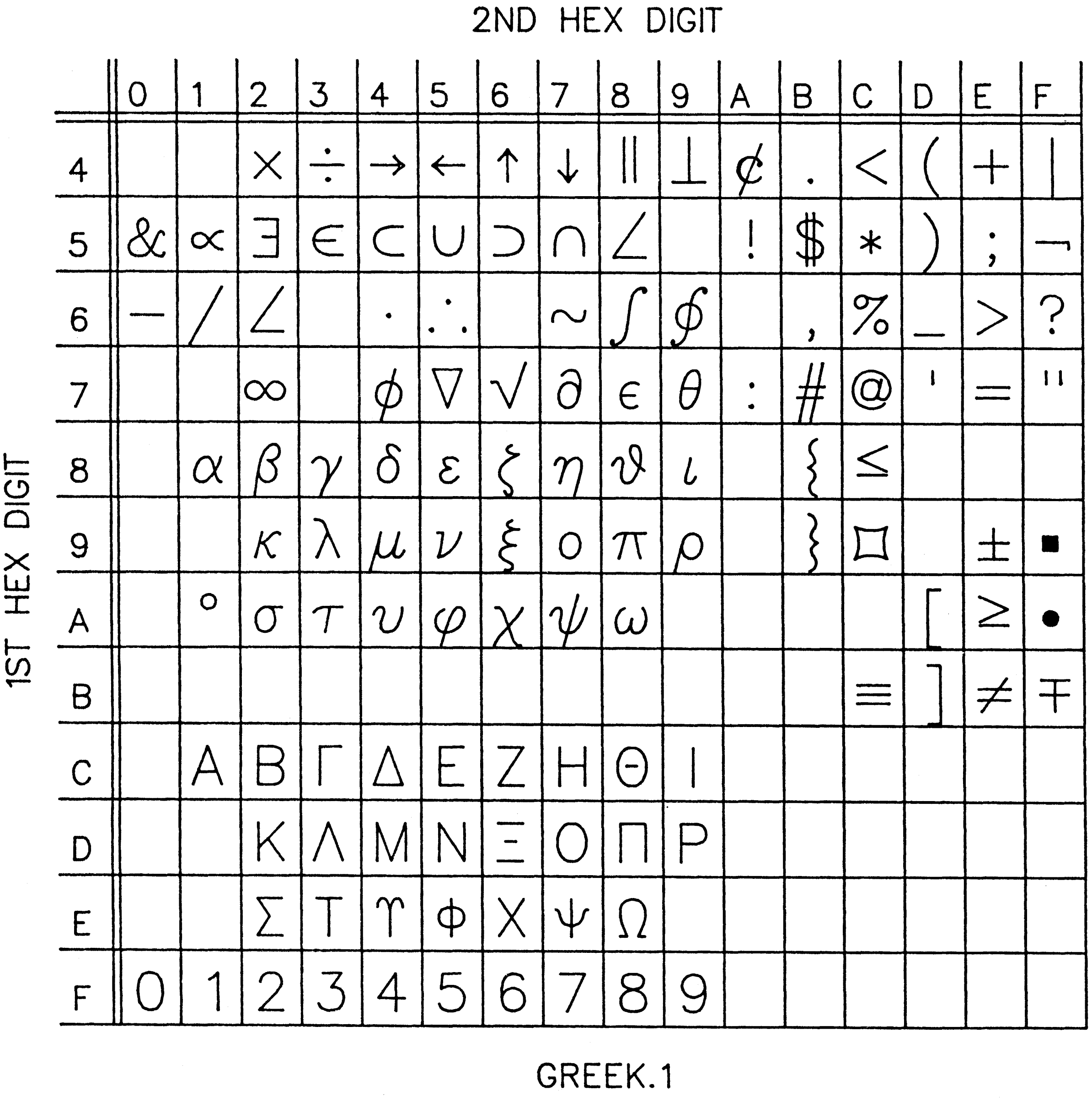
GOTHIC.ITALIAN

VSPAC = 6
PSUP = (0,23)
SCAL = 1.00
GRID = 29

HSPAC = −4
PSUB = (0,0)
SSCAL = 1.33
REF = (2,7)

LSPAC = 3
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 16: Gothic.Italian Character Set



2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4			×	÷	→	←	↑	↓	‖	⊥	¢	.	<	(	+	|
5	&	∝	∃	∈	⊂	∪	⊃	∩	∠		!	$	*	)	;	¬
6	—	/	∠		·	∴		~	∫	∮		,	%	_	>	?
7			∞		ϕ	∇	√	∂	ϵ	θ	:	#	@	'	=	"
8		α	β	γ	δ	ε	ζ	η	ϑ	ι		{	≤			
9			κ	λ	μ	ν	ξ	ο	π	ρ		}	◻		±	■
A		°	σ	τ	υ	φ	χ	ψ	ω					[	≥	●
B													≡	]	≠	∓
C		A	B	Γ	Δ	E	Z	H	Θ	I						
D			K	Λ	M	N	Ξ	O	Π	P						
E			Σ	T	ϒ	Φ	X	Ψ	Ω							
F	0	1	2	3	4	5	6	7	8	9						

GREEK.1

VSPAC = 5
PSUP = (0,23)
SCAL = 1.00
GRID = 34

HSPAC = −5
PSUB = (0,0)
SSCAL = 1.57
REF = (4,7)

LSPAC = 3
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 17: Greek.1 Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4											¢	.	<	(	+	|
5	&										!	$	*	)	;	¬
6	−	/			·			~	∫	⨏		,	%	_	>	?
7		ℵ	∞		ϕ	∇	√	∂	ϵ	θ	:	#	@	′	=	″
8		α	β	γ	δ	ε	ζ	η	ϑ	ι	§	{	≤			
9			κ	λ	μ	ν	ξ	ο	π	ρ	‡	}	□		±	■
A			σ	τ	υ	φ	χ	ψ	ω		†			[	≥	●
B		^	´	`	‿	᾿	῾	῾	᾿				≡	]	≠	∓
C		Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι						
D			Κ	Λ	Μ	Ν	Ξ	Ο	Π	Ρ						
E			Σ	Τ	Υ	Φ	Χ	Ψ	Ω							
F	0	1	2	3	4	5	6	7	8	9						

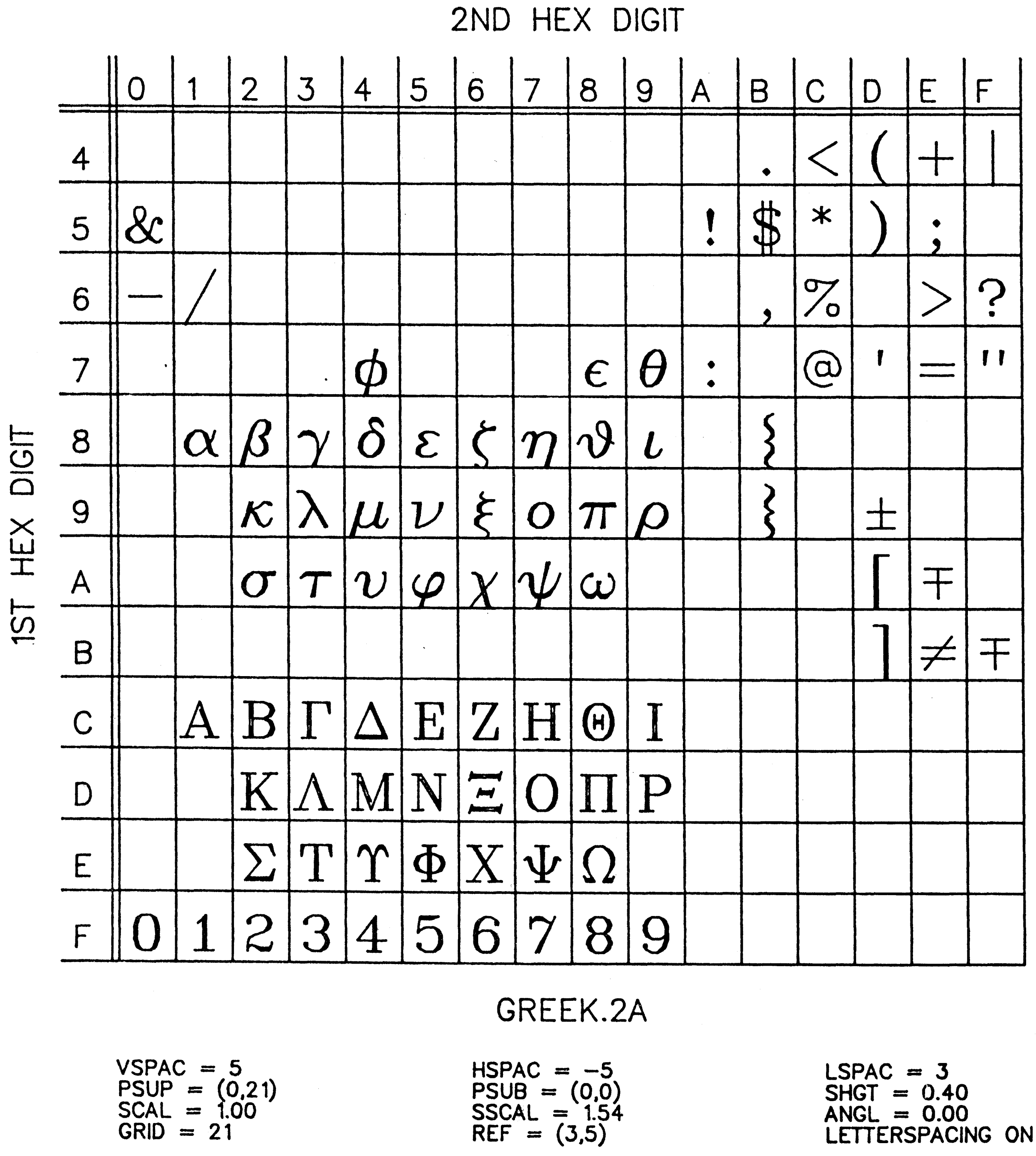
GREEK.2

VSPAC = 5
PSUP = (0,23)
SCAL = 1.00
GRID = 41

HSPAC = −5
PSUB = (0,0)
SSCAL = 1.90
REF = (3,7)

LSPAC = 3
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 18: Greek.2 Character Set



2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.	<	(	+	|
5	&										!	$	*	)	;	
6	—	/										,	%		>	?
7					φ				ϵ	θ	:		@	'	=	"
8		α	β	γ	δ	ε	ζ	η	ϑ	ι		{				
9			κ	λ	μ	ν	ξ	ο	π	ρ		}		±		
A			σ	τ	υ	φ	χ	ψ	ω					[	∓	
B														]	≠	∓
C		Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι						
D			Κ	Λ	Μ	Ν	Ξ	Ο	Π	Ρ						
E			Σ	Τ	Υ	Φ	Χ	Ψ	Ω							
F	0	1	2	3	4	5	6	7	8	9						

GREEK.2A

VSPAC = 5
PSUP = (0,21)
SCAL = 1.00
GRID = 21

HSPAC = −5
PSUB = (0,0)
SSCAL = 1.54
REF = (3,5)

LSPAC = 3
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 19: Greek.2A Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4			×		→							.		(	+	|
5	&										!	$		)	;	
6	—	/			·							,	*			?
7											:	#		'	=	"
8																
9													⌑			
A		°														
B																
C		Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι						
D			Κ	Λ	Μ	Ν	Ξ	Ο	Π	Ρ						
E			Σ	Τ	ϒ	Φ	Χ	Ψ	Ω							
F	0	1	2	3	4	5	6	7	8	9						

GREEK.CART

VSPAC = 3	HSPAC = −6	LSPAC = 2
PSUP = (0,10)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.78	ANGL = 0.00
GRID = 17	REF = (1,3)	LETTERSPACING ON

| Table 20: Greek.Cart Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5											!				;	
6												,		_		?
7											:					
8	а	б	в	г	д	е	ж	з	и	й	к					
9	л	м	н	о	п	р	с	т	у	ф	х					
A	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	ё					
B						»	«									
C	А	Б	В	Г	Д	Е	Ж	З	И	Й	К					
D	Л	М	Н	О	П	Р	С	Т	У	Ф	Х					
E	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	Ё					
F	0	1	2	3	4	5	6	7	8	9						

CYRILLIC.2

VSPAC = 5
PSUP = (0,23)
SCAL = 1.00
GRID = 36

HSPAC = −5
PSUB = (0,0)
SSCAL = 1.67
REF = (3,7)

