Guide to the VAXlab Laboratory Graphics Package

Order Number: AA-KN00C-TE

February 1990

This document describes the VAXlab Laboratory Graphics Package (LGP) routines. It provides an overview of LGP and presents reference information about the procedures you use to plot real-time data or data produced by calculations in two dimensions, three dimensions, and two-dimensional contours from a three-dimensional view.

Revision/Update Information:	This is a revised document.
Operating System and Version:	VMS Version 5.2
Software Version:	VAXlab Software Library Version 1.4

digital equipment corporation
maynard, massachusetts

First Printing, December 1987
Revised, August 1988
Revised, February 1990

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

Restricted Rights: Use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

The Reader's Comments form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC	LN03 PLUS	VAXlab
DECnet	LVP16	VAXstation
DECwindows	MicroVAX	VMS
DRB32	Q-bus	VT
LA50	VAX	VT330
LN03	VAXcluster	

digital™

This document was prepared with VAX DOCUMENT, Version 1.2.

Contents

EXAMPLES

FIGURES

TABLES

Preface

Intended Audience

The *Guide to the VAXlab Laboratory Graphics Package* is intended for use by scientists and engineers working in a laboratory environment. You can use this document initially to learn the basic components of the Laboratory Graphics Package (LGP) application software. Later, you can use it as a reference guide.

This guide assumes an understanding of computer programming concepts and graphics generation techniques and some familiarity with DEC GKS.

Document Structure

The *Guide to the VAXlab Laboratory Graphics Package* is divided into the following chapters:

Chapter Number	Contents
Chapter 1	Presents an overview of the concepts you need to be familiar with before you begin writing programs with the LGP routines.
Chapter 2	Presents the LGP plotting attribute list and describes how to read, modify, and reset attribute values.

Chapter Number	Contents
Chapter 3	Describes how to use LGP to produce multiple plots.
Chapter 4	Provides reference descriptions of the LGP routines, including the routine call syntax and argument descriptions.
Chapter 5	Explains the error-handling facility supported by LGP, describes the error messages, and gives suggested recovery procedures.
Chapter 6	Describes the online sample programs shipped with your VAXlab system and how to access them.

Associated Documents

In addition to this guide, the VAXlab documentation set includes the following guides:

- *Getting Started with VAXlab* is an introduction to the VAXlab system and application software and presents guidelines for developing application programs with VAXlab Softare Library.
- The *Guide to the VAXlab Interactive Data Acquisition Tool* describes how to communicate with VAXlab through the Interactive Data Acquisition Tool (IDAT) to establish parameters for data acquisition and to initiate, control, obtain, analyze, and plot real-time data.
- The *Guide to the VAXlab Laboratory I/O Routines* gives an overview of the LIO facility and describes how to initiate, control, process, and terminate I/O to and from VAXlab I/O devices.
- The *Guide to the VAXlab Laboratory Signal-Processing Routines* describes how to use the signal-processing routines to perform Fourier transforms, correlation functions, and filtering of data.
- The *VAXlab Installation Guide* details how to install the VAXlab software.
- The *VAXlab Master Index* provides a listing of index entries from all of the documents in the VAXlab documentation set.

The following is a list of associated software documents to reference for additional information.

- The *DEC GKS Device Specifics Reference Manual* contains information about all devices supported by DEC GKS.
- The *DEC GKS Reference Manual, Volumes I and II,* contain detailed information about advanced graphics programming concepts and techniques.
- The *DEC GKS User Manual* contains information on programming technique for both the novice and the moderately experienced DEC GKS programmer.
- The *VAX Realtime User's Guide* describes those features of VAX systems that pertain to real-time applications in scientific and industrial settings. If you are unfamiliar with VAX systems, read this guide before you begin using the VAXlab system.

Conventions

The *Guide to the VAXlab Laboratory Graphics Package* uses the following documentation conventions:

Convention	Meaning
Italics	Words appearing in italics refer to a document.
Bold	A boldface word or phrase indicates one of the following: • Emphasis on an important concept or word • A subroutine argument in text • A subsection within a routine or parameter reference description
Return	Press Return on the terminal keyboard.
Ctrl/x	Press Ctrl on the terminal keyboard while simultaneously pressing x. "x" is the variable for C, Y, and Z.
Ellipses	Vertical ellipses indicate that portions of a display or programming example have been excluded for presentation purposes.
[Brackets]	Square brackets enclose optional parameters or arguments in routine lines.
UPPERCASE letters	All VAXlab routine names and DCL commands are presented in UPPERCASE letters.
Lowercase letters	Lowercase letters, in commands and command strings, indicate that you should substitute a word or value.
Print Screen and Shift	Press the Print Screen key (or F2 key) while holding down the Shift key to produce a hardcopy plot directly from the terminal screen.

Chapter 1

What Is LGP?

This chapter provides the following information:

- Overview of laboratory graphics
- LGP plotting routine hierarchy
- Plotting device support
- Using LGP with DECwindows
- Defining logical names
- Including symbolic status definition files
- Device-dependent plotting capabilities
- Producing multicolor plots on the terminal screen
- Using DEC GKS and LGP
- Using Workstation Independent Segment Storage (WISS)
- Producing character output on the terminal screen
- Exiting LGP
- Sample program using LGP routines

1.1 Overview of Laboratory Graphics

In many scientific and engineering applications, information is presented graphically. Although graphical primitives from the Graphical Kernel System (GKS) can be used for plotting mathematical data, it is a complex task for the programmer to use these primitives routinely. LGP is a set of routine calls that enable the user to plot data either gathered from a real-time device or produced by calculations.

You can use LGP to do the following:

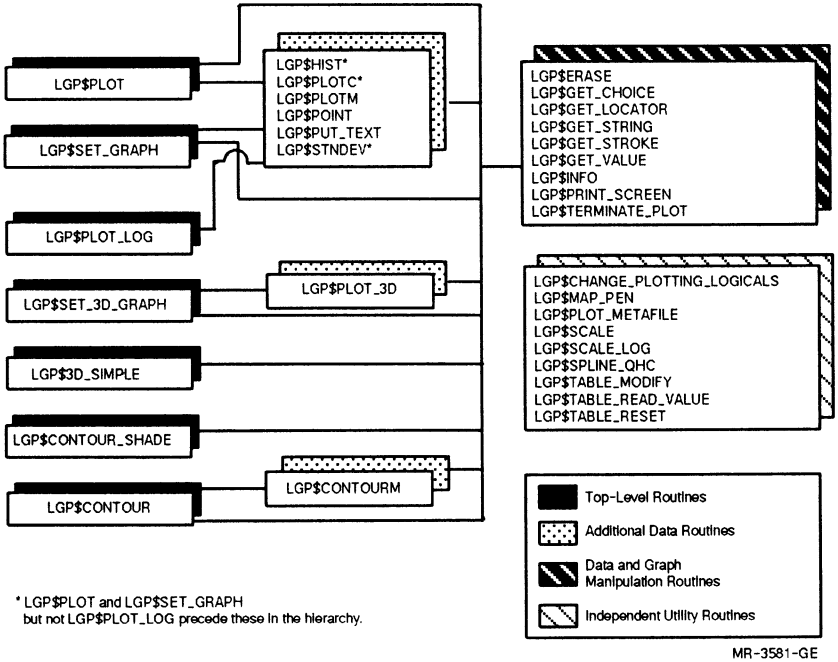
- Produce two-dimensional plots, three-dimensional plots, and contour plots from a three-dimensional view
- Change plotting logical names
- Get status information about a plot
- Map color-to-pen number for hardcopy plots
- Access the autoscaling algorithm
- Access the smooth curve-fitting algorithm (spline)
- Clear plots from the display screen
- Use input devices
- Use output devices not supported by the VAXlab Laboratory Graphics Package

1.2 LGP Plotting Routine Hierarchy

You must follow a hierarchy of operations for VAXlab graphics programming.

Figure 1–1 illustrates the LGP routine call hierarchy.

Figure 1–1: Hierarchy of LGP Routine Calls



1.2.1 Top-Level Routines

The top-level routines create the axis system for the plots. These routines are:

LGP$CONTOUR
LGP$CONTOUR_SHADE
LGP$PLOT
LGP$PLOT_LOG
LGP$SET_GRAPH
LGP$SET_3D_GRAPH
LGP$3D_SIMPLE

You must always call a top-level routine before calling an additional data routine.

For example, call the top-level two-dimensional plotting routines (LGP$PLOT, LGP$SET_GRAPH, or LGP$PLOT_LOG) before calling a two-dimensional additional data plotting routine.

NOTE

With LGP$HIST, LGP$PLOTC, and LGP$STNDEV, you must first call either LGP$PLOT or LGP$SET_GRAPH. LGP$PLOT_LOG does not precede these three calls in the hierarchy.

1.2.2 Additional Data Routines

The additional data routines are:

LGP$CONTOURM
LGP$HIST
LGP$PLOTC
LGP$PLOTM
LGP$PLOT_3D
LGP$POINT
LGP$PUT_TEXT
LGP$STNDEV

Table 1–1 lists the LGP top-level and additional data routines by plotting dimension.

Table 1–1: Two-Dimensional, Three-Dimensional, and Contour Plotting

Plotting Dimensions	Top-Level Routines	Additional Data Routines
Two dimensions	LGP$PLOT LGP$PLOT_LOG LGP$SET_GRAPH	LGP$HIST[1] LGP$PLOTC[1] LGP$PLOTM LGP$POINT LGP$PUT_TEXT LGP$STNDEV[1]
Three dimensions	LGP$SET_3D_GRAPH LGP$3D_SIMPLE	LGP$PLOT_3D
Contour	LGP$CONTOUR LGP$CONTOUR_SHADE	LGP$CONTOURM

[1]LGP$HIST, LGP$PLOTC, and LGP$STNDEV must be preceded by LGP$PLOT or LGP$SET_GRAPH.

1.2.3 Data and Graph Manipulation Routines

The following routines are used to manipulate data and graphs:

LGP$ERASE
LGP$GET_CHOICE
LGP$GET_LOCATOR
LGP$GET_STRING
LGP$GET_STROKE
LGP$GET_VALUE
LGP$INFO
LGP$PRINT_SCREEN
LGP$TERMINATE_PLOT

You can call the data and graph manipulation routines after calling the additional data routines.

1.2.4 Independent Utility Routines

The routines in the following list are independent utility routines. You can use these routines without calling any of the other LGP routines first.

LGP$CHANGE_PLOTTING_LOGICALS
LGP$MAP_PEN
LGP$PLOT_METAFILE
LGP$SCALE
LGP$SCALE_LOG
LGP$SPLINE_OHC
LGP$TABLE_MODIFY
LGP$TABLE_READ_VALUE
LGP$TABLE_RESET

1.3 Plotting Device Support

The VAXlab Laboratory Graphics Package supports most of the devices supported by GKS, including:

- DDIF files
- DECwindows output only, input/output, drawable, and widget devices
- HP7550 pen plotter
- LA12, LA34, LA50, LA75, LA100, and LA210 line printers
- LCG01 inkjet plotter
- LJ250 inkjet and 180 dpi printers
- LN03 PLUS and LN03R laser printers
- LPS40 laser printer
- LVP16 pen plotter (8½ ×11 and 11 ×17)
- PostScript laser printer
- TEKTRONIX 4014 and TEKTRONIX 4107 terminals
- VAXstation II, VAXstation II/GPX, VAXstation 2000, VAXstation 3100, VAXstation 3200, and VAXstation 3500 series workstations
- VT125, VT240, VT241, VT330, and VT340 video display terminals

Table 1–2 lists each supported device in alphabetical order, its GKS workstation type (defined by GKS$WSTYPE), the maximum assignable x-axis and y-axis lengths in inches, and the device-specific x-axis and y-axis default lengths in inches.

Table 1–2: Devices Supported by LGP

Device	GKS$WSTYPE	Maximum Lengths		Default Lengths	
		X Axis	Y Axis	X Axis	Y Axis
DDIF (file)	250	8.5	11.0	5.0	5.0
DECwindows—output only	210	13.0	10.7	5.0	5.0
DECwindows—input/output	211	13.0	10.7	5.0	5.0
DECwindows—drawable	212	13.0	10.7	5.0	5.0
DECwindows—widget	213	13.0	10.7	5.0	5.0
HP7550	53	10.15	7.8	5.0	5.0
LA12	33	11.0	8.0	5.0	5.0
LA34	31	13.0	11.0	5.0	5.0
LA50	32	8.0	11.0	5.0	5.0
LA75	35	8.0	11.0	5.0	5.0
LA100	31	13.0	11.0	5.0	5.0
LA210	34	13.0	11.0	5.0	5.0
LCG01	15	9.9	7.4	5.0	5.0
LJ250 inkjet	91	8.0	11.0	5.0	5.0
LJ250 180 dpi	92	8.0	11.0	5.0	5.0
LN03 PLUS	38	10.41	8.00	5.0	5.0
LN03R	61	10.1	7.6	5.0	5.0
LPS40	61	10.1	7.6	5.0	5.0
LVP16 (8½ ×11)	51	10.15	7.8	5.0	5.0
LVP16 (11 ×17)	52	16.3	10.15	5.0	5.0
TEKTRONIX 4014	72	15.2	10.4	5.0	5.0

Table 1–2 (Cont.): Devices Supported by LGP

Device	GKS$WSTYPE	Maximum Lengths		Default Lengths	
		X Axis	Y Axis	X Axis	Y Axis
TEKTRONIX 4107	82	9.4	7.1	5.0	5.0
VAXstation[1]	41[2]	13.05	10.77	5.0	5.0
VT125 (BW)	12	7.6	4.5	5.0	2.5
VT125 (color)	11	7.6	4.5	5.0	2.5
VT240	14	7.85	5.0	5.0	3.0
VT241	13	9.0	6.0	5.0	4.0
VT330	16	8.5	5.5	5.0	3.5
VT340	17	9.0	6.0	5.0	4.0

[1]These are the values for the 19-inch monitor. For a 15-inch monitor, the same values are used, but the window size is reduced approximately 20%.