LSPAC = 3
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 21: Cyrillic.2 Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5																
6	ー											,				
7																
8																
9																
A		あ	か	さ	た	な	は	ま	や	ら	わ	が	ざ	だ	ば	ぱ
B		い	き	し	ち	に	ひ	み	い	り	ゐ	ぎ	じ	ぢ	び	ぴ
C		う	く	す	つ	ぬ	ふ	む	ゆ	る	う	ぐ	ず	づ	ぶ	ぷ
D		え	け	せ	て	ね	へ	め	え	れ	ゑ	げ	ぜ	で	べ	ぺ
E		お	こ	そ	と	の	ほ	も	よ	ろ	を	ご	ぞ	ど	ぼ	ぽ
F											ん					

HIRAGANA

VSPAC = 8
PSUP = (0,20)
SCAL = 1.00
GRID = 29

HSPAC = −1
PSUB = (0,0)
SSCAL = 1.32
REF = (0,4)

LSPAC = 0
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 22: Hiragana Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5																
6	ー											,				
7																
8																
9																
A		ア	カ	サ	タ	ナ	ハ	マ	ヤ	ラ	ワ	ガ	ザ	ダ	バ	パ
B		イ	キ	シ	チ	ニ	ヒ	ミ	イ	リ	ヰ	ギ	ジ	ヂ	ビ	ピ
C		ウ	ク	ス	ツ	ヌ	フ	ム	ユ	ル	ウ	グ	ズ	ヅ	ブ	プ
D		エ	ケ	セ	テ	ネ	ヘ	メ	エ	レ	ヱ	ゲ	ゼ	デ	ベ	ペ
E		オ	コ	ソ	ト	ノ	ホ	モ	ヨ	ロ	ヲ	ゴ	ゾ	ド	ボ	ポ
F											ン					

KATAKANA

VSPAC = 8
PSUP = (0,20)
SCAL = 1.00
GRID = 29

HSPAC = −1
PSUB = (0,0)
SSCAL = 1.32
REF = (0,4)

LSPAC = 0
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

Table 23: Katakana Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5																
6	一											,				
7																
8																
9																
A		一	七	万	三	上	下	不	並	中	主	乗	乙	九	乱	事
B		二	五	亜	交	京	人	今	仕	他	代	伸	位	低	何	余
C		作	来	例	供	係	値	偏	側	傾	像	価	億	元	先	光
D		入	内	全	両	八	公	六	共	其	具	典	再	冬	氷	冷
E		出	函	刀	分	切	列	利	前	割	力	加	動	化	北	十
F		千	午	半	南	卵	巻	原	去	又	及	反	取	口	古	可

KANJI1

VSPAC = 8	HSPAC = −1	LSPAC = 0
PSUP = (0,20)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.32	ANGL = 0.00
GRID = 29	REF = (0,4)	LETTERSPACING ON

| Table 24: Kanji1 Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5																
6	一											,				
7																
8		右	合	各	同	名	向	吸	周	味	和	品	哲	商	問	単
9		嗅	器	四	因	回	固	国	圏	円	図	土	在	地	均	垂
A		型	基	場	塔	境	増	墨	圧	士	夏	夕	外	多	夜	大
B		太	天	女	如	始	子	存	字	学	宇	宙	定	室	家	密
C		寒	実	射	対	導	小	少	尺	尾	屈	居	層	山	島	嵐
D		川	工	左	差	市	布	帆	常	幅	干	平	年	幹	幾	底
E		度	座	庭	式	弓	引	弗	弧	弦	強	弾	形	役	彼	後
F		径	徒	従	御	微										

KANJI2

VSPAC = 8
PSUP = (0,20)
SCAL = 1.00
GRID = 29

HSPAC = -1
PSUB = (0,0)
SSCAL = 1.32
REF = (0,4)

LSPAC = 0
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 25: Kanji2 Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5																
6	—											,				
7	心	性	思	恒	悪	惑	想	愛	感	態	慣	応	戈	成	或	戸
8	扇	手	折	押	持	指	振	挺	捩	接	摘	支	放	故	教	散
9	数	整	文	料	斜	新	方	於	日	早	明	易	昔	星	春	昨
A	時	昼	晴	晶	暑	暗	曇	曲	書	最	月	有	期	朝	木	未
B	本	束	東	松	林	析	果	枝	柱	柳	核	根	械	森	植	極
C	構	標	楽	横	橋	機	楕	桜	欠	次	止	正	此	歪	帰	死
D	母	毎	比	毛	氏	気	水	永	決	汽	河	油	法	波	海	流
E	液	減	涌	渦	温	測	源	準	溶	潮	滝	沢	湿	火	炭	無
F	然	熱	灯	燃	燐	焼	炉	為	父	片	牙	牛	物	特	犬	状

KANJI3

VSPAC = 8	HSPAC = −1	LSPAC = 0
PSUP = (0,20)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.32	ANGL = 0.00
GRID = 29	REF = (0,4)	LETTERSPACING ON

| Table 26: Kanji3 Character Set

2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5																
6	一											,				
7		率	王	玉	珪	球	現	理	瓜	瓦	甘	生	用	田	甲	男
8		界	異	画	番	当	病	発	白	百	的	皮	皿	目	直	相
9		真	着	矛	矢	知	短	石	研	硝	硫	硼	磁	示	私	秋
A		科	秒	秤	程	種	称	稲	積	穴	究	空	窒	立	竹	第
B		等	筆	算	管	範	米	粒	粘	糸	系	純	紙	級	素	細
C		終	組	結	絶	絹	経	線	締	緯	縦	総	絵	続	置	羊
D		美	羽	翠	翡	翼	老	考	者	耳	聞	肉	胞	臣	自	至
E		舌	舟	般	船	良	色	花	若	茶	草	荷	菊	葉	蒸	薬
F		処	虚	号	虫	血	行	術	衝	衡	衣	表	装	複	西	要

KANJI4

VSPAC = 8
PSUP = (0,20)
SCAL = 1.00
GRID = 29

HSPAC = -1
PSUB = (0,0)
SSCAL = 1.32
REF = (0,4)

LSPAC = 0
SHGT = 0.40
ANGL = 0.00
LETTERSPACING ON

| Table 27: Kanji4 Character Set

2ND HEX DIGIT

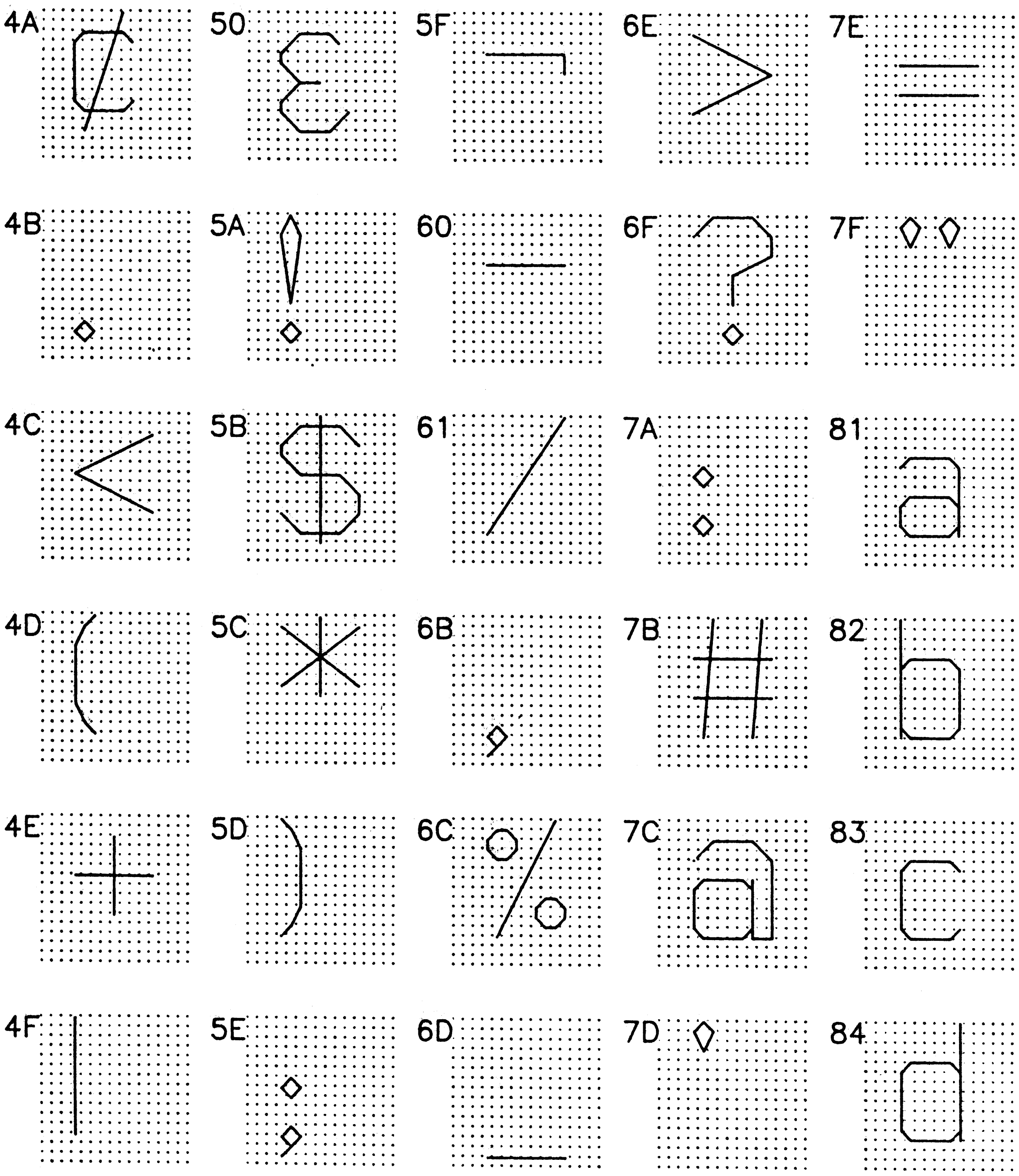
1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4												.				
5																
6	一											,				
7		見	角	解	言	計	記	話	語	誰	論	証	読	変	谷	豆
8		象	豆	負	質	赤	走	超	足	距	路	身	車	軌	軸	較
9		軽	輪	輻	転	辛	辞	込	近	逆	通	速	造	連	進	運
A		過	道	達	遠	遅	還	辺	酒	酸	醜	釈	里	重	量	金
B		銀	銅	鋼	鉄	鉱	長	門	閉	開	間	関	降	限	除	陰
C		陸	陽	雄	雌	双	離	難	雨	雪	雲	零	雷	電	震	霊
D		青	静	非	面	革	音	項	題	類	風	飛	食	首	香	馬
E		験	骨	体	高	鬼	魚	鳥	塩	鹿	麦	麻	黄	黒	点	鼓
F		鼻	歯	竜	亀											

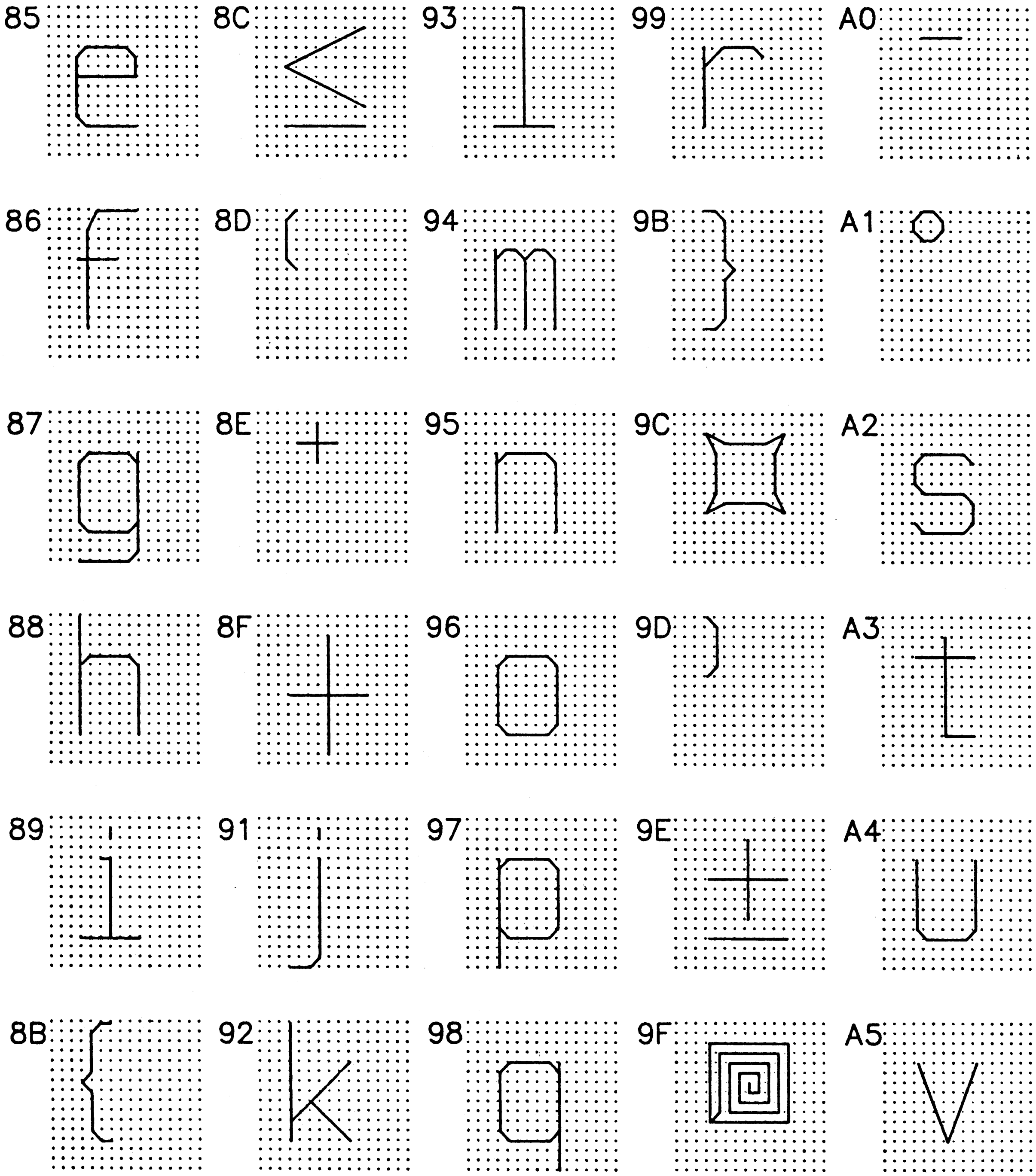
KANJI5

VSPAC = 8	HSPAC = −1	LSPAC = 0
PSUP = (0,20)	PSUB = (0,0)	SHGT = 0.40
SCAL = 1.00	SSCAL = 1.32	ANGL = 0.00
GRID = 29	REF = (0,4)	LETTERSPACING ON