[2]This is the workstation type for UIS-based (VMS Workstation Software) VAXstations. The DECwindows workstation type is 211.

1.4 Using LGP with DECwindows

You can use DECwindows as four different workstation types:

- Output only—workstation type 210
- Input/output—workstation type 211
- Drawable (an application window, output only)—workstation type 212
- Widget (input/output, within an application widget)—workstation type 213

Using LGP with a VAXstation running DECwindows is the same as using LGP with a VAXstation running VWS (VMS Workstation Software), with one exception:

- When you define GKS$CONID, you use a special format to define the DECwindows server. See Section 1.5, Defining Logical Names, for more information.

In this guide, the term "VAXstation" refers to VAXstations running either DECwindows or VWS.

Normally, you use workstation type 211 for DECwindows output.

Workstation types 212 and 213 are useful if you want to include LGP graphics in a DECwindows application.

Workstation type 212 (DECwindows drawable) uses a pre-existing window or pixmap and is output only.

Workstation type 212 (DECwindows widget) uses a pre-existing DECwindows toolkit widget.

For more information about DECwindows workstation types, see the *DEC GKS Device Specifics Reference Manual.*

1.5 Defining Logical Names

To run the LGP plotting routines, you must define the workstation type with the following logical name:

- GKS$WSTYPE

See the *DEC GKS Reference Manual, Volume 1* for further information about GKS logical names.

Normally, all output is directed to SYS$OUTPUT. If you want to direct output somewhere else, you also need to define the following additional logical name:

- GKS$CONID

GKS$CONID defines the DECwindows server, serial line, or file to which all output is directed.

Often, an external hardcopy device is connected to a serial line leading to the back of a MicroVAX that is already defined as CSA0. However, if additional serial lines are available, you need to define the serial line name.

GKS$CONID also defines the serial line when terminals are used externally, and can be used to specify a file name for output-only devices.

You can define these logical names in one of the following ways:

- Using DEFINE statements in your LOGIN.COM file similar to the following:

```
$ DEFINE GKS$WSTYPE 14
$ DEFINE GKS$CONID TT:
```

  or for DECwindows output:

```
$ DEFINE GKS$WSTYPE 211
$ DEFINE GKS$CONID nodename::0
```

  where **nodename** is the name of the system on the network where you want the output to be sent. (It is not necessary to define GKS$CONID if you want the output to be sent to your workstation screen.)
- Defining the logical names in a command file available to the group and called from your LOGIN.COM file
- Placing the logical name definitions in the system logical name tables
- Using the LGP$CHANGE_PLOTTING_LOGICALS routine in your program to define or redefine the logical names

NOTE

You must use the LGP$CHANGE_PLOTTING_LOGICALS routine to define logical names for DECwindows workstation types 212 and 213.

The following sample FORTRAN program segment shows how to define logical names:

```
C
C This program segment directs output to the terminal screen
C with workstation type 14 (a VT240 black-and-white terminal).
C
        CALL LGP$CHANGE_PLOTTING_LOGICALS(14,'TT:')
                        .
                        .
                        .
        CALL LGP$PLOT(...)
                        .
                        .
                        .
C
C This program segment directs output to an external device,
C an LVP16 pen plotter, connected to a serial port.
C
        CALL LGP$CHANGE_PLOTTING_LOGICALS(51,'CSA0:')
                        .
                        .
                        .
        CALL LGP$PLOT(...)
                        .
                        .
                        .
        END
```

To discontinue use of GKS$CONID, deassign the plotting logical by typing the following:

```
$ DEASSIGN GKS$CONID
```

If you deassign or do not assign GKS$CONID when using a VAXstation, LGP automatically assigns a unique logical name to each of the windows in a multiple windowed plot.

NOTE

If you are using a VAXstation with multiple windows and you define GKS$CONID before plotting, you must redefine GKS$CONID for each additional window you open.

1.5.1 Defining Logical Names for Foreign Devices

Take the following steps to define logical names to support a foreign device:

NOTE

You must install a device handler before attempting to use any unsupported devices. See the *DEC GKS Reference Manual* for information about installing a device handler.

1. Define the GKS workstation type.

   ```
   $ DEFINE GKS$WSTYPE xxx
   ```

   where

xxx	is any number from 1 to 999 not used in Table 1-2, LGP Devices, Workstation Types, and Maximum and Default Axis Lengths.

2. Define the maximum length of the x axis.

   ```
   $ DEFINE LGP$WKSTN_XXX_XSIZE xx.x
   ```

   where

XXX	is the value you entered for GKS$WSTYPE.
xx.x	is the length of the x axis in inches.

3. Define the maximum length of the y axis.

   ```
   $ DEFINE LGP$WKSTN_XXX_YSIZE xx.x
   ```

   where

XXX	is the value you entered for GKS$WSTYPE.
xx.x	is the length of the y axis in inches.

For some devices, you can pass a bit mask to specify the workstation type. This is particularly useful if you want to specify portrait or landscape mode or paper size for a device, for example.

When defining the maximum lengths of the x and y axes, you still use the workstation decimal value.

For example, to use an LA75 in portrait mode, do the following:

```
$ DEFINE GKS$WSTYPE %x10000023
$ DEFINE LGP$WKSTN_35_XSIZE 8.0
$ DEFINE LGP$WKSTN_35_YSIZE 11.0
```

To use an LA75 in landscape mode, use %x00000023 as the workstation type.

For more information about bit masks, see the *DEC GKS Device Specifics Reference Manual.*

1.6 Including Symbolic Status Definition Files

The LGP error codes and plotting attribute lists are defined in symbolic status definition files.

Your program must include statements for the appropriate symbolic status definition files for the language you are using. For example, the following lines include the symbolic status definition files for the error codes and the plotting attribute list values required by VAX FORTRAN:

```
INCLUDE 'SYS$LIBRARY:LGPDEF.FOR'
INCLUDE 'SYS$LIBRARY:LGPATTDEF.FOR'
```

Table 1-3 shows the error code file names for the programming languages supported by LGP.

Table 1-3: Error Code Symbolic Status Definition Files

Language	Symbolic Status Definition File
VAX Ada	SYS$LIBRARY:LGPDEF.ADA
VAX BASIC	SYS$LIBRARY:LGPDEF.BAS
VAX C	SYS$LIBRARY:LGPDEF.H
VAX FORTRAN	SYS$LIBRARY:LGPDEF.FOR
VAX Macro	SYS$LIBRARY:LGPDEF.MAR
VAX Pascal	SYS$LIBRARY:LGPDEF.PAS

Table 1-4 shows the plotting attribute list file names for the programming languages supported by LGP.

Table 1–4: Plotting Attribute List Symbolic Status Definition Files

Language	Symbolic Status Definition File
VAX Ada	SYS$LIBRARY:LGPATTDEF.ADA
VAX BASIC	SYS$LIBRARY:LGPATTDEF.BAS
VAX C	SYS$LIBRARY:LGPATTDEF.H
VAX FORTRAN	SYS$LIBRARY:LGPATTDEF.FOR
VAX Macro	SYS$LIBRARY:LGPATTDEF.MAR
VAX Pascal	SYS$LIBRARY:LGPATTDEF.PAS

1.7 Device-Dependent Plotting Capabilities

Note the following device-dependent plotting capabilities when you produce hardcopy plots:

- The LA12, LA34, LA50, LA75, LA100, and LA210 printers can be connected directly to any VT125, VT240, VT241, VT330, or VT340 terminal driven by a serial line. You can produce a hardcopy plot directly from the terminal screen on any of these printers by using the LGP$PRINT_SCREEN routine call. If you are using a VT240, VT241, VT330, or VT340 terminal, you can also produce a hardcopy plot directly from the terminal screen by holding down the Shift key while you press the Print Screen key (or F2 key), one of the top-row function keys on the terminal keyboard.
- The LA12, LA34, LA50, LA75, LA100, LA210, and LN03 PLUS and LN03R printers can be connected to a spare serial line and programmed individually to produce hardcopy plots without using a terminal. When programming these printers individually, you need to use the LGP$TERMINATE_PLOT routine to specify the end of the plot. The GKS system requires that all the plotting vectors be drawn before the software can begin rasterization sorting on the complete plot. If you do not use the LGP$TERMINATE_PLOT routine, the plot cannot be produced, and no output is sent to the device.

  The LGP$TERMINATE_PLOT routine is not required if you use the LGP$PRINT_SCREEN routine, because the plot is already produced and remains to be output to a hardcopy device.

- By default, the LVP16 and HP7550 plotters draw the x axis running parallel to the longer edge of the plotting surface. To override the default by specifying that the x axis run parallel to the shorter edge of the plotting surface, you need to modify the symbolic name, LGP$K_ANGLE, from the plotting attribute list. See Chapter 2 for information about the default plotting attribute list and the plotting routine you use to modify the list.

1.8 Producing Multicolor Plots on the Terminal Screen

The VT125 and VT241 terminals can display four colors simultaneously on the screen. The VT340 can display 16 colors, and a color VAXstation can display from eight to 248 colors. In each case, one of the colors is the background color. The actual number of colors that can be displayed on a color VAXstation is determined by the number of bit map planes.

If the maximum number of colors your device can display is being used and a routine call (such as LGP$PLOTM) specifies another color, the first-specified color currently being used is replaced by the most recently specified color. For example, if red, green, and blue are currently being used, and were specified for use in that order, the next color you specify replaces red. If a routine call specifies a color that is currently being used, no replacement is made.

Appendix H, DEC GKS Color Chart, in the *DEC GKS Reference Manual, Volume II* presents the color intensities supported by the various terminals. See the appropriate appendix in the *DEC GKS Reference Manual, Volume II* for information regarding your specific output device.

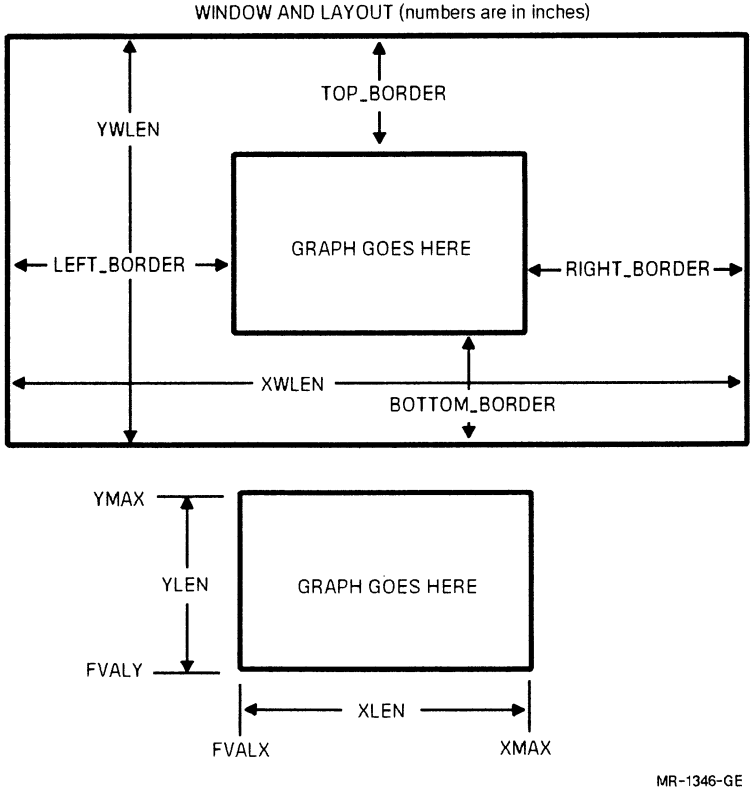
1.9 Using DEC GKS and LGP

To produce more complex plots, you can make calls directly to GKS in conjunction with calls to the LGP plotting routines.

1.9.1 Coordinates

The coordinate system is mapped into the device coordinates shown in Figure 1–2:

Figure 1–2: Window and Layout When Using GKS and Plotting Routines



where

XWLEN	is the total horizontal size of the plot window (in inches).
YWLEN	is the total vertical size of the plot window (in inches).
XLEN	is the length of the x axis (in inches).
YLEN	is the length of the y axis (in inches).
XMAX	is the largest data value of the x axis (in data units).
YMAX	is the largest data value of the y axis (in data units).
FVALX	is the smallest data value of the x axis (in data units).
FVALY	is the smallest data value of the y axis (in data units).
LEFT_BORDER	is the distance in inches between the left edge of the graph and the left edge of the window. The default is 1.75 inches. Modify the symbolic name LGP$K_LBORDER_SIZE in the plotting attribute list to change this value.
RIGHT_BORDER	is the distance in inches between the right edge of the graph and the right edge of the window. The default is 0.5 inches. Modify the symbolic name LGP$K_RBORDER_SIZE in the plotting attribute list to change this value.
BOTTOM_BORDER	is the distance in inches between the bottom of the graph and the bottom of the window. The default is 1.0 inches. Modify the symbolic name LGP$K_BBORDER_SIZE in the plotting attribute list to change this value.
TOP_BORDER	is the distance in inches between the top of the graph and the top of the window. The default is 1.0 inches. Modify the symbolic name LGP$K_TBORDER_SIZE in the plotting attribute list to change this value.

Use the following formulas to calculate:

- Total vertical window size

  YWLEN = YLEN + TOP_BORDER + BOTTOM_BORDER

  The default is YWLEN = YLEN + 2.0.

- Total horizontal window size

  XWLEN = XLEN + LEFT_BORDER + RIGHT_BORDER

  The default is XWLEN = XLEN + 2.25.

- y (world coordinates), the value of the data point y_i, when normalized to world coordinates

  y (world coordinates) = (y_i - FVALY) * YFACTOR

  where

  YFACTOR = YLEN/(YMAX - FVALY)
- x (world coordinates), the value of the data point x_i, when normalized to world coordinates

  x (world coordinates) = (x_i - FVALX) * XFACTOR

  where

  XFACTOR = XLEN/(XMAX - FVALX)

1.9.2 GKS Operating States

You can determine the GKS operating states by using the GKS Inquire function:

```
GKS$INQ_OPERATING_STATE(opstate.wl.r)
```

where

opstate.wl.r is either GKSK_GKCL, GKSK_GKOP, or GKS$K_WSAC.

Table 1–5 shows the numeric code returned by each individual operating state inquiry, and its meaning.

Table 1–5: GKS Operating States

Return Code	Operating State	Meaning
0	GKS$K_GKCL	GKS is closed. This is the operating state before the first call to any of the top-level LGP routines.
1	GKS$K_GKOP	GKS is open. This is the operating state after all active workstations have been erased.
3	GKS$K_WSAC	At least one workstation is active. This is the operating state after the first call to any of the top-level LGP routines.

These operating states are a subset of the GKS operating states. For more information about GKS operating states, see the *DEC GKS Reference Manual, Volume I.*

1.10 Using Workstation Independent Segment Storage (WISS)

You can use Workstation Independent Segment Storage (WISS) with LGP. WISS is a data structure that stores information pertinent to the output primitives contained in a segment. DEC GKS treats WISS as a workstation.

To use WISS, perform the following steps:

1. Open GKS.
2. Open the WISS workstation.
3. Activate the WISS workstation.
4. Make your LGP calls that you want stored in WISS.
5. Deactivate the WISS workstation.
6. Manipulate the data stored in WISS.
7. Close the WISS workstation.
8. Terminate your LGP plots.
9. Close GKS.

For more information, see the *DEC GKS User Manual.*

1.11 Producing Character Output on the Terminal Screen

If character output is directed to the terminal screen during a plotting operation using a write or a print statement from a high-level language, the character output overlays the plot beginning at the upper left corner of the screen. To avoid this situation, you can use the VMS screen management routines to produce a separate character window in a clear or unused portion of the terminal screen.

1.12 Exiting LGP

During normal program development and execution, you use the LGP$TERMINATE_PLOT routine to erase, or close, open workstations before a plotting program exits.

However, a program can abort prematurely because of an error condition, or because a user intentionally aborts a program run using Ctrl/C or Ctrl/Y. If a program run stops before the program executes to completion, the following situations can occur:

- If output is directed to the terminal, the plot(s) may remain on the terminal screen. The output buffer may also be corrupted, displaying unusual characters. You can clear the screen by manually resetting the terminal.
- If output is directed to the terminal, and the program termination was signaled by the user through Ctrl/C, entering CONTINUE after the $ prompt may result in the loss of output.
- There may be a delay in the return of the operating system prompt ($). Enter the DCL command EXIT after the $ prompt as soon as it returns to signal (again) the end of the plotting program.
- If a plotting program does not include the LGP$TERMINATE_PLOT routine call to signal the end of the plotting information, the program may complete execution before all the plotting information can be transferred by serial line. In this case, not all of the plot is produced. By using the LGP$TERMINATE_PLOT routine call in all plotting programs, you can avoid this situation.

1.13 Sample Program Using LGP Routines

Example 1-1 provides a simple illustration of LGP routine call programming. Note the hierarchy of operations in this programming example:

1. Call the LGP$PLOT routine to set up a coordinate system and plot an array of data.
2. Call the LGP$PUT_TEXT routine to write text to the plot.

3. Call the LGP$TERMINATE_PLOT routine to signal the end of the plotting information, clear the plotting screen, and close the open workstation.

Example 1–1: Simple LGP Plot

```
/* this program will plot data on the screen with points emphasized*/
# include stdio
# include <math.h>
# include descrip
# define       NPTS 20
# define       NULL 0
$DESCRIPTOR(mode,"EXSY");
$DESCRIPTOR(title,"Simple LGP plot example");
$DESCRIPTOR(x_lab,"Point number");
$DESCRIPTOR(y_lab,"Random values");
float    x_con[4],                  /* x control array */
         y_con[4],                  /* y control array */
         rad[NPTS],                 /* standard deviation array */
         x_arr[NPTS],               /* x array for lgp */
         y_arr[NPTS];               /* y array for lgp */
long     n[3],                      /* npts,ncol,nrow  */
         igon;                      /* type of point */
unsigned long seed;                 /* random number seed */
int      i,j;                       /* misc looping variables */

 MAIN ()
   {
         x_con[0] = 5;              /* x-axis length in inches */
 x_con[1] = 0;              /* minimum value on x axis */
 x_con[2] = NPTS+1;         /* maximum value on x axis */
 x_con[3] = (int)(NPTS/5); /* delta value on x axis */
 y_con[0] = 3;              /* y-axis length in inches */
 y_con[1] = -200;           /* minimum value on y axis */
 y_con[2] = 200;            /* maximum value on y axis */
 y_con[3] = 80;             /* delta value on y axis */
 n[0]= n[1] = NPTS;         /* set number of points & columns */
 n[2] = 1;                  /* set number of rows */
```

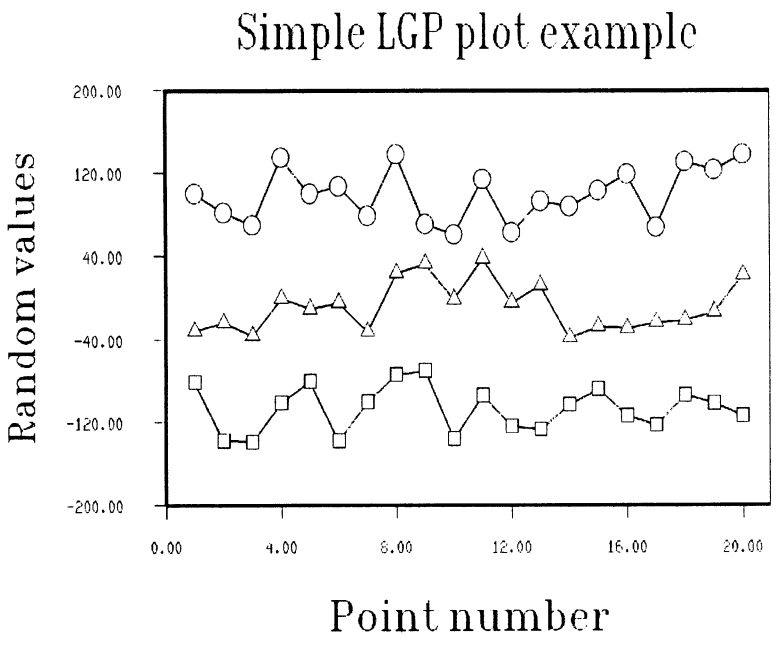
Example 1–1 Cont'd on next page

Example 1–1 (Cont.): Simple LGP Plot

```
/* make the call to set up the plot */
        LGP$PLOT(&1, &mode, x_arr, y_arr, n, &x_lab, &y_lab,NULL,NULL,NULL,
                 x_con, y_con, NULL, &title, NULL,NULL);
        for (j=0; j<3; j++)
        {  for (i=0; i<NPTS; i++)
           { x_arr[i] = i+1;                    /* increment evenly on x axis */
/* fill y_arr with random numbers (range= -40 to 40 + offset) */
             y_arr[i] = (100-(j*100))+((int)(MTH$RANDOM(&seed)))%40;
             rad[i] = .07;
           }                                    /* end i loop */
    igon = (j*2)+4;
/* plot the line */
    LGP$PLOTM(&1, &mode, x_arr, y_arr, n, NULL,NULL,NULL);
/* plot the points */
    LGP$POINT(&1, x_arr, y_arr, rad, &igon, &NPTS,NULL,NULL, &3);
        }                                       /* end j loop */
printf ("\n\nHit return to clear screen\n");
scanf("%c",&seed);                              /* wait for input to clear plot */
lgp$terminate_plot( &1,&1);                     /* clear plot */
  }                                             /* end main */
```

Figure 1–3 shows the graphical output produced by Example 1–1:

Figure 1–3: Simple LGP Plot



Chapter 2

Establishing Plotting Attributes

This chapter describes the following:

- Overview of the plotting attribute list
- Plotting attribute list values
- How to read plotting attribute values
- How to modify plotting attribute values
- How to reset plotting attribute default values

2.1 Overview of the Plotting Attribute List

The plotting attribute list contains information used by the plotting routines to produce graphical representations of data according to your specifications. Plotting attributes include line thickness, type of numerical labeling, height of the x and y axis, offset distances, and fonts.

The values of the plotting attribute list are defined in symbolic status definition files, which must be included in your LGP programs. See Section 1.6, Including Symbolic Status Definition Files, for more information.

2.2 Plotting Attribute List Values

Table 2-1 lists the ranges and default values for the plotting attribute list that are supplied by VAXlab. The attributes are arranged in functional, not alphabetical, order.

Table 2-1: Plotting Attribute List Values

Symbolic Name	Description
LGP$K_NUMLABEL_X	Specifies the type of numeric labeling for the x axis: 1.0: axis labeling is done in floating-point format, for example, 16.349 2.0: axis labeling is done in exponential format, for example, 1.2E+01 **Min** 1.0 **Max** 2.0 **Default** 1.0
LGP$K_NUMLABEL_Y	Specifies the type of numeric labeling for the y axis: 1.0: axis labeling is done in floating-point format, for example, 12.957 2.0: axis labeling is done in exponential format, for example, 9.3E+01 **Min** 1.0 **Max** 2.0 **Default** 1.0
LGP$K_DECIMALANNOT_X	Specifies the number of digits to the right of the decimal point in the x-axis notation. (This applies to both floating-point and exponential representations.) **Min** 0.0 **Max** 5.0 **Default** 2.0

Table 2–1 (Cont.): Plotting Attribute List Values

Symbolic Name	Description
LGP$K_DECIMALANNOT_Y	Specifies the number of digits to the right of the decimal point in the y-axis notation. (This applies to both floating-point and exponential representations.) **Min** 0.0 **Max** 5.0 **Default** 2.0
LGP$K_HORGRID_THICKNESS	Specifies the horizontal grid line thickness. The number 1.0 represents the minimum line thickness a device is capable of drawing. Numbers larger than 1.0 represent integer multiples of the minimum thickness. **Min** 1.0 **Max** 22.0 **Default** 1.0
LGP$K_VERTGRID_THICKNESS	Specifies the vertical grid line thickness. The number 1.0 represents the minimum line thickness a device is capable of drawing. Numbers larger than 1.0 represent integer multiples of the minimum thickness. **Min** 1.0 **Max** 22.0 **Default** 1.0
LGP$K_AXISLINE_THICKNESS	Specifies the axis line thickness. The number 1.0 represents the minimum line thickness a device is capable of drawing. Numbers larger than 1.0 represent integer multiples of the minimum thickness. **Min** 1.0 **Max** 22.0 **Default** 1.0

Table 2–1 (Cont.): Plotting Attribute List Values

Symbolic Name	Description
LGP$K_DATALINE_THICKNESS	Specifies the line thickness used for drawing lines of data. The number 1.0 represents the minimum line thickness a device is capable of drawing. Numbers larger than 1.0 represent integer multiples of the minimum thickness. **Min** **Max** **Default** 1.0 22.0 1.0
LGP$K_POINT_THICKNESS	Specifies the line thickness used for drawing a data point. **Min** **Max** **Default** 1.0 22.0 1.0
LGP$K_STD_THICKNESS	Specifies the line thickness used for drawing a standard deviation marker. **Min** **Max** **Default** 1.0 22.0 1.0
LGP$K_HISTLINE_THICKNESS	Specifies the line thickness used for drawing histograms. **Min** **Max** **Default** 1.0 22.0 1.0
LGP$K_AXISOFFSET_X	Specifies the x-axis offset distance. **Min** **Max** **Default** 0.0 20.0 0.0

Table 2–1 (Cont.): Plotting Attribute List Values

Symbolic Name	Description		
LGP$K_AXISOFFSET_Y	Specifies the y-axis offset distance.		
	Min	**Max**	**Default**
	0.0	20.0	0.0
LGP$K_LABELHEIGHT_X	Specifies the x-axis label height.		
	Min	**Max**	**Default**
	0.03	0.5	0.33
LGP$K_LABELHEIGHT_Y	Specifies the y-axis label height.		
	Min	**Max**	**Default**
	0.03	0.5	0.33
LGP$K_TITLEHEIGHT	Specifies the title label height.		
	Min	**Max**	**Default**
	0.03	0.9	0.39
LGP$K_LABELFONT_X	Specifies the x-axis font number. To determine the x-axis font number, refer to the appropriate appendix for your output device in the *DEC GKS Reference Manual*.		
	Min	**Max**	**Default**
	-32768.0	32768.0	1.0
LGP$K_LABELFONT_Y	Specifies the y-axis font number. To determine the y-axis font number, refer to the appropriate appendix for your output device in the *DEC GKS Reference Manual*.		
	Min	**Max**	**Default**
	-32768.0	32768.0	1.0