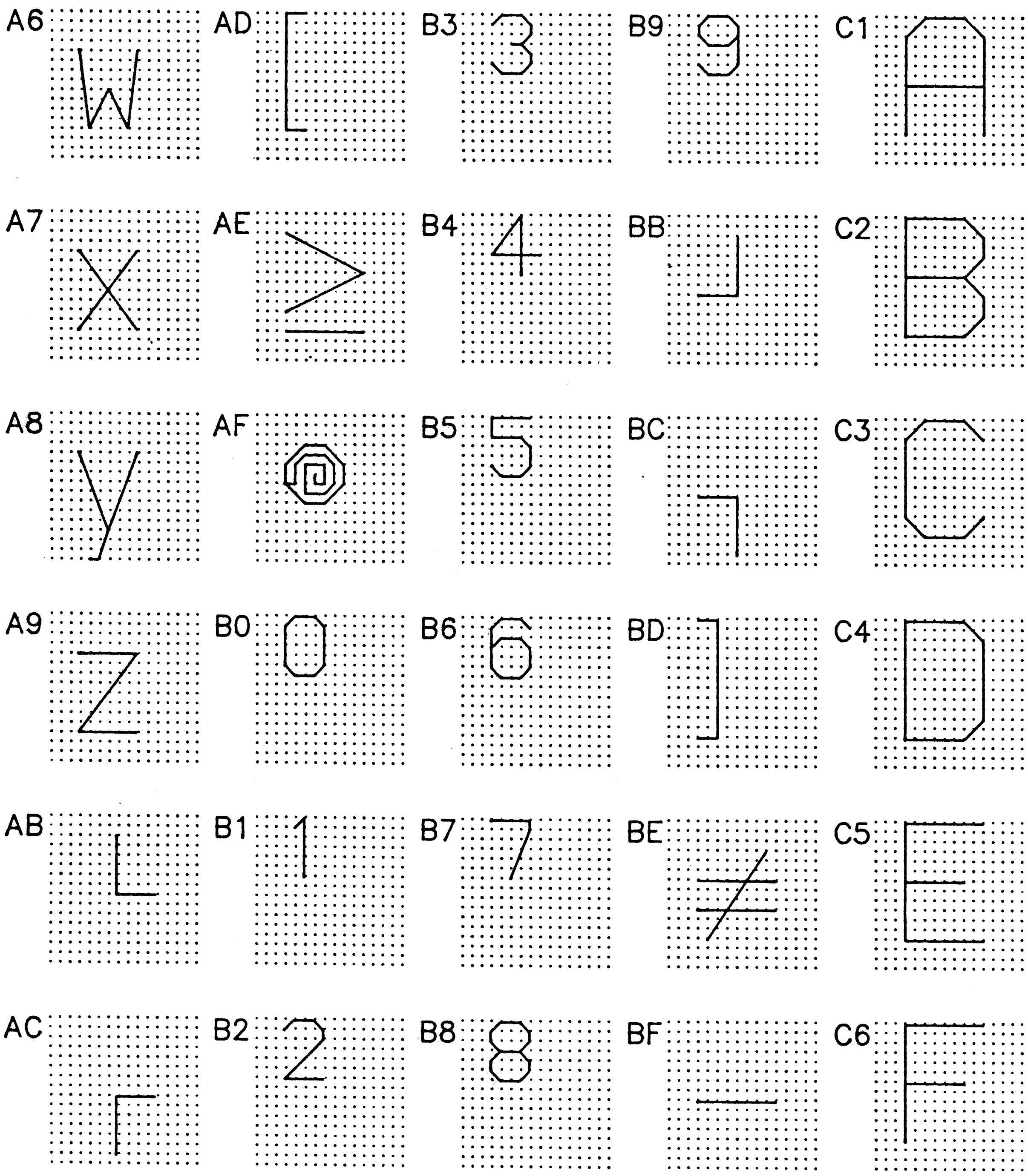
| Table 28: Kanji5 Character Set



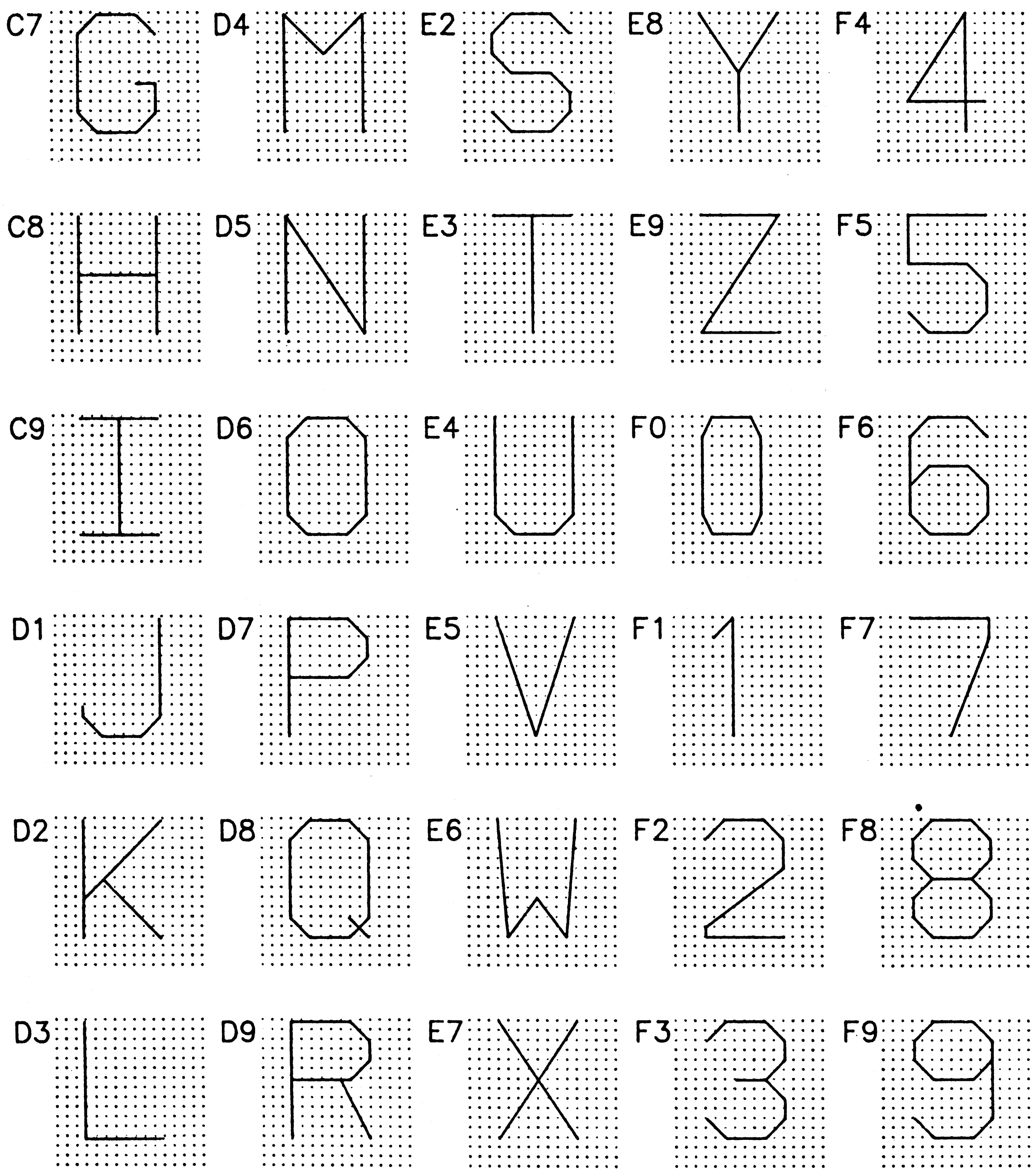
| Standard Character Set Grid



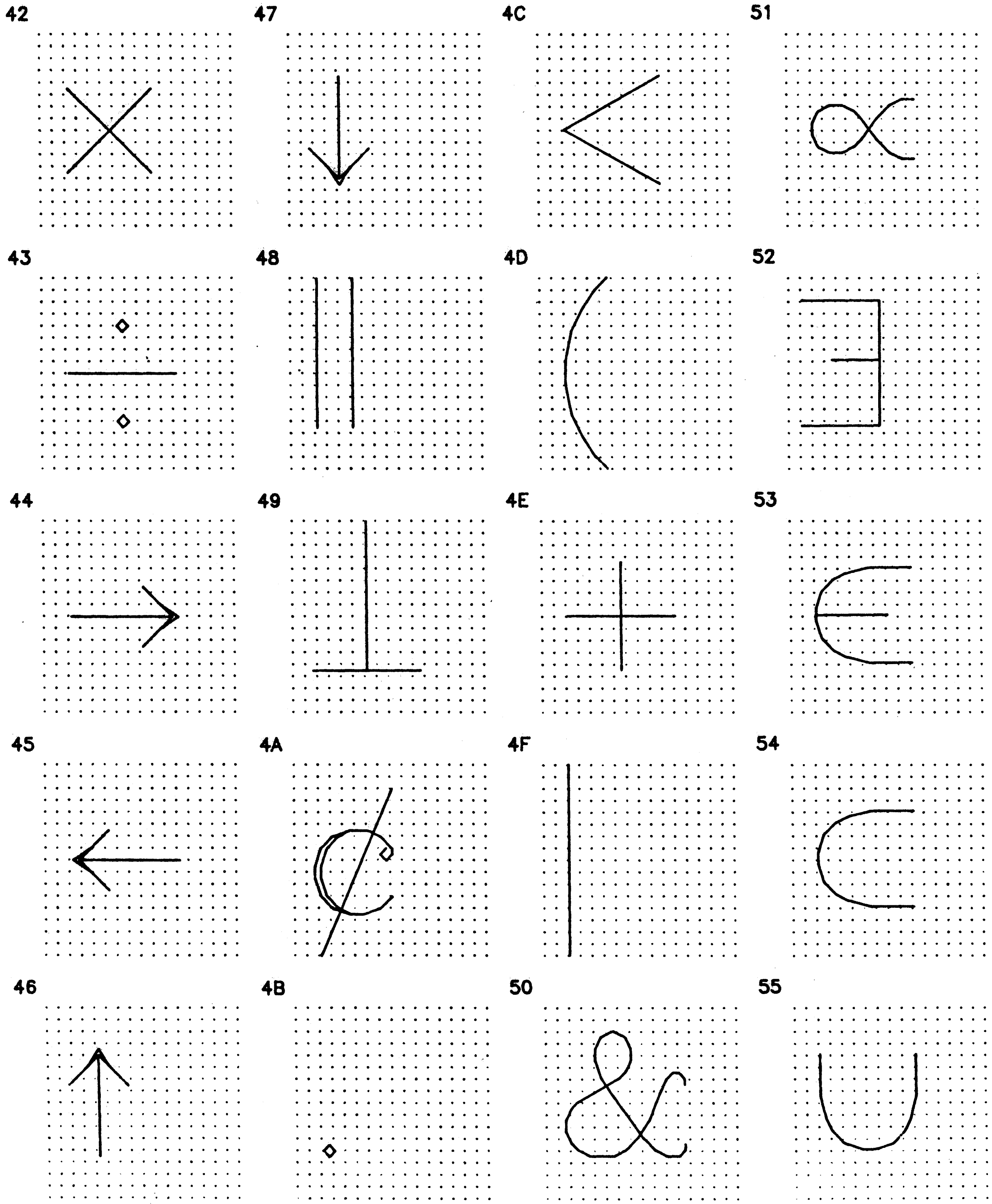
| Standard Character Set Grid



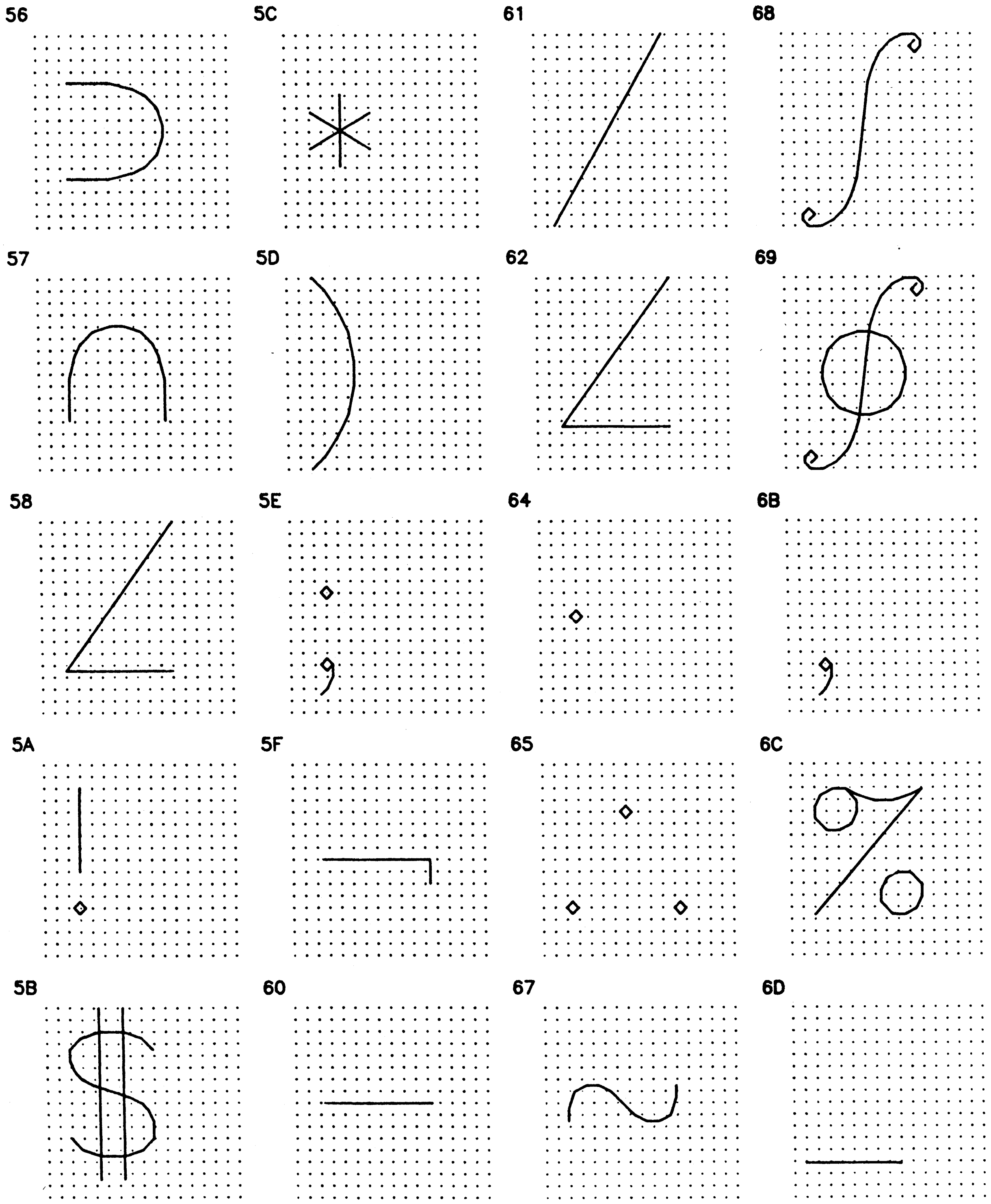
| Standard Character Set Grid



| Standard Character Set Grid

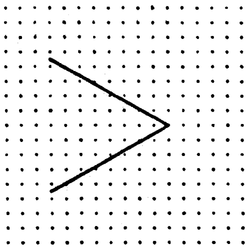


| Sanserif.1 Character Set Grid

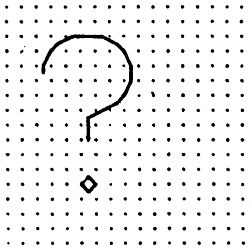


| Sanserif.1 Character Set Grid

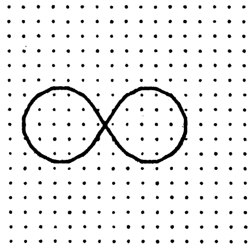
6E



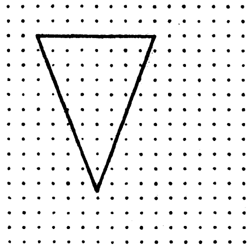
6F



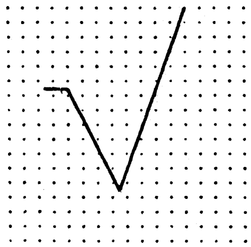
72



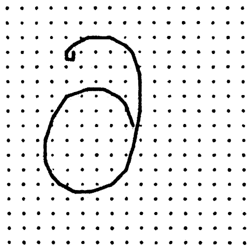
75



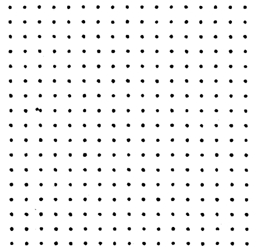
76



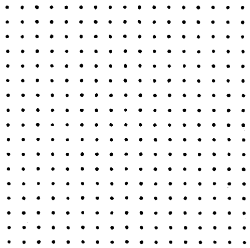
77



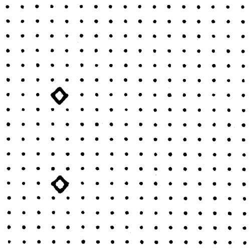
78



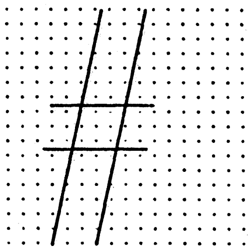
79



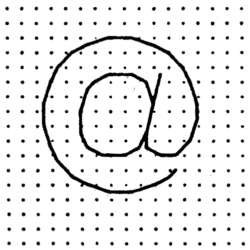
7A



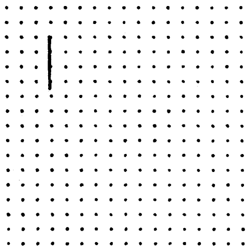
7B



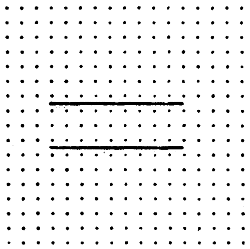
7C



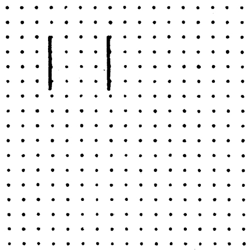
7D



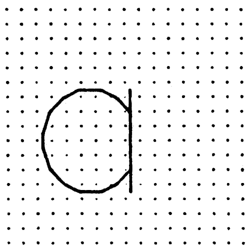
7E



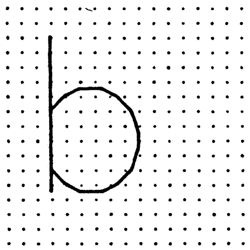
7F



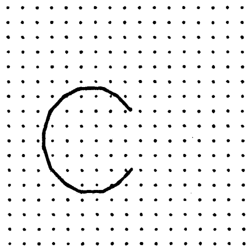
81



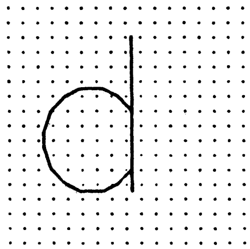
82



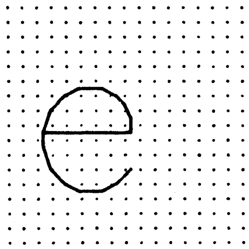
83



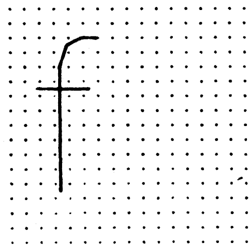
84



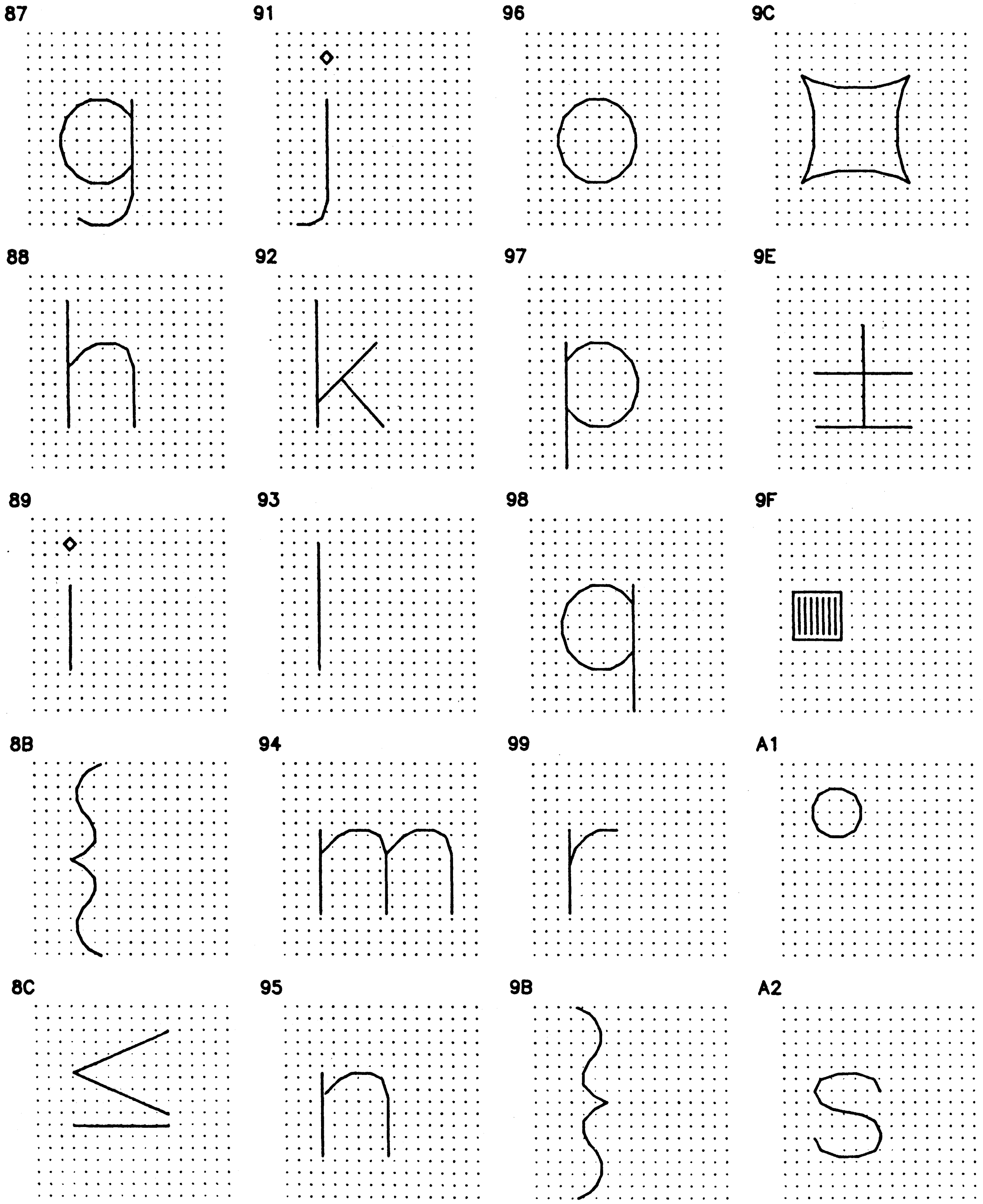
85



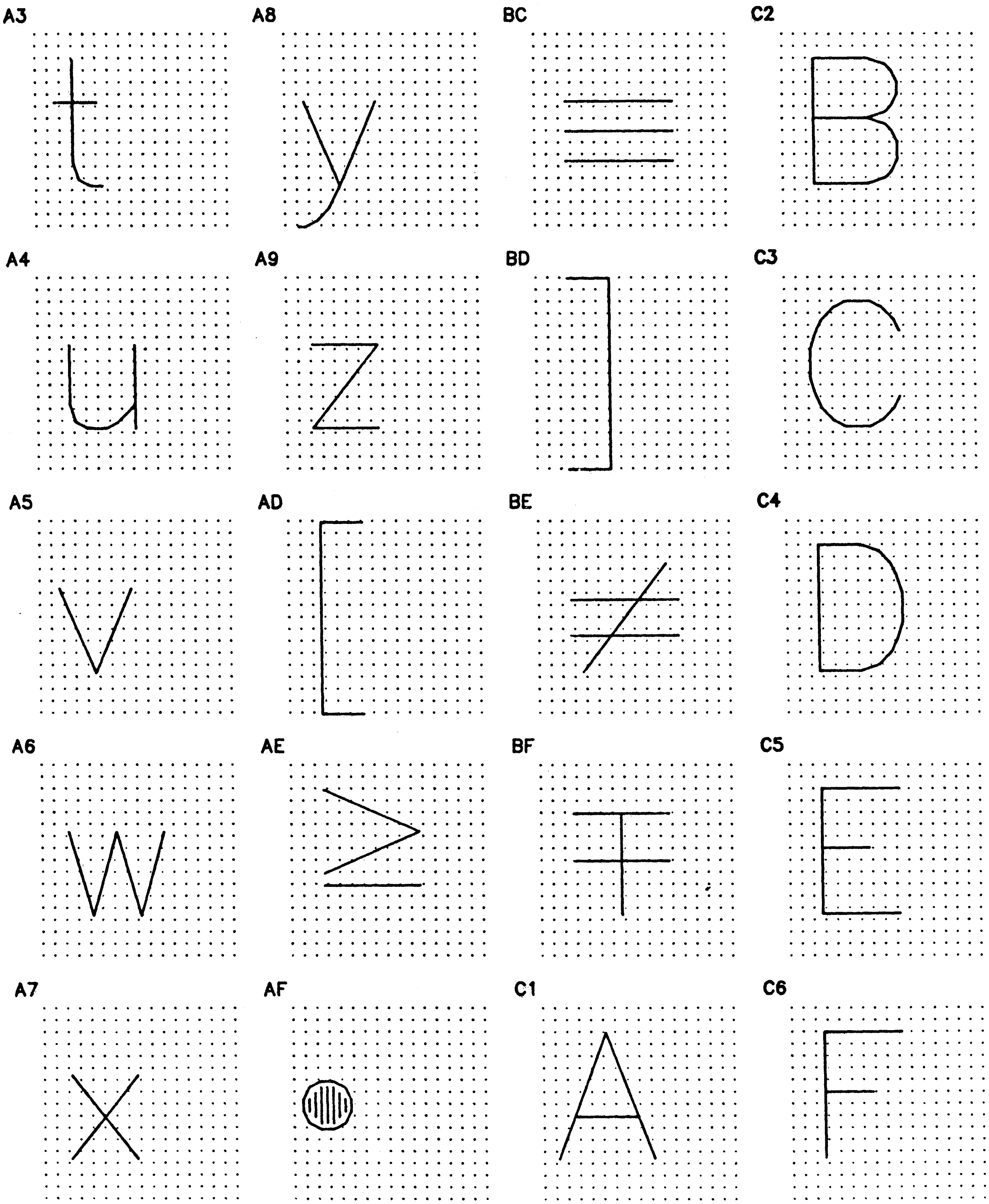
86



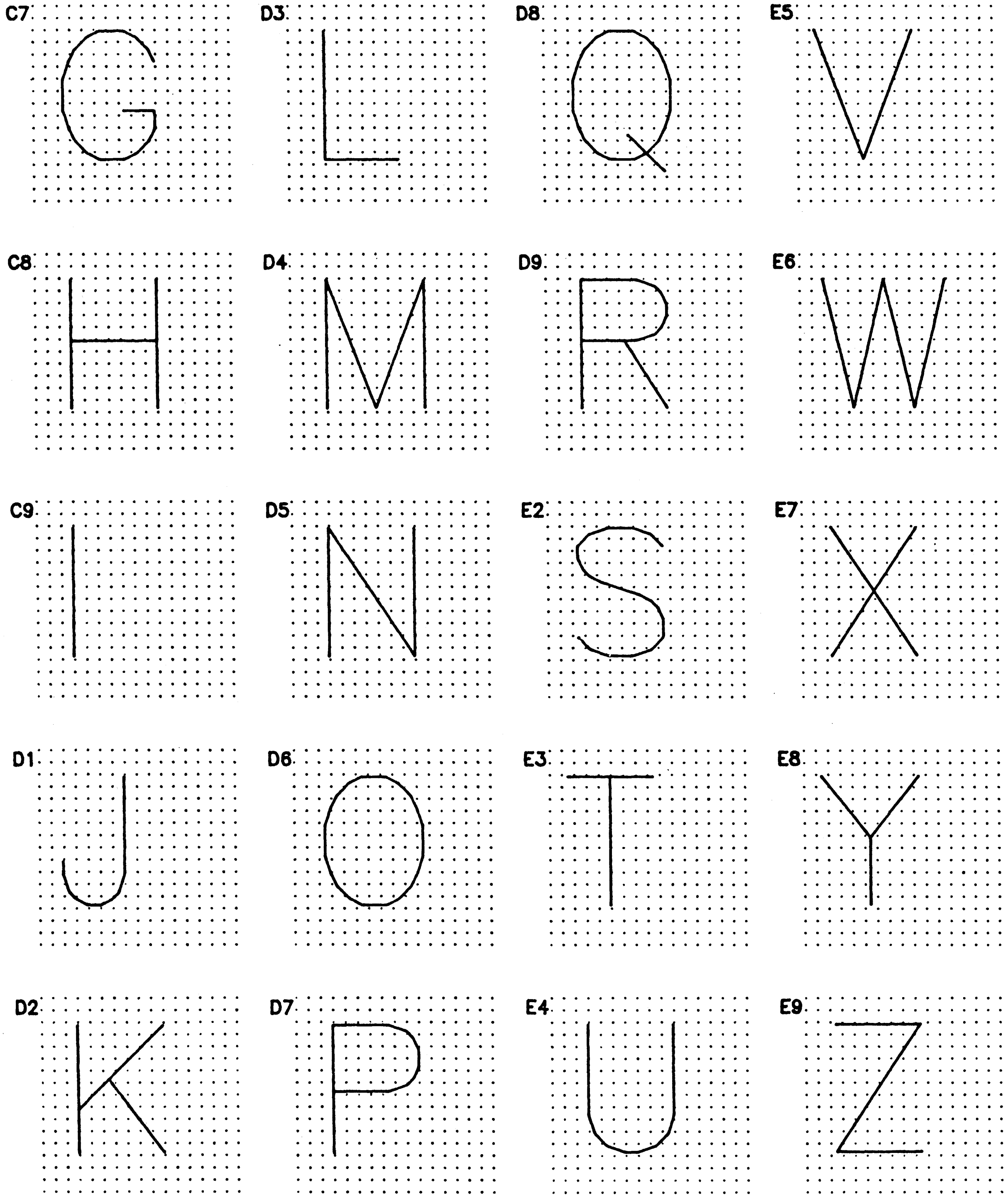
| Sanserif.1 Character Set Grid



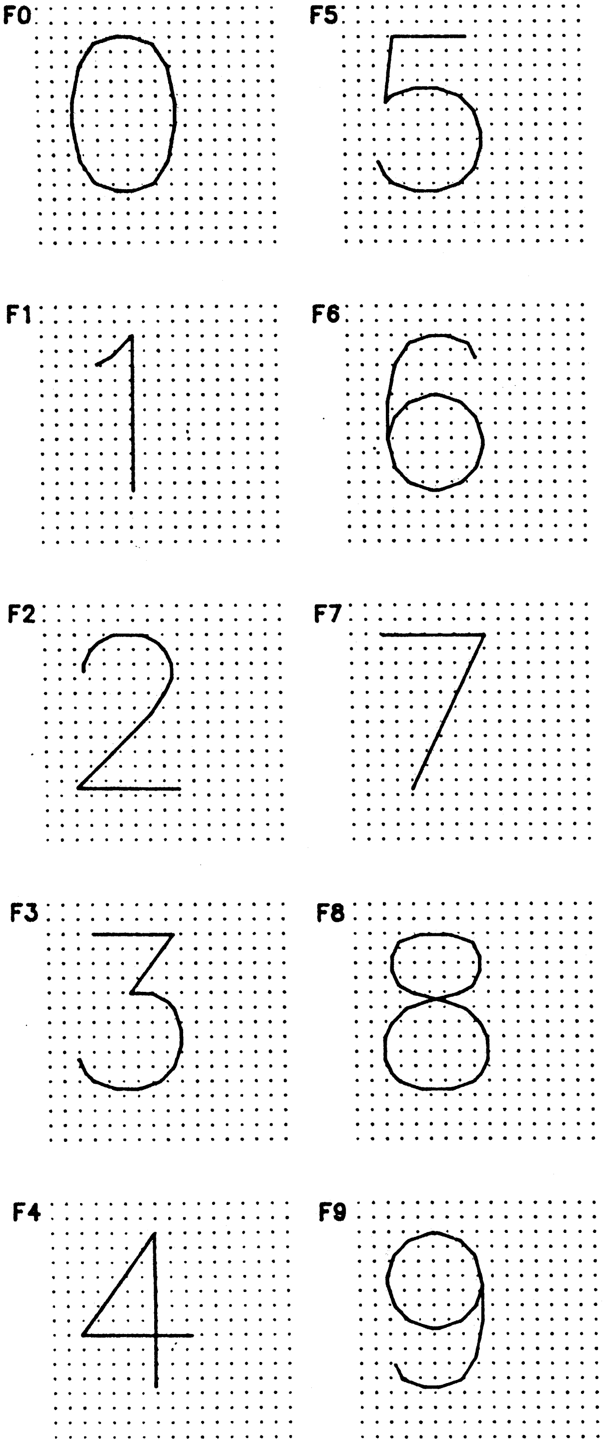
| Sanserif.1 Character Set Grid



| Sanserif.1 Character Set Grid



| Sanserif.1 Character Set Grid



| Sanserif.1 Character Set Grid

APPENDIX B: OBSOLETE SUBROUTINES