Table 2–1 (Cont.): Plotting Attribute List Values

Symbolic Name	Description
LGP$K_TITLEFONT	Specifies the title font number To determine the title font number, refer to the appropriate appendix for your output device in the *DEC GKS Reference Manual.* **Min** -32768.0 **Max** 32768.0 **Default** 1.0
LGP$K_NUMANNOTHEIGHT_X	Specifies the x-axis number notation height. **Min** 0.03 **Max** 0.5 **Default** 0.1
LGP$K_NUMANNOTHEIGHT_Y	Specifies the y-axis number notation height. **Min** 0.03 **Max** 0.5 **Default** 0.1
LGP$K_NUMANNOTFONT_X	Specifies the x-axis numerical label font number. **Min** -32768.0 **Max** 32768.0 **Default** 1.0
LGP$K_NUMANNOTFONT_Y	Specifies the y-axis numerical label font number. **Min** -32768.0 **Max** 32768.0 **Default** 1.0

Table 2–1 (Cont.): Plotting Attribute List Values

Symbolic Name	Description
LGP$K_ANGLE	Specifies the writing angle used by the LVP16 or the HP7550 plotter. If LGP$K_ANGLE is 0.0, then the plot is drawn with the x axis parallel to the longer edge of the paper. If LGP$K_ANGLE is 1.0, then the plot is drawn with the x axis running parallel to the shorter edge of the paper. **Min** **Max** **Default** 0.0 1.0 0.0
LGP$K_LBORDER_SIZE	Specifies the distance in inches between the left edge of the graph and the left edge of the window. **Min** **Max** **Default** 0.0 5.0 1.75
LGP$K_RBORDER_SIZE	Specifies the distance in inches between the right edge of the graph and the right edge of the window. **Min** **Max** **Default** 0.0 5.0 0.50
LGP$K_BBORDER_SIZE	Specifies the distance in inches between the bottom of the graph and the bottom of the window. **Min** **Max** **Default** 0.0 5.0 1.0
LGP$K_TBORDER_SIZE	Specifies the distance in inches between the top of the graph and the top of the window. **Min** **Max** **Default** 0.0 5.0 1.0

Table 2–1 (Cont.): Plotting Attribute List Values

Symbolic Name	Description
LGP$K_BORDER	Specifies whether or not a border is drawn. Set LGP$K_BORDER to 0.0 if you do not want a border. **Min** **Max** **Default** 0.0 1.0 1.0
LGP$K_SCREENDIV	Specifies the number of line segments in which to divide the plotting window during real-time plotting. See the appropriate volume of the *DEC GKS Reference Manual* for the output device you are using. **Min** **Max** **Default** 1.0 100.0 5.0
LGP$K_ERASE_COND	Specifies whether you have full-screen or partial erase during real-time plotting. Set LGP$K_ERASE_COND to 1.0 to enable full-screen erase. **Partial** **Full-Screen** **Default** 0.0 1.0 0.0
LGP$K_TBORDER_MAJOR_TICKMARK	Specifies whether or not the top border of your graph has major incremental tickmarks. Set LGP$K_TBORDER_MAJOR_TICKMARK to 1.0 to enable major incremental tickmarks. **No** **Yes** **Default** 0.0 1.0 0.0

Table 2–1 (Cont.): Plotting Attribute List Values

Symbolic Name	Description
LGP$K_RBORDER_MAJOR_TICKMARK	Specifies whether or not the right border of your graph has major incremental tickmarks. Set LGP$K_RBORDER_MAJOR_TICKMARK to 1.0 to enable major incremental tickmarks. **No** **Yes** **Default** 0.0 1.0 0.0
LGP$K_TBORDER_MINOR_TICKMARK	Specifies whether or not the top border of your graph has minor incremental tickmarks. Set LGP$K_TBORDER_MINOR_TICKMARK to 1.0 to enable minor incremental tickmarks. **No** **Yes** **Default** 0.0 1.0 0.0
LGP$K_BBORDER_MINOR_TICKMARK	Specifies whether or not the bottom border of your graph has minor incremental tickmarks. Set LGP$K_BBORDER_MINOR_TICKMARK to 1.0 to enable minor incremental tickmarks. **No** **Yes** **Default** 0.0 1.0 0.0
LGP$K_LBORDER_MINOR_TICKMARK	Specifies whether or not the left border of your graph has minor incremental tickmarks. Set LGP$K_LBORDER_MINOR_TICKMARK to 1.0 to enable minor incremental tickmarks. **No** **Yes** **Default** 0.0 1.0 0.0

Table 2-1 (Cont.): Plotting Attribute List Values

Symbolic Name	Description			
LGP$K_RBORDER_MINOR_TICKMARK	Specifies whether or not the right border of your graph has minor incremental tickmarks. Set LGP$K_RBORDER_MINOR_TICKMARK to 1.0 to enable minor incremental tickmarks.			
	No	**Yes**	**Default**	
	0.0	1.0	0.0	
LGP$K_NUM_MINOR_TICKS	Specifies the number of minor tickmarks between two major tickmarks.			
	Min	**Max**	**Default**	
	0.0	100.0	0.0	
LGP$K_PLACE_TICKMARK	Specifies the placement of major and minor tickmarks on the graph. Set LGP$K_PLACE_TICKMARK to 1.0 to place the major and minor tickmarks inside the graph. Set LGP$K_PLACE_TICKMARK to 2.0 to place major and minor tickmarks on the graph. The default is 0.0, to place major and minor tickmarks outside the graph.			
	Outside	**Inside**	**On**	**Default**
	0.0	1.0	2.0	0.0
LGP$K_META_TYPE	Specifies whether a metafile is a GKS or a CGM metafile. Set LGP$K_META_TYPE to 1.0 to specify a CGM metafile.			
	GKS	**CGM**	**Default**	
	0.0	1.0	0.0	

2.3 Reading Plotting Attribute Values

To read the current value of a specified plotting attribute, use the LGP$TABLE_READ_VALUE routine.

1. Include the LGP symbolic definition files:

```
INCLUDE 'SYS$LIBRARY:LGPDEF.FOR'
INCLUDE 'SYS$LIBRARY:LGPATTDEF.FOR'
```

   This example is written in VAX FORTRAN. Be sure to include the VAXlab-supplied symbolic definition files appropriate for your programming language.

2. Declare the data types and variables you need to use in your program.

```
REAL*4 VALUE        !Returns the current value of the attribute
INTEGER*4 STATUS    !Returns the status of the plotting operation
```

3. Specify the symbolic name of the attribute, and read the current value into the return variable.

```
CALL LGP$TABLE_READ_VALUE (LGP$K_AXISOFFSET_X, VALUE, STATUS)
```

   This routine call requests the current value of the x-axis offset value.

4. Check the status of the operation.

```
IF (STATUS .NE. LGP$_SUCCESS) THEN
    CALL LIB$SIGNAL(%VAL(STATUS))
ELSE
    .
    .
    .
```

   If the operation is successful, the program continues to execute. If the operation is not successful, the appropriate condition value is returned.

NOTE

You can also check the status of the operation by omitting the **status** in the routine call. If status is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR. See Section 5.2, Checking Routine Call Status, for further information.

5. Print out the attribute value.

```
.
.
.
    TYPE 10, VALUE
10  FORMAT (2X,'x-axis offset = ',F6.4)

    ENDIF
```

The current value of LGP$K_AXISOFFSET_X is displayed on the terminal screen.

2.4 Modifying Plotting Attribute Values

To make changes to the current values contained in the plotting attribute list, use the LGP$TABLE_MODIFY routine.

The following example modifies the x-axis and y-axis offset distances.

1. Include the LGP symbolic definition files.

```
INCLUDE 'SYS$LIBRARY:LGPDEF.FOR'
INCLUDE 'SYS$LIBRARY:LGPATTDEF.FOR'
```

   This example is written in VAX FORTRAN. Be sure to include the VAXlab-supplied symbolic definition file appropriate for your programming language.

2. Declare the data types and variables you need to use in your program.

```
INTEGER*4 STATUS    !Returns the status of the plotting operation

XOFFSET = 10.0      !New x-axis offset value
YOFFSET = 10.0      !New y-axis offset value
```

3. Modify the x-axis offset and check the status of the operation.

```
CALL LGP$TABLE_MODIFY (LGP$K_AXISOFFSET_X, XOFFSET, STATUS)
  IF (STATUS .NE. LGP$_SUCCESS) THEN
      CALL LIB$SIGNAL(%VAL(STATUS))
  ENDIF
```

4. Modify the y-axis offset and check the status of the operation.

```
CALL LGP$TABLE_MODIFY (LGP$K_AXISOFFSET_Y, YOFFSET, STATUS)
  IF (STATUS .NE. LGP$_SUCCESS) THEN
      CALL LIB$SIGNAL(%VAL(STATUS))
  ENDIF
```

5. You can read the modified x-axis and y-axis offset values by following the steps outlined in Section 2.3, Reading Plotting Attribute Values.

2.5 Resetting Plotting Attribute Default Values

To restore the default values of the plotting attribute list you previously modified with the LGP$TABLE_MODIFY routine, use the LGP$TABLE_RESET routine:

```
CALL LGP$TABLE_RESET
```

Chapter 3

Creating Multiple Plots

This chapter provides the following information on how to use LGP to create multiple plots on your terminal screen using either DECwindows or VWS (VMS Workstation Software):

- A description of how to use workstation numbers and plotting windows
- An example of how to create multiple plots

3.1 Workstation Number and Plotting Windows

The first argument in many of the plotting routines is **ws_number**, the workstation number. You use the workstation number to produce multiple plots on the terminal screen with the ability to modify plots individually by addressing them as separate workstations.

A workstation number is arbitrary. Its function is to act as a pointer to a set of attributes describing the size and position of the plotting window. The plotting window is denoted by a call to one of the following top-level plotting routines:

- LGP$PLOT
- LGP$SET_GRAPH
- LGP$PLOT_LOG
- LGP$SET_3D_GRAPH
- LGP$3D_SIMPLE

- LGP$CONTOUR
- LGP$CONTOUR_SHADE

NOTE

See Figure 1-1 for an illustration of the LGP routine call hierarchy, including the top-level routine calls.

You can open a maximum of 16 windows at any time. Because of this windowing feature, you can produce up to 16 separate plots at your workstation. You can display these plots individually on the terminal screen.

The size of the plotting window is limited to the maximum physical size of the display screen. Since the maximum size of the window and layout allow enough space for the placement of the graph title and labels for the x and y axes, the actual size of the window is calculated using the following equations:

- Total length of x axis (XLEN) in inches, where xcontrol(1) = XLEN:

  xcontrol(1) = 5 + LEFT_BORDER (1.75) + RIGHT_BORDER (0.5)

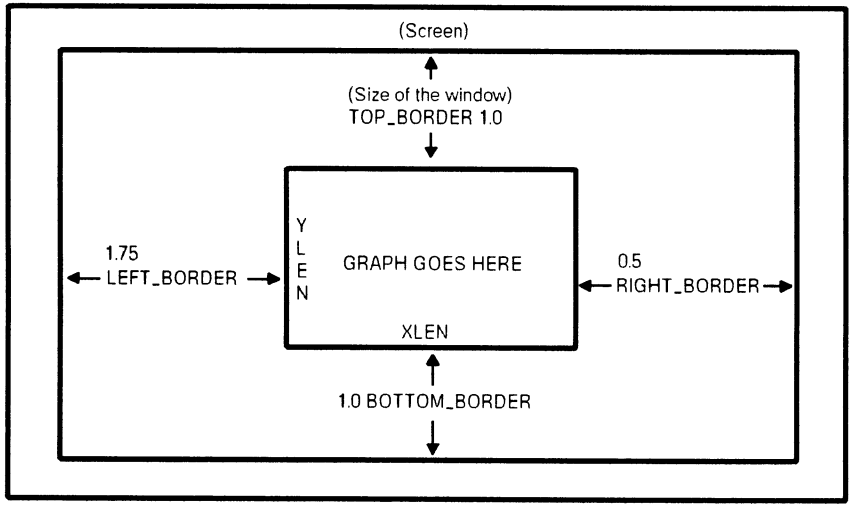
  xcontrol(1) = 7.25
- Total length of y axis (YLEN) in inches, where ycontrol(1) = YLEN:

  ycontrol(1) = 5 + TOP_BORDER (1.0) + BOTTOM_BORDER (1.0)

  ycontrol(1) = 6.0

Figure 3-1 shows the plotting window and layout.

Figure 3–1: Window and Layout When Using the Plotting Routines



NOTE: ALL devices

All window placement on a screen/device can be controlled by modifying the attribute LGP$K_AXISOFFSET_X and/or LGP$K_AXISOFFSET_Y in the plotting attribute list in Chapter 3.

By default your window (origin) is placed at the lower left corner on device.

MR-1347-GE

Again, you can change the values of LEFT_BORDER, RIGHT_BORDER, TOP_BORDER, and BOTTOM_BORDER by modifying the appropriate plotting attribute values. See Table 2–1 for a list of plotting attribute defaults and ranges for the following plotting attributes:

- LGP$K_LBORDER_SIZE
- LGP$K_RBORDER_SIZE
- LGP$K_TBORDER_SIZE
- LGP$K_BBORDER_SIZE

3.2 Example of Creating Multiple Plots

To produce multiple plots on VAXlab-supported graphics devices, do the following:

1. Calculate the fixed origin (the x,y coordinate forming the lower left corner of the axis) of the plots to be displayed simultaneously.
2. Using LGP$PLOT, create and plot data in the first axis system.
3. Modify the LGP$K_AXISOFFSET_X and LGP$K_AXISOFFSET_Y (the x-axis and y-axis offset distances) plotting attribute values.
4. Using LGP$PLOT again, create another axis system at the location specified by the modified x-axis and y-axis offset distances.

NOTE

For VT devices, if you do not modify the x-axis and y-axis offset distances before you create the second axis system, the second axis system supersedes the first axis system.

For VAXstations (using either DECwindows or VWS), if you do not modify the x-axis and y-axis offset distances before you create the second axis system, the second axis system is superimposed over the first axis system; it does not supersede it. In this case, you can use the mouse to move the second axis system to another location on the display screen.