The subroutines described in this appendix are considered obsolete. They will remain in *PLOTSYS indefinitely but no maintenance will be performed for them. In general, subroutines still exist in *PLOTSYS which perform the same functions as all of the subroutines in this appendix. (Several subroutines described here never have performed a useful function.)

The descriptions in this appendix are provided to aid in deciphering or modifying existing programs. They should not be used to write new programs. With the exception of the first three subroutines (PSYMB, PSYMLN, and PSYMSV) which are described fully, only a short description is given. A complete description of each of the other subroutines is available on request.

PSYMB

Purpose: PSYMB produces labels and/or symbols for a plot.

Calling Sequence:

```
CALL PSYMB(x,y,height,char,angle,length,mode)
```

Parameters:

x,y — are the floating-point (REAL*4) coordinates of the first character to be drawn. This is either the lower-left corner of the first character of a string or the center of a special symbol. If either coordinate is -0.0 (hexadecimal 80000000), PSYMB continues from the end of the last character string drawn.

height — is the floating-point (REAL*4) character or symbol height in inches.

char — is either the character string to be drawn or a fullword-integer (INTEGER*4) number designating a special symbol.

angle — is the floating-point (REAL*4) angle in degrees at which the character string or special symbol is drawn.

length — is a fullword integer (INTEGER*4). If length is positive, it is the number of characters in char. If length is negative, char is assumed to be the index of a special symbol. The valid negative values for length are:

-1 Move (with the pen up) to (x,y), and draw the symbol centered about this point.

-2 Draw (with the pen down) to (x,y), and draw the symbol centered about this point.

-3 Move (with the pen up) to (x,y), and draw the symbol with the lower-left corner at this point.

-4 Draw (with the pen down) to (x,y), and draw the symbol with the lower-left corner at this point.

mode — is a fullword-integer (INTEGER*4) switch whose value indicates whether x and y are interpreted as absolute or relative coordinates. x and y are interpreted as absolute if mode is zero or omitted. For any other value of mode, x and y are treated as relative coordinates.

Description: Four special control characters exist in the character set. These are backspace (hexadecimal 11), carriage return (hexadecimal 15), superscript (hexadecimal 2E), and subscript (hexadecimal 2F). Backspace backspaces over the previous character. Carriage return returns to the beginning of the line and spaces down by 1.5*<u>height</u>. If a superscript code is encountered in normal text, succeeding characters are raised 0.7*<u>height</u> and drawn with a size of 0.5*<u>height</u>. A subscript code reverts to normal placement and size. Subscript positioning is similar to superscript positioning.

As long as PSYMB is called with a positive value for <u>length</u> and the string contains only characters in the TN character set, the PSYM subroutine may be called instead using exactly the same arguments. If any other symbols are used, an alternate character set may be needed to convert to PSYM. Appendix A and the PALPHA subroutine description describe the use of alternate character sets. The centered symbols produced by calls to PSYMB with a negative value for <u>length</u> may be produced by the PLINE subroutine. See the PLINE subroutine description for information.

The character set used by PSYMB is on the next page. This character set is also available to the PSYM subroutine. To use it, the statement

```
     CALL PALPHA('OLDALPH ',0)
```

should be inserted prior to any calls to PSYM. See the descriptions of the PSYM and PALPHA subroutines for more information.

SYMBOL SET (HEX CODE : SYMBOL OR CONTROL)

00	□	10	|	20	}	30	Σ	40		50	&	60	-	70	0
01	○	11	"BS"	21	{	31	÷	41	A	51	J	61	/	71	1
02	△	12	∧	22	μ	32	≤	42	B	52	K	62	S	72	2
03	+	13	≡	23	π	33	≥	43	C	53	L	63	T	73	3
04	×	14	→	24	Φ	34	△	44	D	54	M	64	U	74	4
05	◇	15	"CR"	25	θ	35	[	45	E	55	N	65	V	75	5
06	⍏	16	≠	26	ψ	36	]	46	F	56	O	66	W	76	6
07	⊼	17	±	27	χ	37	\	47	G	57	P	67	X	77	7
08	Z	18	_	28	ω	38	↑	48	H	58	Q	68	Y	78	8
09	Y	19	≡	29	λ	39	√	49	I	59	R	69	Z	79	9
0A	⌧	1A		2A	∝	3A	†	4A	¢	5A	!	6A	∞	7A	:
0B	✳	1B	∫	2B	δ	3B	‡	4B	.	5B	$	6B	,	7B	#
0C	⧗	1C	⊃	2C	∈	3C	←	4C	<	5C	*	6C	%	7C	@
0D	|	1D	∨	2D	η	3D	×	4D	(	5D	)	6D	—	7D	'
0E	☆	1E	~	2E	"UC"	3E	↑	4E	+	5E	;	6E	>	7E	=
0F	_	1F	≈	2F	"LC"	3F	↓	4F	|	5F	¬	6F	?	7F	"

PSYMLN

Purpose: PSYMLN determines the length of a character string to be drawn.

Calling Sequence:

```
cl = PSYMLN(height,length)
```

Parameters:

height is the floating-point (REAL*4) character height in inches.

length is the fullword-integer (INTEGER*4) number of characters in the string.

Value Returned:

_cl_ The function result is the floating-point (REAL*4) length of the character string in inches.

Description: The PSYMLN subroutine cannot compute the actual length of a character string since the string is not one of the parameters. For the character set produced by the PSYMB subroutine and for the standard character set used by the PSYM subroutine, PSYMLN provides a good approximation of the string length. The PSMLEN subroutine computes the exact string length for any character set available to the PSYM subroutine.

PSYMSV

Purpose: PSYMSV obtains the coordinates of the end of the last symbol drawn by the PSYMB subroutine.

Calling Sequence:

```
CALL PSYMSV(x,y)
```

Values Returned:

x,*y* are floating-point (REAL*4) variables whose values are set to the X and Y coordinate values of the end of the last symbol drawn by PSYMB.

Description: The coordinates returned in *x* and *y* are the absolute coordinates of the point at which PSYMB would begin lettering if called again with coordinates of (-0.0,-0.0).

The PSYMLC subroutine, which has the same calling sequence, obtains the corresponding position for the PSYM subroutine.

PAXFMT

CALL PAXFMT(fwfmt)

PAXFMT provides a four-character IOH format term for use with the numbering produced by PAXIS. Use PAXFRM instead.

PAXFM2

CALL PAXFM2(dwfmt)

PAXFM2 provides an eight-character IOH format term for use with the numbering produced by PAXIS. Use PAXFRM instead.

PDFSYM

CALL PDFSYM(direct,movdef,x,y,h,char,angle,nc,mode)

PDFSYM produces symbols using a user-defined character set. Use PALPHA and PSYM instead.

PDSYMB

CALL PDSYMB(x,y,height,defblk,angle,npen,mode)

PDSYMB draws a single symbol using a definition provided in DEFBLK. Use PALPHA and PSYM instead.

PDTAB

CALL PDTAB(direct,movdef)

PDTAB defines an alternate character set to be used by PSYMB. Use PALPHA instead.

PDTABR, PDTABS

CALL PDTABR(t1,t2,tswt)
CALL PDTABS(t1,t2,tswt)

PDTABR and PDTABS provide the means of manipulating several user-defined character sets by saving and restoring the definition tables. Use PALPHA instead.

PDTABU

CALL PDTABU(tbswt)

PDTABU controls the use of the user-defined character sets by PSYMB. Use PALPHA instead.

PENSPD

CALL PENSPD(nspeed)

PENSPD was intended to control the pen speed. It never performed any function since the pen speed is not controllable. There is no replacement.

PFNMBR

CALL PFNMBR(x,y,height,numb,angle,fmt,mode,str)

PFNMBR is identical to PNUMBR.

PINT

CALL PINT(tens)

PINT was intended to control the intensity of the line drawn. Calling PINT has no effect.

PLOTNO

NPLT = PLOTNO(0)

The result returned by PLOTNO was the current plot number. This number is no longer used or maintained. The value returned by PLOTNO is undefined.

PLTBGN

CALL PLTBGN(paper,pen,pgxwth,npltno)

PLTBGN initializes the Plot Description System (if necessary) and begins a new plot. It is called automatically whenever needed by other subroutines in the PDS. The only parameter value which is used is PGXWTH, which is the same as the parameter of the PLTXMX subroutine. The other three parameters are ignored.

PLTNBR

CALL PLTNBR(npltno)

PLTNBR sets the plot number. The plot number is no longer used and therefore a call to PLTNBR has no effect.

PLTPAP

CALL PLTPAP(paper)

PLTPAP was intended to change the type of paper used on the plotter. Since only one type of paper is available, calling PLTPAP has no effect.

PLTPEN

CALL PLTPEN(pen)

PLTPEN was intended to specify the pen to be used to draw a plot. Calling PLTPEN has no effect. Use PENCHG instead.

PLTTRM

CALL PLTTRM

PLTTRM was intended to generate tape-termination codes for an off-line plotter. Calling PLTTRM no longer has any effect.

POFRES

CALL POFRES(ofpars)

POFRES was intended to be used to restore the coordinate transformation parameters after they had been saved by POFSAV. It has the same calling sequence as POFSAV. Use PLTOFS instead.

PONRST, POFRST

CALL PONRST
CALL POFRST

These are identical to PONWIN and POFWIN, respectively.

PRSTER

CALL PRSTER(xo,yo,delx,dely)

This is identical to PWIND.

PTSYMB

CALL PTSYMB(x,y,height,char,angle,length,mode)

PTSYMB functions exactly like PSYMB but uses the user-defined character set defined by calling PDTAB. Use PALPHA and PSYM instead.

PXABS, PYABS

X = PXABS(0)
Y = PYABS(0)

PXABS and PYABS obtain the absolute linear-rectangular X and Y coordinate of the current pen position. Use PENABS instead.

PXFACT, PYFACT, PXMIN, PYMIN, PXORG, PYORG

```
XFACT = PXFACT(0)
YFACT = PYFACT(0)
XMIN = PXMIN(0)
YMIN = PYMIN(0)
XORG = PXORG(0)
YORG = PYORG(0)
```

Each of these functions returns one of the arguments of the most recent call to PLTOFS. The POFSAV subroutine returns all six at once.

PXREL, PYREL

```
X = PXREL(0)
Y = PYREL(0)
```

PXREL and PYREL obtain the relative linear-rectangular X and Y coordinates of the current pen position. Use PENREL instead.

XABS, YABS

```
XA = XABS(XR)
YA = YABS(YR)
```

XABS and YABS convert the X or Y relative linear-rectangular coordinates specified by XR and YR to the absolute coordinates XA and YA. Use PLOOK instead. PLOOK performs the same function but works in logarithmic and polar modes also.

XREL, YREL

```
XR = XREL(XA)
YR = YREL(YA)
```

XREL and YREL convert the X or Y absolute coordinates specified by XA and YA to the relative linear-rectangular coordinates XR and YR.

APPENDIX C: CALCOMP STANDARD SUBROUTINE INTERFACE

The subroutines described in this appendix were originally developed by California Computer Products Corporation (CALCOMP). It is not recommended that new programs intended for use at MTS installations be written using these subroutines. Programs intended for distribution to non-MTS installations may be written with these subroutines. Programs from elsewhere which use only these routines should run without modification.

The subroutines in this appendix operate by calling other subroutines in the Plot Description System and therefore produce a standard plot description file. It is not recommended that calls to the subroutines in this appendix be mixed with calls to any other plot description subroutines in this volume as unexpected results may occur.

AXIS

Subroutine Description

Purpose: AXIS draws an axis with numbering and a title.

Calling Sequence:

CALL AXIS(x,y,title,nc,axlen,angle,xmin,dx)

Parameters:

_x_,_y_	are the floating-point (REAL*4) absolute coordinates of the axis origin.
title	is a character string to be used as a title.
_nc_	is the fullword-integer (INTEGER*4) number of characters in the axis title. If _nc_ is positive, all labeling is placed on the counterclockwise side of the axis. If _nc_ is negative, all labeling is placed on the clockwise side of the axis.
axlen	is the floating-point (REAL*4) length of the axis in inches.
angle	is the floating-point (REAL*4) angular orientation of the axis in degrees (using a positive counterclockwise convention).
xmin	is the floating-point (REAL*4) number to be drawn at the first tick mark.
_dx_	is the floating-point (REAL*4) increment added to _xmin_ for each succeeding tick mark. The value drawn at the n+1st tick mark is _xmin_+(_dx_*n).

FACTOR

Subroutine Description

Purpose: FACTOR changes the size of the entire plot.

Calling Sequence:

```
CALL FACTOR(fctr)
```

Parameter:

fctr is a floating-point (REAL*4) multiplicative scale factor which is greater than 0.0 and less than or equal to 1.0.

Description: Everything within the plotting region is scaled down by fctr. The FACTOR subroutine must be called before any plotting is done. Calling FACTOR after plotting has begun has no effect.

LINE

Subroutine Description

Purpose: Draws a curve using pairs of data values in two arrays.

Calling Sequence:

```
CALL LINE(x,y,n,k,j,l)
```

Parameters:

x is a floating-point (REAL*4) array of X coordinate values. *x*(*n***k*+1) and *x*(*n***k*+*k*+1) must contain the minimum value and increment such as produced by the SCALE subroutine.

y is a floating-point (REAL*4) array similar to *x* for the Y coordinate values. It must also contain a minimum and increment.

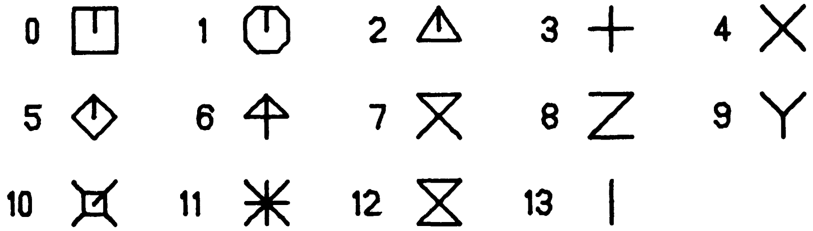
n is the fullword-integer (INTEGER*4) number of elements in the *x* and *y* arrays excluding the minimum and increment.

k is the fullword-integer (INTEGER*4) spacing of the elements in the *x* and *y* arrays. The elements are situated in every *k*th position of the array starting with the first position. *k* should be 1 if the elements are in adjacent positions. If every other value in the arrays are to be used, *k* should be 2. If double-precision arrays are used, *k* must be 2 (or a multiple of 2) to use the more significant part of each element.

j is the fullword-integer (INTEGER*4) plot type.