You can modify the x-axis and y-axis offset distances up to 16 times to produce 16 separate windows. Then, you can reference the individual windows by the **ws_number** assigned to each window. You can close a window (LGP$TERMINATE_PLOT) or create a new one at any time. Once a window is closed, you can no longer access it.

Example 3-1 is a sample FORTRAN program illustrating the use of the LGP$PLOT and LGP$SET_GRAPH routines to create multiple plots for simultaneous display on a graphics terminal.

Example 3–1: Creating Multiple Plots on a Graphics Terminal

```
C PROGRAM: multi_window

C The following program shows how to use several LGP routines to
C create multiple plots for simultaneous display on a graphics
C terminal screen.

C Include the symbolic-value definition files:

      INCLUDE 'SYS$LIBRARY:LGPDEF.FOR'
      INCLUDE 'SYS$LIBRARY:LGPATTDEF.FOR'

C Declare variables and data types:

      INTEGER         n(3)              ! contents of data array
      REAL            xarray(5)         ! array of x-axis values
      REAL            yarray(5,3)       ! array of y-axis values
      INTEGER         igrid             ! number of grid lines
      REAL            xcontrol(4)       ! array of x-axis parameters
      REAL            ycontrol(4)       ! array of y-axis parameters
      REAL            color(3)          ! combination of red, green,
                                        ! and blue color intensities

C Assign constant values to the three elements of the n array:

      n(1) = 5      ! npts - number of data values per line segment
      n(2) = 5      ! ncol - number of columns to be plotted
      n(3) = 3      ! nrow - number of data segments to be plotted

C Assign constant values to the five elements of the xarray.  Assign
C constant values to the 15 elements of the yarray:

      DO i = 1,5
            xarray(i) = i
            yarray(i,1) = i
      ENDDO
      yarray(1,2) = -0.81
      yarray(2,2) = 1.2
      yarray(3,2) = 1.5
      yarray(4,2) = 2.
      yarray(5,2) = 1.
      yarray(1,3) = 0.1
      yarray(2,3) = 0.2
      yarray(3,3) = 0.3
      yarray(4,3) = 0.2
      yarray(5,3) = 0.1
```

Example 3–1 Cont'd on next page

Example 3–1 (Cont.): Creating Multiple Plots on a Graphics Terminal

```
C Assign constant values to the three elements of the color array.
C Specify the color set up for the VT241 graphics terminal:

        color(1) = 1.0     ! red intensity
        color(2) = 0.14    ! green intensity
        color(3) = 0.57    ! blue intensity

C Specify the x-axis and y-axis parameters:

        xcontrol(1) = 1.6     ! xlen - length of x axis in inches
        xcontrol(2) = 0.      ! xlow - first annotated value for x (in units)
        xcontrol(3) = 6.      ! xhi - last annotated value for x (in units)
        xcontrol(4) = 2.      ! xdelta - delta incremental value for x

        ycontrol(1) = 0.8     ! ylen - length of y axis in inches
        ycontrol(2) = -2.     ! ylow - first annotated value for y (in units)
        ycontrol(3) = 6.      ! yhi - last annotated value for y (in units)
        ycontrol(4) = 2.      ! ydelta - delta incremental value for y

C Specify the placement of the first window on the screen:

        CALL LGP$TABLE_MODIFY(LGP$K_AXISOFFSET_X,3.8)
        CALL LGP$TABLE_MODIFY(LGP$K_AXISOFFSET_Y,2.3)

C Create the axis system for the first graph, "Test data", at the location
C specified by the x-axis and y-axis offset distances, and plot one data set:

        CALL LGP$PLOT(1,'EXMY',xarray,yarray,n,,'Y',,,
     -          ,xcontrol,ycontrol,color,'Test data')

C Specify the placement of the second window on the screen:

        CALL LGP$TABLE_MODIFY(LGP$K_AXISOFFSET_X,0.)
        CALL LGP$TABLE_MODIFY(LGP$K_AXISOFFSET_Y,0.)

C Create the axis system for the second graph, "Grid", at the location
C specified by the x-axis and y-axis offset distances, and set up the grid:

        igrid = 1
        CALL LGP$PLOT(2,'IXSY',,yarray,1,'X','Y',,,
     -          igrid,xcontrol,ycontrol,color,'Grid')

C Specify the placement of the third window on the screen:

        CALL LGP$TABLE_MODIFY(LGP$K_AXISOFFSET_X,0.)
        CALL LGP$TABLE_MODIFY(LGP$K_AXISOFFSET_Y,2.3)
```

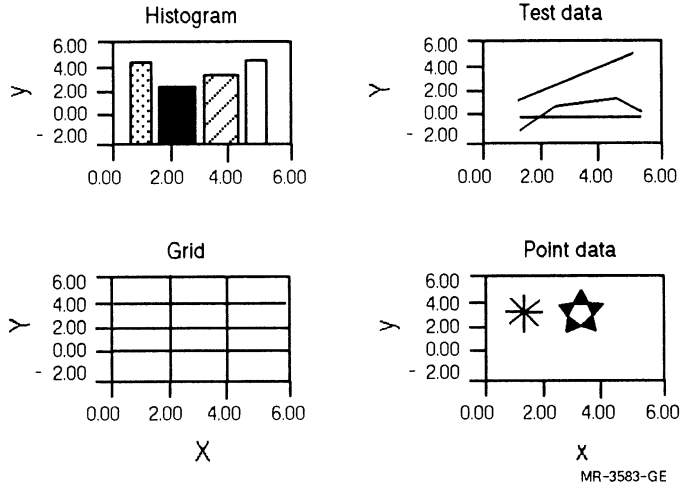
Example 3–1 Cont'd on next page

Example 3–1 (Cont.): Creating Multiple Plots on a Graphics Terminal

```
C Set up the axis coordinate system for the third graph, "Histogram".
C This routine does not plot a data set:

        CALL LGP$SET_GRAPH(3,xcontrol,ycontrol,,,'Histogram',,'y')

C Specify the placement of the fourth window:

        CALL LGP$TABLE_MODIFY(LGP$K_AXISOFFSET_X,3.8)
        CALL LGP$TABLE_MODIFY(LGP$K_AXISOFFSET_Y,0.)

C Set up the axis coordinate system for the fourth graph, "Point data".
C This routine does not plot a data set:

        CALL LGP$SET_GRAPH(4,xcontrol,ycontrol,,,'point data','x','y')

C Plot the first two histogram bars for the third graph:

        CALL LGP$HIST(3, 0.5, 1.0, 4.5, 1,,color,2)
        CALL LGP$HIST(3, 1.2, 2.3, 3.0, 1,,,1)

C Change the red, green, and blue color intensities:

        color(1) = 1.0     ! red
        color(2) = 1.0     ! green
        color(3) = 0.14    ! blue

C Modify the line thickness before plotting the second two histogram
C bars in the third graph:

        CALL LGP$TABLE_MODIFY(LGP$K_HISTLINE_THICKNESS,5.)

C Plot the second two histogram bars for the third graph:

        CALL LGP$HIST(3, 2.5, 4.0, 4.0, 1,,color,2)
        CALL LGP$HIST(3, 4.2, 5.0, 5.0, 1,,color)

C Plot the data points on the fourth graph:

        CALL LGP$POINT(4, 1.5, 3.0, .2, 3,1,,color,2)
        CALL LGP$POINT(4, 3.5, 3.0, .2, 9,,,,1)

C Close the plotting windows for all four graphs and clear the second
C graph from the screen:

        CALL LGP$TERMINATE_PLOT(1)     ! close first plotting window
        CALL LGP$TERMINATE_PLOT(2,1)   ! close second window and clear screen
        CALL LGP$TERMINATE_PLOT(3)     ! close third plotting window
        CALL LGP$TERMINATE_PLOT(4)     ! close fourth plotting window
        STOP ' This is it '
        END
```

Figure 3-2 shows the output produced by Example 3-1:

Figure 3-2: Multiple Plots



Chapter 4
Plotting Routines

This chapter contains the following:

- A table summarizing the plotting routines and their functions
- Reference descriptions of the plotting routines in alphabetical order

4.1 Summary of Plotting Routines

Table 4–1 summarizes the plotting routines in alphabetical order.

Table 4–1: Summary of Plotting Routines

Routine Call	Function
LGP$CHANGE_PLOTTING_LOGICALS	Defines or redefines plotting logical names in a user program
LGP$CONTOUR	Plots a two-dimensional array in contour form
LGP$CONTOUR_SHADE	Plots a two-dimensional array in contour form with the area between contour lines shaded
LGP$CONTOURM	Plots additional contours from a two-dimensional array previously defined by LGP$CONTOUR
LGP$ERASE	Clears or partially clears data from the workstation window
LGP$GET_CHOICE	Allows you to create a menu screen and to select options from the menu using a mouse or the arrow keys

Table 4–1 (Cont.): Summary of Plotting Routines

Routine Call	Function
LGP$GET_LOCATOR	Allows you to position the cursor on a graph and to return the x, y coordinates of that position in data units, using a mouse or the arrow keys
LGP$GET_STRING	Allows you to prompt for a text string
LGP$GET_STROKE	Allows you to select and store a specified number of x, y coordinate points from a graph in data units using a mouse or the arrow keys
LGP$GET_VALUE	Allows you to input a real number to a specified range of numbers using a mouse or the arrow keys
LGP$HIST	Plots a histogram bar
LGP$INFO	Returns plotting status information
LGP$MAP_PEN	Maps a color to a pen number for hardcopy plots to the LVP16 or the HP7550 pen plotters
LGP$PLOT	Establishes plotting arguments and plots an array of data
LGP$PLOT_LOG	Plots an array of data and creates a scaled logarithmic or linear coordinate system for the x axis, the y axis, or both
LGP$PLOT_METAFILE	Plots a previously stored metafile
LGP$PLOT_3D	Produces a three-dimensional plot on an axis system previously created by the LGP$SET_3D_GRAPH routine
LGP$PLOTC	Plots a data set iteratively in a two-dimensional space previously defined by the LGP$PLOT routine or the LGP$SET_GRAPH routine
LGP$PLOTM	Plots an additional array in two-dimensional space previously defined by the LGP$PLOT routine or the LGP$SET_GRAPH routine
LGP$POINT	Plots a data point or an array of data points on a coordinate system previously defined by the LGP$PLOT routine or the LGP$SET_GRAPH routine
LGP$PRINT_SCREEN	Produces a hard copy of the plot displayed on the terminal screen (for VT125, 240, 241, 330, and 340 terminals connected to any LA device)

Table 4–1 (Cont.): Summary of Plotting Routines

Routine Call	Function
LGP$PUT_TEXT	Writes a specified text string to a plot
LGP$SCALE	Accesses the autoscaling algorithm used in a linear plot
LGP$SCALE_LOG	Accesses the autoscaling algorithm used in a logarithmic plot
LGP$SET_GRAPH	Creates an axis system for two-dimensional plotting
LGP$SET_3D_GRAPH	Creates an axis system for a three-dimensional plot to be produced with LGP$PLOT_3D
LGP$SPLINE_QHC	Generates the spline values for smooth curve plotting and accesses the smooth curve-fitting algorithm to interpolate points between given input values
LGP$STNDEV	Plots a standard deviation marker
LGP$3D_SIMPLE	Produces a three-dimensional plot
LGP$TABLE_MODIFY	Modifies the plotting attribute list
LGP$TABLE_READ_VALUE	Returns the current value of a specific plotting attribute from the plotting attribute list
LGP$TABLE_RESET	Resets the plotting attribute list to default values
LGP$TERMINATE_PLOT	Signals the end of plotting information, clears the plotting screen, and closes the open workstation window

4.2 Plotting Routine Descriptions

Each LGP routine is described using the following format:

- The routine name and overview appear at the top of the first page of each reference description.
- **Format** presents the routine entry point name, or routine name, and the routine argument list in the correct syntactical form.
- **Returns** lists the information returned by the routine.
- **Arguments** describes which information the argument passes to the routine or which information the argument returns from the routine, and the data type, access, mechanism, and acceptable values of the argument.
- **Description** contains the following information: includes interaction between routine arguments; interactions or dependencies between the routine and other LGP routines; restrictions for use; and actions specific to the routine when used with certain devices.
- **Condition Values** contains a table of condition values a specific routine generates. See Section 5.3, List of Error Messages, for an explanation of condition values and suggested user action.
- **Graphical Output** provides an example of the output the LGP routine produces.

The reference section that follows describes the LGP routines in alphabetical order and how to use them.

LGP$CHANGE_PLOTTING_LOGICALS

The LGP$CHANGE_PLOTTING_LOGICALS routine defines or redefines plotting logical names in a user program.

Format LGP$CHANGE_PLOTTING_LOGICALS *([wstype], [port], [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

wstype
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the integer used to define the GKS$WSTYPE logical name. The appropriate numbers are presented in Appendix A of the *DEC GKS Reference Manual* and in Table 1-2 in this guide.

These numbers are used to denote the type of output device that is to be used for plotting. This value must be greater than or equal to 0. If the **wstype** argument is omitted, GKS$WSTYPE does not change.

port
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument defining the logical name GKS$CONID. This string value must be less than or equal to 31 characters in length. If the **port** argument is omitted, GKS$CONID does not change.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Use this routine to change the two logical names that govern the type of plotting device and the external location of that device.

The device and its external location are controlled by the logicals GKS$WSTYPE and GKS$CONID. You can change the value of either logical name or both logical names with one call to this routine.

If you are using a VAXstation with multiple windows and you define GKS$CONID before plotting, you must redefine GKS$CONID for each additional window you open.

This routine changes the logical name assignments in your process, but these names are volatile. When your process terminates, these logical names are replaced by the process logical names established before you ran the program containing this routine.

LGP$CHANGE_PLOTTING_LOGICALS is useful when you want to direct DECwindows output to different servers or when you want to produce a plot on a hardcopy device specified during program execution.

NOTE

You must use LGP$CHANGE_PLOTTING_LOGICALS to define logical names when you use DECwindows workstation types 212 or 213. The **port** argument is in the form "display_id_number!window_id_number" for type 212 and "widget_id_number" for type 213.

Examples of this routine are as follows:

```
LGP$CHANGE_PLOTTING_LOGICALS (14, "TT:")
LGP$CHANGE_PLOTTING_LOGICALS (211, "mynodename::0.1")
LGP$CHANGE_PLOTTING_LOGICALS (212, "1806975!531224")
LGP$CHANGE_PLOTTING_LOGICALS (213, "758910")
```

Note that if you are using LGP$CHANGE_PLOTTING_LOGICALS to send different plots to the same DECwindows server, you must use unique port names, such as "mynodename::0.1", "mynodename::0.2", "mynodename::0.3", and so forth.

See Section 1.5, Defining Logical Names, for further information about making logical name assignments.

Condition Values

Symbolic Status	Description
LGP$_ILL_PORT_LEN	length of port string is illegal
LGP$_ILL_WSTYPE	wstype is less then 0
LGP$_SUCCESS	success

LGP$CONTOUR

The LGP$CONTOUR plotting routine produces a two-dimensional contour plot.

Format

LGP$CONTOUR *(ws_number, array3, nx, ny, contrs, [ncons], [xlabel], [ylabel], [status], [iline], [igrid], [xcontrol], [ycontrol], [color], [title], [metaflag], [metafile_name])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

array3
VMS Usage: **floating_point**
type: **f_floating**
access: **read only**
mechanism: **by reference, array reference**
A two-dimensional array specifying the z values to be contoured.

nx
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of columns contained in the **array3** argument. The value of **nx** must be an integer greater than or equal to 3.

ny
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of rows contained in the **array3** argument. The value of **ny** must be an integer greater than or equal to 3.

contrs
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array containing the isopotential values to be contoured.

ncons
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of isopotential values contained in **contrs** array. The default value is 1.

xlabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the x-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

ylabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the y-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

iline
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the type of line to draw. The value of **iline** can be one of the following:

Value	Line Type
1	Solid line
2	Dashed line
3	Dotted line
4	Dashed-dotted line

The default value is 1. Any integer outside the range (1 to 4) generates an error.

igrid
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of grid lines per major increment. The value of **igrid** can be one of the following:

Value	Number of Grid Lines
0	No grid lines
1	One grid line for each number label on the axis
2	Two grid lines for each number label on the axis
	.
	.
	.
100	100 grid lines for each number label on the axis

The default value is 0. Any integer outside the range (0 to 100) generates an error.

xcontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following x-axis parameters:

The first value, **xlen**, specifies the length of the x axis in inches.

The second value, **xlow**, specifies the first x-axis annotation value.

The third value, **xhi**, specifies the last x-axis annotation value. This value is the maximum x-axis value.

The fourth value, **xdelta**, specifies the increment at which numerical values are annotated on the x axis. If this value is 0, the x axis is not annotated.

If the **xcontrol** argument is defaulted, the x axis is 5 inches in length and is not annotated.

ycontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following y-axis parameters:

The first value, **ylen**, specifies the length of the y axis in inches.

The second value, **ylow**, specifies the first y-axis annotation value.

The third value, **yhi**, specifies the last y-axis annotation value. This is the maximum y-axis value.

The fourth value, **ydelta**, specifies the increment at which numerical values are annotated on the y axis. If this value is 0, the y axis is not annotated.

If the **ycontrol** argument is defaulted, the y-axis length is determined by your workstation type. See Section 1.3, Plotting Device Support, for the y-axis default length for your workstation type. If the **ycontrol** argument is defaulted, the y axis is not annotated.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

title
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the graph title up to 72 characters in length. The title is automatically centered above the graph. If the **title** argument is omitted, a title is not printed.

metaflag
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying where to output the plot. The value of **metaflag** can be one of the following:

Value	Output is directed to:
0	Screen or plotter only
1	Metafile only
2	Screen or plotter and a metafile

Any integer outside this range (0 to 2) generates an error. The default is 0.

metafile_name
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the name of the metafile. The default metafile name is LGPMETAFILE.DAT.

Description

The LGP$CONTOUR routine maps a three-dimensional surface into two dimensions by the projection of the isopotential lines resulting in the contour plotting of three-dimensional data.

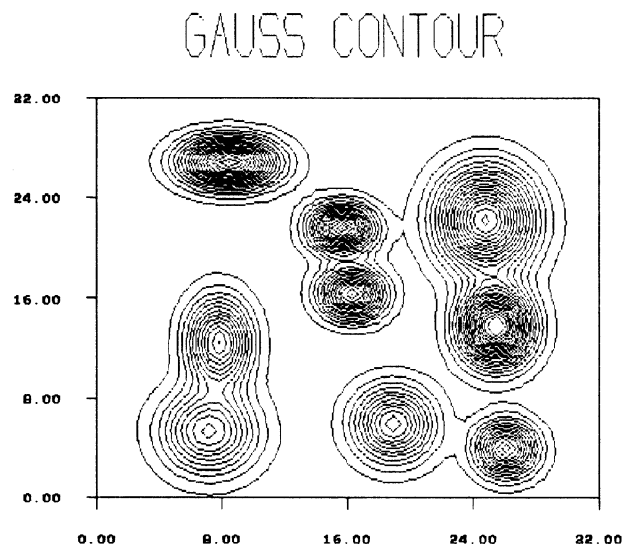
Condition Values

Symbolic Status	Description
LGP$_AXIS_RESIZE	axis too big, resized
LGP$_ILL_COLOR_ARRAY	color entry out of range
LGP$_ILL_IGRID	igrid is out of range
LGP$_ILL_ILINE	iline is not equal to 1, 2, 3, or 4
LGP$_ILL_METAFLAG	metaflag is not 0, 1, or 2
LGP$_ILL_NX	nx argument less than 3
LGP$_ILL_NY	ny argument less than 3
LGP$_ILL_PORT	output port is not recognized
LGP$_ILL_WSN	illegal workstation number
LGP$_ILL_XDELTA	xdelta is out of range
LGP$_ILL_XLEN	xlen is out of range
LGP$_ILL_XLOW	xlow is not less than xhigh
LGP$_ILL_YDELTA	ydelta is out of range
LGP$_ILL_YLEN	ylen is out of range
LGP$_ILL_YLOW	ylow is not less than yhigh
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use

Graphical Output

Figure 4–1 shows the output produced by the LGP$CONTOUR routine:

Figure 4–1: Sample Output of LGP$CONTOUR Routine



LGP$CONTOUR_SHADE

The LGP$CONTOUR_SHADE plotting routine produces a two-dimensional contour plot from a three-dimensional view with the area between the contour lines shaded.

Format

LGP$CONTOUR_SHADE *(ws_number, array3, nx, ny, contrs, [ncons], [xlabel], [ylabel], [status], [shade_array], [igrid], [xcontrol], [ycontrol], [color], [title], [metaflag], [metafile_name])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

array3
VMS Usage: **floating_point**
type: **f_floating**
access: **read only**
mechanism: **by reference, array reference**
A two-dimensional array specifying the z values to be contoured. This argument must be dimensioned to at least 3 X 3.

nx
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of columns contained in the **array3** argument. The value of **nx** must be an integer greater than or equal to 3.

ny
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of rows contained in the **array3** argument. The value of **ny** must be an integer greater than or equal to 3.

contrs
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array containing the isopotential values to be contoured.

ncons
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of isopotential values contained in **contrs** array. The default value is 1.

xlabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the x-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

ylabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the y-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

shade_array
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference, array reference**
An array of length **ncons** containing the fill or shade styles to be used for the contour value in the corresponding position in the **contrs** array. The value of **shade_array** can be any of the following:

Value	Shading
0	Unfilled or outlined
1	Filled
2	Cross-hatched
3	Filled with background color

The default value is 0. Any integer outside the range (0 to 3) generates an error.

igrid
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of grid lines per major increment. The value of **igrid** can be one of the following:

Value	Number of Grid Lines
0	No grid lines
1	One grid line for each number label on the axis
2	Two grid lines for each number label on the axis
	.
	.
	.
100	100 grid lines for each number label on the axis

The default value is 0. Any integer outside the range (0 to 100) generates an error.

xcontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following x-axis parameters:

The first value, **xlen**, specifies the length of the x axis in inches.

The second value, **xlow**, specifies the first x-axis annotation value.

The third value, **xhi**, specifies the last x-axis annotation value. This value is the maximum x-axis value.

The fourth value, **xdelta**, specifies the increment at which numerical values are annotated on the x axis. If this value is 0, the x axis is not annotated.

If the **xcontrol** argument is defaulted, the x axis is 5 inches in length and is not annotated.

ycontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following y-axis parameters:

The first value, **ylen**, specifies the length of the y axis in inches.

The second value, **ylow**, specifies the first y-axis annotation value.

The third value, **yhi**, specifies the last y-axis annotation value. This is the maximum y-axis value.

The fourth value, **ydelta**, specifies the increment at which numerical values are annotated on the y axis. If this value is 0, the y axis is not annotated.

If the **ycontrol** argument is defaulted, the y-axis length is determined by your workstation type. See Section 1.3, Plotting Device Support, for the y-axis default length for your workstation type. If the **ycontrol** argument is defaulted, the y axis is not annotated.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
A two dimensional array of (**ncons**, 3) format. Each row in the array contains the red, green, and blue intensities corresponding to the contour value in the same row in the **ncons** array. For example, the first row in the color array is used to fill the area between the contour lines generated by the first and second entries in the **ncons** array.

title
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the graph title up to 72 characters in length. The title is automatically centered above the graph. If the **title** argument is omitted, a title is not printed.

metaflag
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying where to output the plot. The value of **metaflag** can be one of the following:

Value	Output is directed to:
0	Screen or plotter only
1	Metafile only
2	Screen or plotter and a metafile

Any integer outside this range (0 to 2) generates an error. The default is 0.

metafile_name
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the name of the metafile. The default metafile name is LGPMETAFILE.DAT.

Description

Use this routine to map a three-dimensional surface into two dimensions by the projection of the isopotential lines resulting in the contour plotting of three-dimensional data with the area between the contour lines shaded. Equispaced data points (from the x and y axis) are mapped to a two-dimensional projection according to the values specified by the argument list.

This routine creates its own axis system according to the values you specify in the argument list. You do not need to call LGP$CONTOUR before you call LGP$CONTOUR_SHADE.

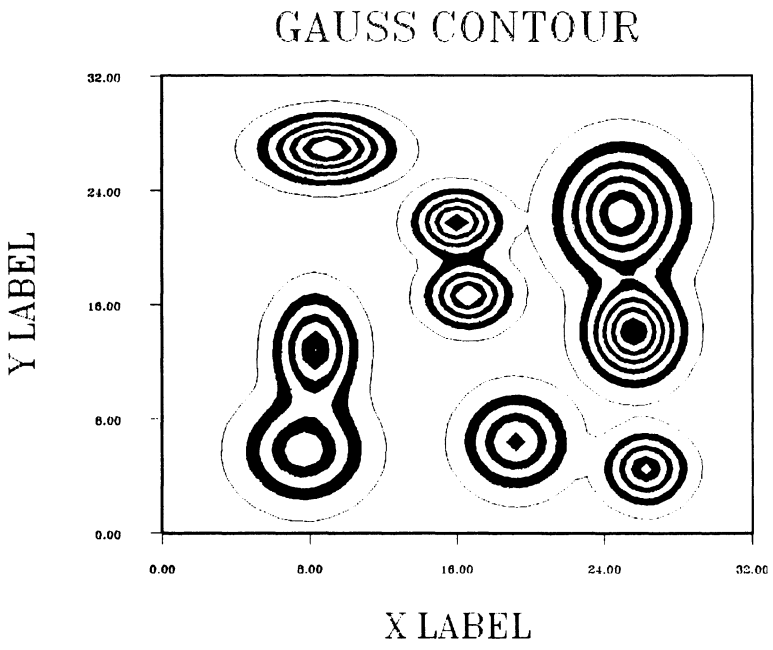
Condition Values

Symbolic Status	Description
LGP$_AXIS_RESIZED	Axis too big, resized
LGP$_ILL_COLOR_ARRAY	color entry out of range
LGP$_ILL_IGRID	igrid is out of range
LGP$_ILL_METAFLAG	metaflag was not 0, 1, or 2
LGP$_ILL_NX	nx argument less than 3
LGP$_ILL_NY	ny argument less than 3
LGP$_ILL_PORT	output port is not recognized
LGP$_ILL_WSN	illegal workstation number
LGP$_ILL_XDELTA	xdelta is out of range
LGP$_ILL_XLEN	xlen is out of range
LGP$_ILL_XLOW	xlow is not less than xhigh
LGP$_ILL_YDELTA	ydelta is out of range
LGP$_ILL_YLEN	ylen is out of range
LGP$_ILL_YLOW	ylow is not less than yhigh
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use

Graphical Output

Figure 4–2 shows the output produced by the LGP$CONTOUR_SHADE routine:

Figure 4–2: Sample Output of LGP$CONTOUR_SHADE Routine



LGP$CONTOURM

The LGP$CONTOURM plotting routine produces additional two-dimensional contours from a two-dimensional array. The LGP$CONTOUR routine must be called first. LGP$CONTOURM is useful when varying contour plot color and line type.