- *j*=0 produces a line plot with no symbols. The points are connected by a sequence of straight line segments. If *j* is zero, the value of *l* is ignored.
- *j*<0 produces a point plot with a symbol at every *j*th point. No line segments are drawn between points.
- *j*>0 produces a line plot with a symbol at every *j*th point.

l is a fullword integer (INTEGER*4) which specifies a symbol to be centered at the data points. If *j* is zero, the value of *l* is not used, but a value must be present. The table below shows the symbols available. The number to the left of each symbol is the value for *l*.


0
1
2
3
4
5
6
7
8
9
10
11
12
13

NEWPEN

Subroutine Description

Purpose: NEWPEN selects the pen to be used.

Calling Sequence:

```
CALL NEWPEN(ipen)
```

Parameter:

ipen is the fullword-integer (INTEGER*4) number of the pen to be used.

Description: The pen designated by the value of ipen is used to draw all lines and symbols until NEWPEN is called again. If NEWPEN is not called, pen number one is used. The pens used by NEWPEN are numbered 1 to 3 and contain black, red, and blue ink.

NUMBER

Subroutine Description

Purpose: NUMBER draws a number.

Calling Sequence:

```
CALL NUMBER(x,y,height,fnum,angle,lfrac)
```

Parameters:

x,*y* are the floating-point (REAL*4) coordinates of the lower-left corner of the first character of the number.

height is the floating-point (REAL*4) height of the characters to be drawn.

fnum is the floating-point (REAL*4) number to be drawn.

angle is the floating-point (REAL*4) angle in degrees at which the characters are drawn.

lfrac *lfrac* is a fullword integer (INTEGER*4). If *lfrac* is positive, it is the number of digits to the right of the decimal point. The maximum value is nine. If *lfrac* is zero, the number is drawn without a fraction, but with a decimal point. If *lfrac* is -1, the decimal point is not drawn. If *lfrac* is less than -1, the *lfrac*-1 low-order digits are truncated.

<u>OFFSET</u>

Subroutine Description

Purpose: OFFSET defines the parameters of a relative coordinate system used for mapping of data into the plotting region.

Calling Sequence:

```
CALL OFFSET(xmin,xfact,ymin,yfact)
```

Parameters:

<u>xmin</u>,<u>ymin</u> are the floating-point (REAL*4) X and Y values of the relative coordinate system at the origin of the absolute coordinate system.

<u>xfact</u>,<u>yfact</u> are the floating-point (REAL*4) X and Y scale factors in units/inch.

Description: The values given to OFFSET are used to translate relative coordinates into absolute coordinates (inches). This transformation is performed only for coordinates passed to the PLOT subroutine and then only if <u>ic</u>>10. The transformation equations are

$$XA = (XR-\underline{xmin})/\underline{xfact}$$
$$YA = (YR-\underline{ymin})/\underline{yfact}$$

where XA and YA are the resultant absolute coordinates in inches and XR and YR are the relative coordinates passed to the PLOT subroutine.

PLOT

Subroutine Description

Purpose: PLOT moves the pen from its current position to the point (X,Y). PLOT may also be used to move the origin or terminate the plot file.

Calling Sequence:

```
CALL PLOT(x,y,ic)
```

Parameters:

x,y are the floating-point (REAL*4) coordinate values of the new pen position. x and y may be either absolute or relative, depending on the value of IC.

ic is a fullword integer (INTEGER*4) specifying the function to be performed by PLOT. The values for ic are:

2 Draw (with the pen down) to the absolute coordinate (x,y).

3 Move (with the pen up) to the absolute coordinate (x,y).

12 Draw (with the pen down) to the relative coordinate (x,y).

13 Move (with the pen up) to the relative coordinate (x,y).

999 The pen is moved to the end of the plotting region and the plot is terminated.

If the value of ic is negative, the function specified by the absolute value of ic is performed. In addition, the new pen position becomes the origin of the absolute coordinate system.

PLOTS

Subroutine Description

Purpose: PLOTS initializes the plot.

Calling Sequence:

```
CALL PLOTS
```

Description: The PLOTS subroutine should be called before any other plotting subroutines are called. It should be called only once.

All arguments are ignored. Some installations may require arguments to the PLOTS subroutine. The arguments required vary depending on the installation.

SCALE

Subroutine Description

Purpose: SCALE determines the scale factors used by the LINE subroutine.

Calling Sequence:

```
CALL SCALE(x,axlen,n,k)
```

Parameters:

x is a floating-point (REAL*4) array of data for which an adjusted minimum and scale factor are desired. The calculated adjusted minimum is stored in *x*(*n***k*+1); the scale factor is stored in *x*(*n***k*+*k*+1). The array must be dimensioned large enough to contain these values.

axlen is the floating-point (REAL*4) length of the axis for which the data set is to be scaled.

n is the fullword-integer (INTEGER*4) number of elements in the array *x*.

k is the fullword-integer (INTEGER*4) spacing of the elements in the array *x*. The *n* elements are stored in every *k*th position of the array starting with the first position. *k* should be 1 if the elements are in adjacent positions. If every other value in the array is to be used, *k* should be 2. If a double-precision array is used, *k* must be 2 (or a multiple of 2) to use the more significant part of each element.

SYMBOL

Subroutine Description

Purpose: SYMBOL draws symbols.

Calling Sequence:

```
CALL SYMBOL(x,y,height,char,angle,nc)
```

Parameters:

x,*y* are floating-point (REAL*4) coordinates. If *nc* is positive, *x* and *y* are the coordinates of the lower-left corner of the first character drawn. If *nc* is negative, *x* and *y* are the coordinates of the center of the symbol.

height is the floating-point (REAL*4) height of the characters to be drawn.

char If *nc* is positive, *char* is the character string to draw. If *nc* is negative, *char* is an integer number specifying the single symbol to draw.

angle is the floating-point (REAL*4) angle in degrees at which the characters are drawn.

nc is a fullword integer (INTEGER*4). If *nc* is positive, it is the number of characters in *char*. The valid negative values for *nc* are

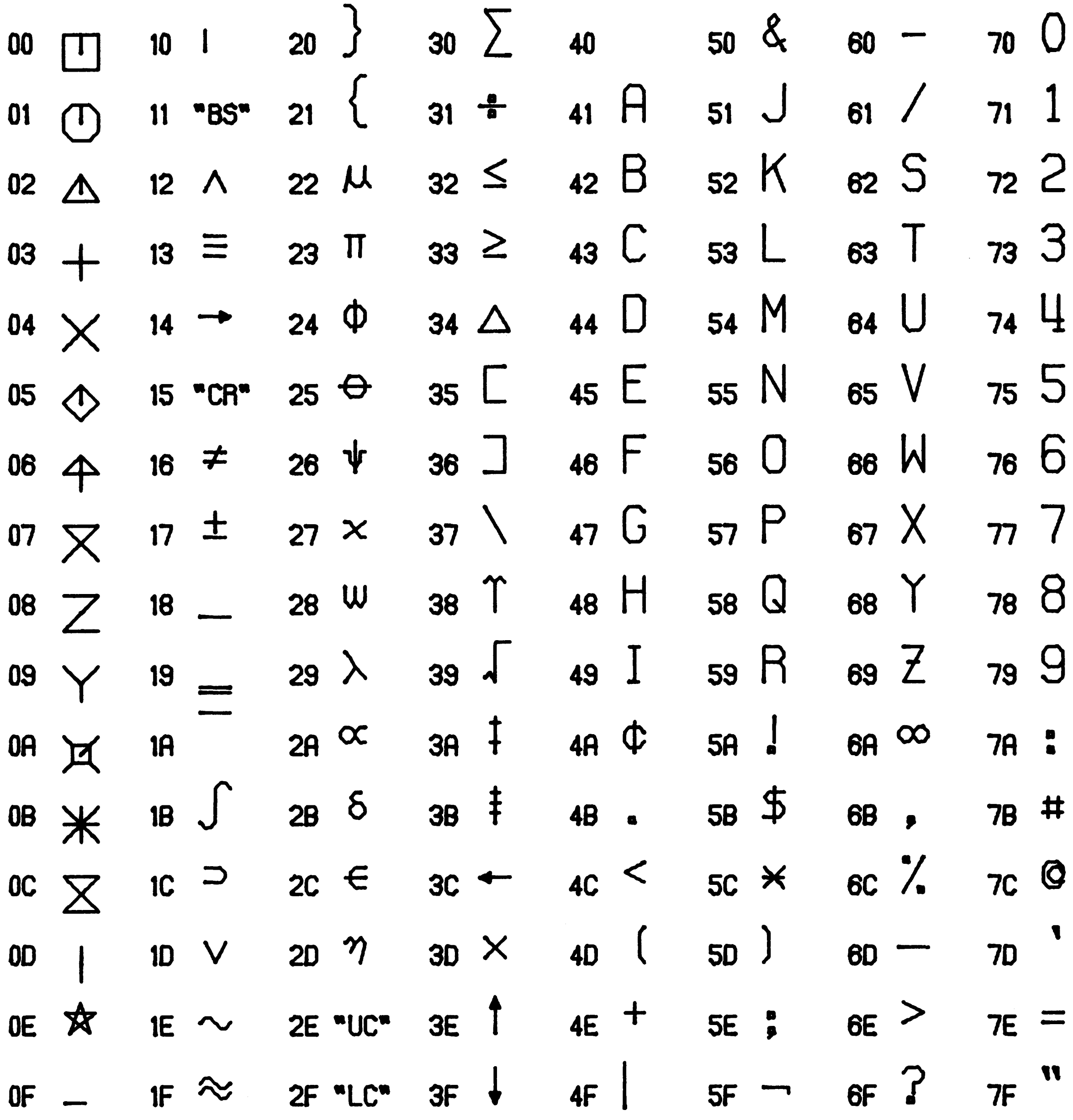
-1 Move with the pen up to the center of the symbol.

-2 Move with the pen down to the center of the symbol.

Description: Four special control characters exist in the character set. These are backspace (hexadecimal 11), carriage return (hexadecimal 15), superscript (hexadecimal 2E), and subscript (hexadecimal 2F). Backspace backspaces over the previous character. Carriage return returns to the beginning of the line and spaces down by 1.5**height*. If a superscript code is encountered in normal text, succeeding characters are raised 0.7**height* and are drawn with a size of 0.5**height*. A subscript code reverts to normal placement and size. Subscript positioning is similar to superscript positioning.

The following page defines the character set produced by SYMBOL.

SYMBOL SET (HEX CODE : SYMBOL OR CONTROL)



00	□	10	|	20	}	30	Σ	40		50	&	60	-	70	0
01	○	11	"BS"	21	{	31	÷	41	A	51	J	61	/	71	1
02	⚠	12	∧	22	μ	32	≤	42	B	52	K	62	S	72	2
03	+	13	≡	23	π	33	≥	43	C	53	L	63	T	73	3
04	×	14	→	24	Φ	34	△	44	D	54	M	64	U	74	4
05	◇	15	"CR"	25	θ	35	[	45	E	55	N	65	V	75	5
06	⍏	16	≠	26	ψ	36	]	46	F	56	O	66	W	76	6
07	⊼	17	±	27	×	37	\	47	G	57	P	67	X	77	7
08	Z	18	_	28	ω	38	↑	48	H	58	Q	68	Y	78	8
09	Y	19	≡	29	λ	39	√	49	I	59	R	69	Z	79	9
0A	⌧	1A		2A	∝	3A	‡	4A	¢	5A	!	6A	∞	7A	:
0B	✳	1B	∫	2B	δ	3B	‡	4B	.	5B	$	6B	,	7B	#
0C	⧗	1C	⊃	2C	∈	3C	←	4C	<	5C	*	6C	%	7C	@
0D	|	1D	∨	2D	η	3D	×	4D	(	5D	)	6D	—	7D	'
0E	☆	1E	∼	2E	"UC"	3E	↑	4E	+	5E	;	6E	>	7E	=
0F	−	1F	≉	2F	"LC"	3F	↓	4F	|	5F	¬	6F	?	7F	"

WHERE

Subroutine Description

Purpose: WHERE obtains the current pen position.

Calling Sequence:

```
CALL WHERE(xc,yc,fctr)
```

Parameters:

_xc_,_yc_ are floating-point (REAL*4) variables into which the current absolute pen position is stored.

fctr is a floating-point (REAL*4) variable into which the current scale factor (from the FACTOR subroutine) is stored.

APPENDIX D: PLOT DESCRIPTION FILE FORMAT

Each line in a plot description contains a four-character header which indicates the type of information contained on that line. The header is followed by a halfword (INTEGER*2) length of the data on the line. This length does not include the length of the header information.