Format

LGP$CONTOURM *(ws_number, array3, nx, ny, contrs, [ncons], [status], [iline], [color])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

array3
VMS Usage: **floating_point**
type: **f_floating**
access: **read only**
mechanism: **by reference, array reference**
A two-dimensional array specifying the z values to be contoured. This array must be dimensioned to at least 3 X 3.

nx
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of columns contained in the **array3** argument. The value of **nx** must be an integer greater than or equal to 3.

ny
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of rows contained in the **array3** argument. The value of **ny** must be an integer greater than or equal to 3.

contrs
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array containing the isopotential values to be contoured.

ncons
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of isopotential values contained in **contrs** array. The default value is 1.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

iline
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the type of line to draw. The value of **iline** can be one of the following:

Value	Line Type
1	Solid line
2	Dashed line
3	Dotted line
4	Dashed-dotted line

The default value is 1. Any integer outside the range (1 to 4) generates an error.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information on producing multicolor plots.

Description

Use this routine in conjunction with LGP$CONTOUR to map an additional three-dimensional surface into two dimensions by the projection of the isopotential lines resulting in the contour plotting of three-dimensional data. The three-dimensional surface and axis system are previously established by the LGP$CONTOUR routine.

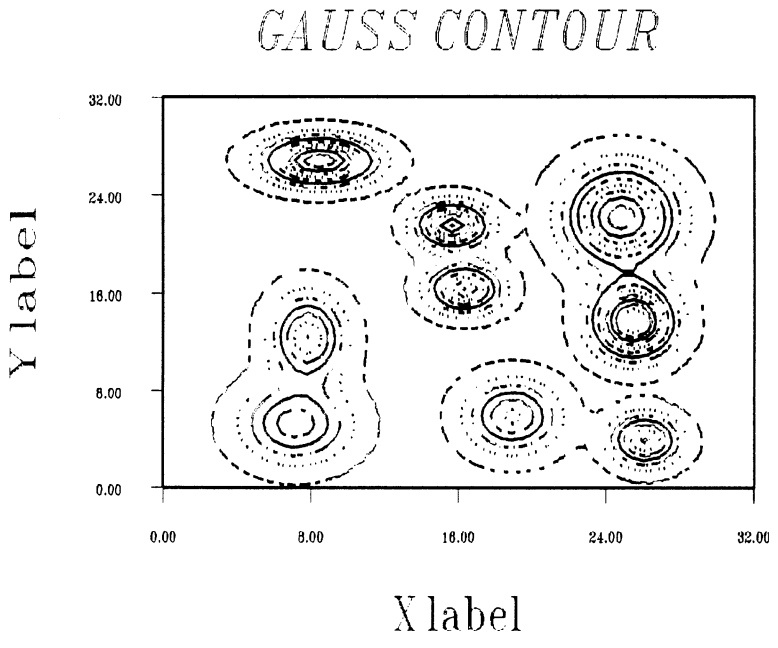
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry out of range
LGP$_ILL_ILINE	iline is not equal to 1, 2, 3, or 4
LGP$_ILL_NX	nx argument less than 3
LGP$_ILL_NY	ny argument less than 3
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

Graphical Output

Figure 4-3 shows the output produced by the LGP$CONTOURM routine after a call to LGP$CONTOUR:

Figure 4-3: Sample Output of LGP$CONTOURM Routine



LGP$ERASE

The LGP$ERASE routine specifies the x, y coordinates of a polygon to be erased from a graph. This routine clears or partially clears data from the workstation window. If no points are specified, data inside the window is erased.

Format

LGP$ERASE *(ws_number, [npts], [xcoordinate], [ycoordinate], [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

npts
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of end points in the polygon to be erased. The area enclosed by the polygon will be erased. Since a polygon must have at least three end points, this argument must be greater than or equal to 3. If this argument is defaulted, all data within

the graph is erased. The coordinate system and labels outside the graph are not erased.

xcoordinate
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array specifying the x coordinates of the end points of the polygon within the workstation window to be erased. The LGP$ERASE routine ignores this array if the **npts** argument is defaulted.

ycoordinate
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array specifying the y coordinates of the end points of the polygon within the workstation window to be erased. The LGP$ERASE routine ignores this array if the **npts** argument is defaulted.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

The LGP$ERASE routine allows you to erase a specified area in a workstation window. The area is defined by specifying its end points in the **xcoordinate** and **ycoordinate** arrays. The x coordinate of the ith end point is contained in the ith element of the **xcoordinate** array. The y coordinate of that point is contained in the ith element of the **ycoordinate** array. Coordinates for points are determined by their distance in inches from the bottom left corner of the window.

For example, suppose you want to erase the bottom right quarter of a workstation window with a window that is 6.0 inches high and 10.0 inches wide. The bottom right quarter of this window is contained within the polygon created by the end points (5.0, 0.0), (5.0, 3.0), (10.0, 0.0) and (10.0, 3.0). The area to be erased is defined by these four end points, so **npts** equals 4. The **xcoordinate** array would contain the x coordinates of these points (5.0, 5.0, 10.0, 10.0) and the **ycoordinate** array would contain the y coordinates of the points (0.0, 3.0, 0.0, 3.0).

If you specify only the **ws_number** argument and default all other arguments, your call to the LGP$ERASE routine clears all data from the graph. The coordinate system and labels outside the graph remain.

To erase the entire window (graph, coordinate system, and labels) without closing the workstation window, call LGP$ERASE specifying the four corners of the window as the end points of the polygon to be erased. Use LGP$TERMINATE_PLOT if you want to erase the window and close it.

Condition Values

Symbolic Status	Description
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_NOT_IN_USE	workstation number is not in use

LGP$GET_CHOICE

The LGP$GET_CHOICE routine allows you to create a menu screen and make a selection from that menu using a mouse or the arrow keys.

Format

LGP$GET_CHOICE *(ws_number, choice_array, choice_array_size, echo_area, [status], choice)*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

choice_array
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An array of character strings containing the text for each of the menu choices.

choice_array_size
VMS Usage: **longword_signed**
type: **longword**
access: **read only**
mechanism: **by reference**
An argument specifying the number of entries in the **choice_array** argument.

echo_area
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four controlling the size and the position of the menu with the following x, y coordinates:

The first value, **x_low**, specifies the first x-coordinate value in inches.

The second value, **x_hi**, specifies the last x-coordinate value in inches.

The third value, **y_low**, specifies the first y-coordinate value in inches.

The fourth value, **y_hi**, specifies the last y-coordinate value in inches.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

choice
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **write only**
mechanism: **by reference**
An argument returning the selected menu entry.

LGP$GET_CHOICE

Description

The LGP$GET_CHOICE routine allows you to display a menu on the screen and to select an entry from the menu. On a VAXstation or VT300-series terminal, use the mouse to highlight menu entries. Press the left mouse button to select an entry. On a VT125, VT240, or VT241 terminal, use the arrow keys to highlight menu entries. Press Return to select an entry.

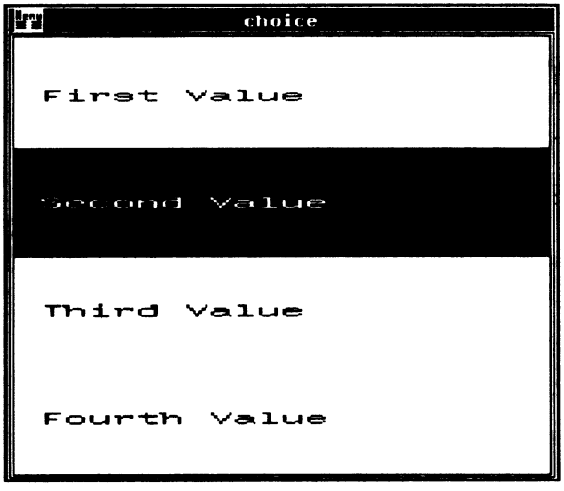
Condition Values

Symbolic Status	Description
LGP$_ILL_ARRAY_SIZE	choice array size is negative or 0
LGP$_ILL_ECHO_AREA	invalid echo area specified
LGP$_ILL_INPUT_WS	workstation does not allow input
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_NOT_IN_USE	workstation number is not in use

Graphical Output

Figure 4-4 shows the output produced by the LGP$GET_CHOICE routine:

Figure 4-4: Sample Output of LGP$GET_CHOICE Routine



LGP$GET_LOCATOR

The LGP$GET_LOCATOR routine allows you to position the cursor on a graph and return the x, y coordinate values of that position in data units.

Format

LGP$GET_LOCATOR *(ws_number, initial_x, initial_y, [status], x_pos, y_pos)*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number

VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**

An argument specifying the plotting workstation number. By specifying the workstation number you can select the workstation from which the x, y coordinate value is returned.

initial_x

VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**

An argument specifying the initial x position of the cursor in inches.

initial_y
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the initial y position of the cursor in inches.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

x_pos
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference**
An argument returning the x coordinate of the position of the cursor.

y_pos
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference**
An argument returning the y coordinate of the position of the cursor.

Description

Use the LGP$GET_LOCATOR routine to select and return the x, y coordinate values of a point on an existing graph in data units. At a later time, you can use this value in a program.

On a VAXstation or a VT300-series terminal, use the mouse to position the cursor. To select a point, press the left mouse button. On a VT125, VT240, or VT241 terminal, use the arrow keys to position the cursor. To select a point, press Return.

Condition Values

Symbolic Status	Description
LGP$_ILL_INPUT_WS	workstation does not allow input
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_NOT_IN_USE	workstation number is not in use

LGP$GET_STRING

The LGP$GET_STRING routine allows you to prompt for a text string.

Format

LGP$GET_STRING *(ws_number, prompt, echo_area, [status], string, string_size)*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

prompt
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the text prompt up to 72 characters in length.

echo_area
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four controlling the size and the position of the prompt screen with the following x, y coordinates:

The first value, **x_low**, specifies the first x-coordinate value in inches.

The second value, **x_hi**, specifies the last x-coordinate value in inches.

The third value, **y_low**, specifies the first y-coordinate value in inches.

The fourth value, **y_hi**, specifies the last y-coordinate value in inches.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

string
VMS Usage: **char_string**
type: **character string**
access: **write only**
mechanism: **by descriptor**
An argument returning the text input prompted by the **prompt** argument.

string_size
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **write only**
mechanism: **by reference.**
An argument returning the number of characters returned in the **string** argument.

Description

The LGP$GET_STRING routine prompts you to enter a text string. For example, you could use LGP$GET_STRING to enter a label for an existing graph.

To use the LGP$GET_STRING routine:

1. Write a program including a call to the LGP$GET_STRING routine. Supply the parameters to create a string prompt. When you run the program, it prompts you to enter a string.
2. Enter the string.
3. Press Return to return the string.

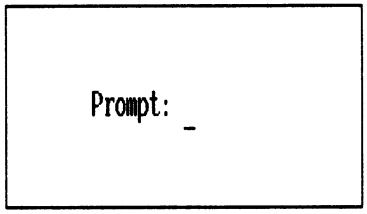
Condition Values

Symbolic Status	Description
LGP$_ILL_ECHO_AREA	invalid echo area specified
LGP_ILL_INPUT_WS	workstation does not allow input
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_NOT_IN_USE	workstation number is not in use

Graphical Output

Figure 4-5 shows the output produced by the LGP$GET_STRING routine:

Figure 4-5: Sample Output of LGP$GET_STRING Routine



LGP$GET_STROKE

The LGP$GET_STROKE routine allows you to position the cursor on a specified number of points on a graph and return the x and y coordinates of the selected points.

Format

LGP$GET_STROKE *(ws_number, num_points, [status], x_values, y_values)*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the workstation number, you can indicate the workstation from which the coordinate points are to be collected.

num_points
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of points you are selecting.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

x_values
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference, array reference**
An array returning the x coordinates of the selected points.

y_values
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference, array reference**
An array returning the y coordinates of the selected points.

Description

Using the LGP$GET_STROKE routine, you can position the cursor on a specified number of points on a graph and return the x and y coordinates of the selected points. If you are specifying only one point, you can also use LGP$GET_LOCATOR.

Use this routine to save a series of points from one plot for use in another plot. For example, you can use the LGP$GET_STROKE routine to save a series of points defining a triangle in an existing plot, and later include those saved points in another plot using the LGP$PLOTM routine.

On a VT300-series terminal, use the mouse to position the cursor. To select a point, press the space bar. To delete a selected point, press the delete key. To end selection of points, press the left mouse button.

On a VT125, VT240, or VT241 terminal, use the arrow keys to position the cursor. To select a point, press the space bar. To delete a selected point, press the delete key. To end selection of points, press Return.

On a VAXstation, point collection starts when the mouse is moved. To end selection of points, press the left mouse button. For precise point collection on a VAXstation, use multiple LGP$GET_LOCATOR calls instead of LGP$GET_STROKE.

Condition Values

Symbolic Status	Description
LGP$_ILL_INPUT_WS	workstation does not allow input
LGP$_ILL_NUM_POINT	number of points is negative or 0
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_NOT_IN_USE	workstation number is not in use

LGP$GET_VALUE

The LGP$GET_VALUE routine allows you to input a real number within a specified range of values.

Format LGP$GET_VALUE *(ws_number, range, initial_value, echo_area, [status], ret_value)*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

range
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length two defining the range of possible input values. The first entry specifies the lowest value in the range, and the second entry specifies the highest value in the range.

initial_value
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the initial position of the valuator.

echo_area
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four controlling the position of the valuator with the following x, y coordinates:

The first value, **x_low**, specifies the first x-coordinate value in inches.

The second value, **x_hi**, specifies the last x-coordinate value in inches.

The third value, **y_low**, specifies the first y-coordinate value in inches.

The fourth value, **y_hi**, specifies the last y-coordinate value in inches.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

ret_value
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference**
An argument returning the value input to the range of values.