Each plot begins with a PBGN line (which is automatically generated) and ends with a PEND line (which is generated by the PLTEND subroutine). The other types of lines may appear in any order between the PBGN and PEND lines. Several plots may be in one file.

PBGN LINE

The PBGN line contains a scale factor and the size of the plotting region. All sizes and coordinates in the plot must be multiplied by this scale factor to obtain inches. No other information in the PBGN line is used.

Type	Length	Contents
character	4	PBGN
integer	2	constant 36
	8	unused (obsolete)
real	4	maximum paper allowed (from PLTXMX)
real	4	scale factor (multiply everything by this to get inches)
	20	unused (obsolete)

PEND Line

The PEND line is always the last line of a plot.

Type	Length	Contents
character	4	PEND
integer	2	constant 20
real	4	right margin length (from PXMARG)
character	8	time PEND line is generated as HH:MM.SS
character	8	date PEND line is generated as MM-DD-YY

PCOD Line

The PCOD line contains (X,Y) coordinate pairs. The pen is moved in sequence to each of the coordinate pairs given. If a Y value is positive, the pen draws while moving, that is, the pen is down. If a Y value is negative, the pen moves without drawing, i.e., the pen is up. In any case, the pen is moved to (X,|Y|). The coordinates must be multiplied by the scale factor.

Type	Length	Contents
character	4	PCOD
integer	2	length of the following information (in bytes)
real	4	X coordinate
real	4	Y coordinate
.	.	.
.	.	.

The X and Y coordinate pairs are repeated for the number of bytes specified.

PSYM Line

A PSYM line contains the information necessary to draw a character string. Only character strings drawn using the standard character set generate PSYM lines (all others generate PCOD lines). The four special control characters (up, down, carriage return, and backspace) are never present in a PSYM line. In addition, checking is performed by the PSYM subroutine to ensure that all characters placed in a PSYM line lie entirely within the plotting region. The coordinates and height must be multiplied by the scale factor from the PBGN line to obtain inches.

Type	Length	Contents
character	4	PSYM
integer	2	length of the following information (in bytes)
real	4	X coordinate of the lower-left corner of the first character
real	4	Y coordinate of the lower-left corner of the first character
real	4	height of the character string
real	4	angle in degrees of the string
integer	4	number of characters in the string (n)
character	n	the character string

PPEN Line

The PPEN line defines the pen to be used to draw the information on the succeeding lines. It is produced by the PENCHG subroutine. If the color or size is omitted or defaulted, the field is filled with hex zeros.

Type	Length	Contents
character	4	PPEN
integer	2	constant 8
character	4	pen color
hexadecimal	4	pen size

PCOM Line

PCOM lines are ignored by the plotting system. A PCOM line is usually written by the user using the PINFO subroutine, and may contain any information.

Type	Length	Contents
character	4	PCOM
integer	2	length of comment (n)
character	n	comment information

APPENDIX E: PEN COLORS AND SIZES

The tables below list the allowable values for the color and size parameters to the PENCHG subroutine. The pens available are divided into two sets. Pens from different sets may not be mixed in one plot. The PENCHG subroutine does not check the validity of its arguments. However, *CCQUEUE does check and reject all invalid combinations. The pen sets below are installation dependent; other installations may use different sets.

Both parameters to PENCHG are character strings. Only the first four characters are used. If the string is less than four characters, it must be padded on the right with blanks.

The numbers given with the pen sizes are approximate. A size or color followed by an asterisk is the default value. The color name 'BLAK' is equivalent to the name 'BLAC'.

Set 1:

Color	PENCHG Arg.	Size	PENCHG Arg.
black*	'BLAC'	.4mm*	'FINE'
blue	'BLUE'	.4mm*	'FINE'
red	'RED '	.4mm*	'FINE'
green	'GREE'	.4mm*	'FINE'

Set 2:

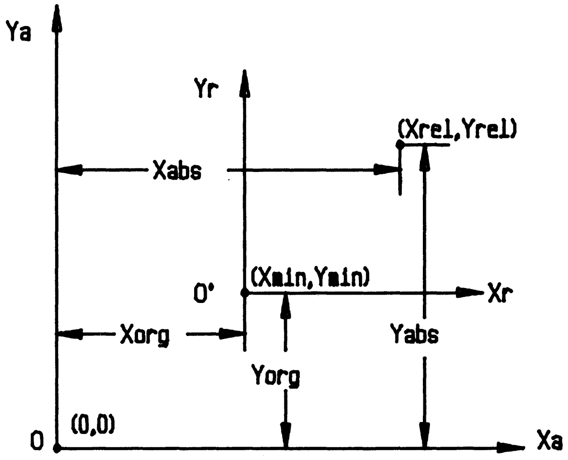
Color	PENCHG Arg.	Size	PENCHG Arg.
black*	'BLAC'	.4mm*	'FINE'
black*	'BLAC'	.6mm*	'MED '
black*	'BLAC'	.8mm*	'WIDE'

Set 2 is not available at all plot times; however, plots using set 2 are plotted at least once a day. Such plots may be queued anytime and will be held by the system until they are plotted. See the public file *PLOTTIMES for the current plotting schedule.

APPENDIX F: COORDINATE TRANSFORMATION EQUATIONS

RECTANGULAR SYSTEMS

The diagram below applies to both linear and logarithmic coordinate systems. In the diagram, the point O is the lower-left corner of the plotting region. The absolute rectangular coordinate axes are labeled Xa and Ya. The point O' (Xmin,Ymin) is the origin of the defined relative coordinate system and is at the absolute coordinate point (Xorg,Yorg). Xabs and Yabs are the absolute coordinates corresponding to the relative coordinates Xrel and Yrel.



In the equations below, Xrel and Yrel are the coordinates of an arbitrary point in the relative coordinate system. Xmin and Ymin are the relative values assigned to the absolute coordinate (Xorg,Yorg). Xfact and Yfact are scale factors. Xabs and Yabs are the resultant absolute coordinates corresponding to Xrel and Yrel.

$$\text{Xabs} = \frac{(\text{Xrel} - \text{Xmin})}{\text{Xfact}} + \text{Xorg} \tag{1}$$

$$\text{Yabs} = \frac{(\text{Yrel} - \text{Ymin})}{\text{Yfact}} + \text{Yorg} \tag{2}$$

$$(3)\qquad Xabs = \frac{\log(Xrel) - Xmin}{Xfact} + Xorg$$

$$(4)\qquad Yabs = \frac{\log(Yrel) - Ymin}{Yfact} + Yorg$$

Equations (1) to (4) are used in various combinations according to the plotting mode. The table below describes the combinations.

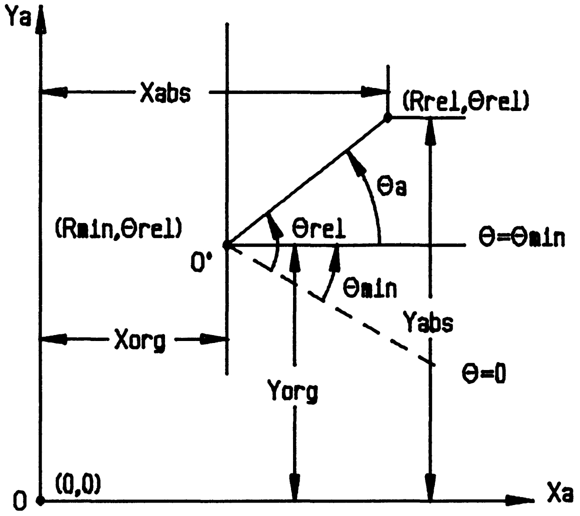
In equations (1) and (2) the values of Xmin and Ymin are in units of Xrel and Yrel, respectively. Xfact and Yfact are scale factors in (units of Xrel)/inch and (units of Yrel)/inch. Xabs, Yabs, Xorg, and Yorg are in inches.

In equations (3) and (4) the values of Xmin and Ymin are integer base 10 exponents. Xfact and Yfact are scale factors having units of exponents/inch. Xabs, Yabs, Xorg, and Yorg are in inches. "log" is the base 10 logarithm.

Plot Type	Equations Used
Linear X, Linear Y	(1) and (2)
Linear X, Log Y	(1) and (4)
Log X, Linear Y	(3) and (2)
Log X, Log Y	(3) and (4)

POLAR SYSTEMS

The diagram below applies to both linear polar and logarithmic polar-coordinate systems. In the diagram, the point O is the lower-left corner of the plotting region. The absolute rectangular coordinate axes are labeled Xa and Ya. The point O’ (Rmin,θrel) is the origin of the defined relative coordinate system and is at the absolute coordinate point (Xorg,Yorg). Xabs and Yabs are the absolute coordinates corresponding to the relative coordinates Rrel and θrel. θa is the physical angular coordinate of the point with respect to the horizontal.



In the equations below, Rrel and θrel are the coordinates of an arbitrary point in the relative coordinate system. Rmin is the relative value of the radial coordinate corresponding to the absolute coordinate (Xorg,Yorg). θa is the physical angular coordinate of the point with respect to the horizontal. θrel and θmin may be in either degrees or radians, and the angular scale factor θfact is such that θa is in either degrees or radians. Normally θfact is dimensionless and the units of θa are the units of θrel and θmin. For most polar plotting, θfact is 1.0 and θmin is 0.0.

$$\theta a = \frac{\theta rel - \theta min}{\theta fact} \tag{5}$$

$$Xabs = \frac{(Rrel - Rmin)}{Rfact} \operatorname{Cos}(\theta a) + Xorg \tag{6}$$

$$Yabs = \frac{(Rrel - Rmin)}{Rfact} \operatorname{Sin}(\theta a) + Yorg \tag{7}$$

$$\text{(8)} \qquad Xabs = \frac{(\log(Rrel) - Rmin)}{Rfact} \cos(\theta a) + Xorg$$

$$\text{(9)} \qquad Yabs = \frac{(\log(Rrel) - Rmin)}{Rfact} \sin(\theta a) + Yorg$$

Equation (5) defines the angle used in the other equations. Equations (6) and (7) are used for linear polar-coordinate systems. Equations (8) and (9) are used for logarithmic polar-coordinate systems.

In equations (6) and (7) the units of Rmin are the units of Rrel. In many cases, the value of Rmin is zero. Rfact is in (units of Rrel)/inch. Xabs, Yabs, Xorg, and Yorg are in inches.

In equations (8) and (9) the value of Rmin is a integer base 10 exponent. Rfact is a scale factor having units of exponents/inch. Xabs, Yabs, Xorg, and Yorg are in inches. "log" is the base 10 logarithm.

INDEX

Reader’s Comment Form

Plot Description System
Volume 11
August 1978
(September 1982 Reprint)

Errors noted in publication:

Suggestions for improvement:

Your comments will be much appreciated. The completed form may be sent to the Computing Center by Campus Mail or U.S. Mail, or dropped in the Suggestion Box at the Computing Center, NUBS, or BSAD.

Date ____________________

Name ____

Address ____

____

____

Publications
Computing Center
University of Michigan
Ann Arbor, Michigan 48109

Update Request Form

Plot Description System
Volume 11
August 1978
(September 1982 Reprint)

Updates to this manual will be issued periodically as errors are noted or as changes are made to MTS. If you desire to have these updates mailed to you, please submit this form.

Updates are also available in the memo files at both the Computing Center and NUBS; there you may obtain any updates to this volume that may have been issued before the Computing Center receives your form. Please indicate below if you desire to have the Computing Center mail to you any previously issued updates.

Name ____

Address ____

____

____

Previous updates needed (if applicable):__________

The completed form may be sent to the Computing Center by Campus Mail or U.S. Mail, or dropped in the Suggestion Box at the Computing Center, NUBS, or BSAD. Campus Mail addresses should be given for local users.

Publications
Computing Center
The University of Michigan
Ann Arbor, Michigan 48109

Users associated with <u>other MTS installations</u> (except the University of British Columbia) should return this form to their respective installations. Addresses are given on the reverse side.

| Addresses of other MTS installations:

| Publications Clerk
| 352 General Services Bldg.
| Computing Services
| The University of Alberta
| Edmonton, Alberta
| Canada T6G 2H1

| Information Officer, NUMAC
| Computing Laboratory
| The University of Newcastle upon Tyne
| Newcastle upon Tyne
| England NE1 7RU

| Rensselaer Polytechnic Institute
| Documentation Librarian
| 310 Voorhees Computing Center
| Troy, New York 12181

| Simon Fraser University
| Computing Centre
| User Services Information Group
| Burnaby, British Columbia
| Canada V5A 1S6

| Wayne State University
| Computing Services Center
| Academic Services Documentation Librarian
| 5950 Cass Ave.
| Detroit, Michigan 48202