LGP$GET_VALUE

Description

The LGP$GET_VALUE routine allows you to input a real number within a range of values. A valuator is displayed showing the specified range of numbers. Use the slider to select a value.

On a VAXstation or a VT300-series terminal, use the mouse to position the slider. To select a value, press the left mouse button. On a VT125, VT240, or VT241 terminal, use the arrow keys to position the slider. To select a value, press Return.

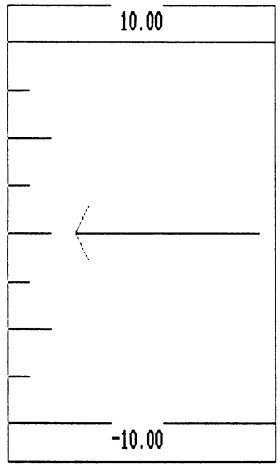
Condition Values

Symbolic Status	Description
LGP$_ILL_ECHO_AREA	invalid echo area specified
LGP$_ILL_INIT_VALUE	initial value is out of range
LGP$_ILL_INPUT_WS	workstation does not allow input
LGP$_ILL_RANGE	invalid range specified
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

Graphical Output

Figure 4–6 shows the output produced by the LGP$GET_VALUE routine:

Figure 4–6: Sample Output of LGP$GET_VALUE Routine



LGP$HIST

The LGP$HIST routine is a two-dimensional plotting routine that plots a histogram bar.

Format

LGP$HIST *(ws_number, xlow, xhi, y, [npts], [status], [color], [ishade])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

xlow
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the leftmost x-axis position of a histogram element. Many histogram elements can be plotted with one call by specifying the **npts** argument. If you specify the **npts** argument larger than 1, the **xlow** argument is an array.

xhi
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the rightmost x-axis position of a histogram element. Many histogram elements can be plotted with one call by specifying the **npts** argument. If you specify the **npts** argument larger than 1, the **xhi** argument is an array.

y
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the y-axis position of a histogram element. Many histogram elements can be plotted with one call by specifying the **npts** argument. If you specify the **npts** argument larger than 1, the **y** argument is an array.

npts
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of histogram elements to plot. The default is 1. Any integer less than 1 generates an error.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information producing multicolor plots.

ishade
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying how to shade the histogram. The value of **ishade** can be one of the following:

Value	Interior Shading
0	Interior unfilled
1	Interior filled
2	Interior cross-hatched
3	Interior filled with background color

The default is 0. Any integer outside the range (0 to 3) generates an error.

Description

Use the LGP$HIST plotting routine to produce a histogram or bar chart. The axis system is set up by calling LGP$PLOT or LGP$SET_GRAPH with no points plotted.

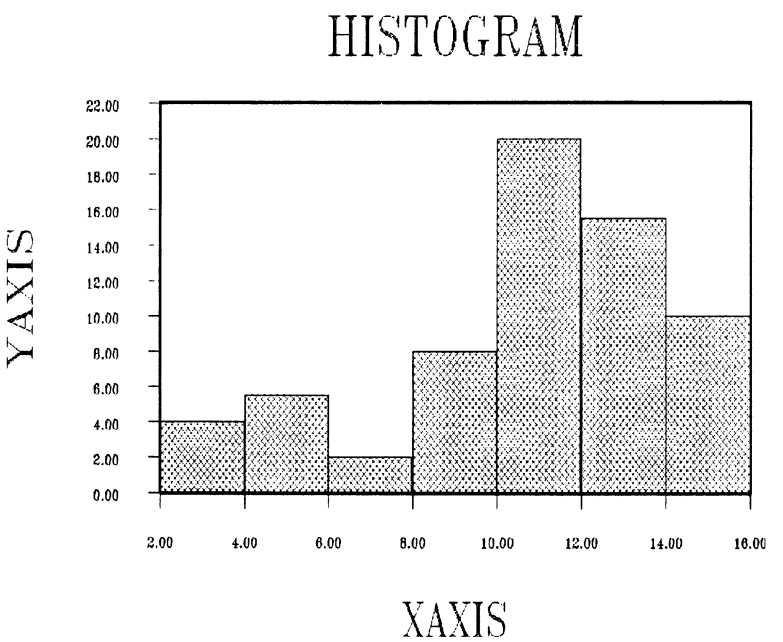
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry out of range
LGP$_ILL_NPTS	npts less than or equal to zero
LGP$_ILL_PROC_HIST	improper calling order
LGP$_ILL_SHADE	ishade is out of range
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use
LGP$_WSN_NOT_IN_USE	workstation number is not in use

Graphical Output

Figure 4–7 shows the output produced by the LGP$HIST routine:

Figure 4–7: Sample Output of LGP$HIST Routine



LGP$INFO

The LGP$INFO routine returns information about opened workstations and the workstation most recently used.

Format

LGP$INFO *(mrwn, [nopen], [wslist], [wslist_size], [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

mrwn
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **write only**
mechanism: **by reference**
An argument returning the most recently accessed workstation number. If no workstations are open, a 0 is returned.

nopen
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **write only**
mechanism: **by reference**
An argument returning the number of open workstations. If no workstations are open, a 0 is returned.

wslist
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **write only**
mechanism: **by reference, array reference**
An array returning the workstation numbers of the open workstations.

wslist_size
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the size of the **wslist** array.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

The routine LGP$INFO returns the workstation number most recently used and additional information, such as the number of opened workstations and their workstation numbers.

Condition Values

Symbolic Status	Description
LGP$_ILL_WLIST_SIZE	wlist_size is less than or equal to zero
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_NOT_IN_USE	workstation number not in use

LGP$MAP_PEN

The LGP$MAP_PEN routine maps a color to a pen number for hardcopy plots to the LVP16 or the HP7550 pen plotters.

Format LGP$MAP_PEN *(color, pnum, [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

pnum
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the pen number. This number can be between 1 and 16, inclusive. Any integer outside the range generates an error.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

The LVP16 and HP7550 pen plotters do not respond to a continuum of colors, but rather to exact colors mapped to pen numbers. This routine allows you to map the colors wanted for a plot to the associated pen numbers.

The LVP16 supports six pens. If you specify a value higher than 6 as the value of the **pnum** argument, the LVP16 uses the default pen (pen number 1).

The HP7550 supports eight pens. If you specify a value higher than 8 as the value of the **pnum** argument, the HP7550 uses the default pen (pen number 1).

Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_PEN_NUMBER	illegal pen number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

LGP$PLOT

The LGP$PLOT plotting routine creates an axis system and plots one array of data.

Format

LGP$PLOT *(ws_number, mode_string, [xarray], yarray, n, [xlabel], [ylabel], [status], [iline], [igrid], [xcontrol], [ycontrol], [color], [title], [metaflag], [metafile_name])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

mode_string
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
A character string containing four characters of value IXSY, IXMY, EXSY, or EXMY. The first letter of the string is either **I** (implicit) or **E** (explicit), specifying an implicit or explicit x array. The second letter is always **X** (x array). The third letter is either **S** (single) or **M** (multiple), specifying a one-dimensional or a multidimensional array. The fourth letter is always **Y**.

The character string is not case sensitive. You can specify the character string in either uppercase or lowercase letters.

xarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifing the x-axis values to be plotted.

If **mode_string** specifies an explicit x array (EXSY or EXMY), the **xarray** argument contains **npts** data values in the x array. **If an explicit x array is specified, the xarray argument is required.**

If **mode_string** specifies an implicit x array (IXSY or IXMY), the single value in the **xarray** argument is equal to the x value between data points, and the first x value of data is assumed to be 0. **If an implicit x array is specified, the xarray argument is optional.**

When **mode_string** specifies an implicit x array, the plotting of data across the x axis and the x-axis annotation vary according to the following argument specifications:

- If you default both the **xarray** argument and the **xcontrol** argument, the data is plotted at equal increments across the x axis. The x axis is not annotated.
- If you default the **xarray** argument and you specify the **xcontrol** argument, the data is plotted at equal increments across the x axis independently of the x-axis annotation.

- If you specify the **xarray** argument and you default the **xcontrol** argument, the data is plotted at equal increments across the x axis. The x-axis annotation is determined by the autoscaling algorithm.
- If you specify both the **xarray** argument and the **xcontrol** argument, the data is plotted beginning with x = 0, with the spacing determined by the single value contained in the **xarray** argument. The x-axis annotation is determined by the values contained in the **xcontrol** argument.

yarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the independent y-axis values. The array can be either one-dimensional (IXSY, EXSY), or multidimensional (IXMY or EXMY).

n
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the following three values: **npts**, **nsize**, and **nsets**. If a one-dimensional array is specified (IXSY, EXSY), then **n** should be a single longword containing **npts**. If a two-dimensional array is specified (IXMY, EXMY), then you must specify both **nsize** and **nsets**.

The first entry, **npts**, specifies the number of data values in each line segment. The value must be greater than or equal to 0. No plotting action is taken if **npts** equals 0 or 1.

The second entry, **nsize**, specifies the total size of the array that you declared to hold the data values. Specify the total size of the declared array whether or not you are using all of the points in the array. For example, if you declared an array of size 100, but are plotting only the first 50 points stored in that array, **npts** is equal to 50, but **nsize** must be specified as 100. The value of **nsize** must be greater than or equal to **npts**.

The third entry, **nsets**, specifies the number of data sets to be drawn. Each set is one line or curve or other element on the graph. The value must be greater than or equal to 1.

xlabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the x-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

ylabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the y-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

iline
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the type of line to draw. The value of **iline** can be one of the following.

Value	Line Type
1	Solid line
2	Dashed line
3	Dotted line
4	Dashed-dotted line

The default value is 1. Any integer outside the range (1 to 4) generates an error.

igrid
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of grid lines per major increment. The value of **igrid** can be one of the following:

Value	Number of Grid Lines
0	No grid lines
1	One grid line for each number label on the axis
2	Two grid lines for each number label on the axis
	.
	.
	.
100	100 grid lines for each number label on the axis

The default value is 0. Any integer outside the range (0 to 100) generates an error.

xcontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following x-axis parameters:

The first value, **xlen**, specifies the length of the x axis in inches.

The second value, **xlow**, specifies the first x-axis annotation value. This value is the minimum x-axis value.

The third value, **xhi**, specifies the last x-axis annotation value. This value is the maximum x-axis value.

The fourth value, **xdelta**, specifies the increment at which numerical values are annotated on the x axis. If this value is 0, the x axis is not annotated.

If the **xcontrol** argument is defaulted, the x axis is 5 inches in length and is not annotated.

Note that **xcontrol** specifies the size of the graph and its associated annotation values. It does not prevent data from being plotted outside these limits.

ycontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following y-axis parameters:

The first value, **ylen**, specifies the length of the y axis in inches.

The second value, **ylow**, specifies the first y-axis annotation value. This is the minimum y-axis value.

The third value, **yhi**, specifies the last y-axis annotation value. This is the maximum y-axis value.

The fourth value, **ydelta**, specifies the increment at which numerical values are annotated on the y axis. If this value is 0, the y axis is not annotated.

If the **ycontrol** argument is defaulted, the y-axis length is determined by your workstation type. See Section 1.3, Plotting Device Support, for the y-axis default length for your workstation type. If the **ycontrol** argument is defaulted, the y axis is not annotated.

Note that **ycontrol** specifies the size of the graph and its associated annotation values. It does not prevent data from being plotted outside these limits.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

title
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the graph title up to 72 characters in length. The title is automatically centered above the graph. If the **title** argument is omitted, a title is not printed.

metaflag
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying where to output the plot. The value of **metaflag** can be one of the following:

Value	Output is directed to:
0	Screen or plotter only
1	Metafile only
2	Screen or plotter and a metafile

Any integer outside this range (0 to 2) generates an error. The default is 0.

metafile_name
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the name of the metafile. The default metafile name is LGPMETAFILE.DAT.

Description

Use this routine to create the x- and y-axis system. In a program, this routine must precede the other plotting routines that do not create an axis system themselves. This routine also plots one data set.

Condition Values

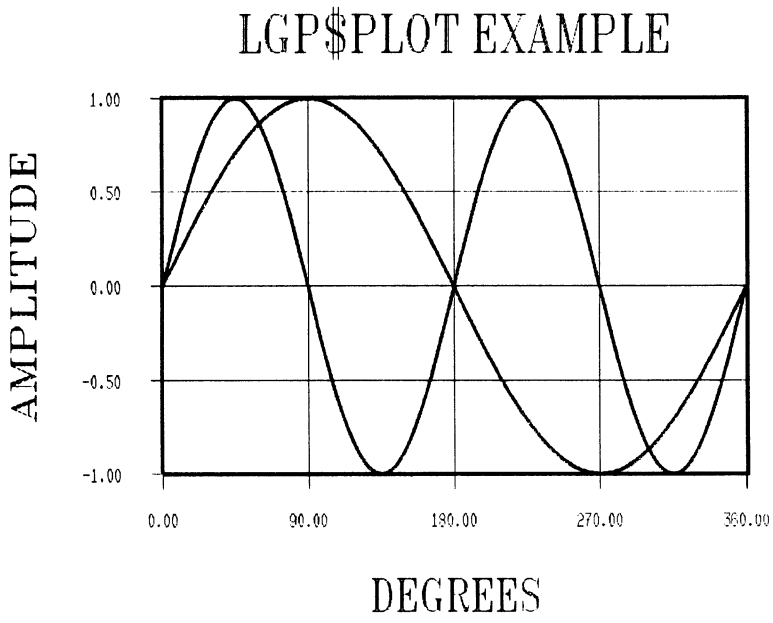
Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_IGRID	igrid is out of range
LGP$_ILL_ILINE	iline is not 1, 2, 3, or 4
LGP$_ILL_METAFLAG	metaflag is not 0, 1, or 2
LGP$_ILL_MODE_STRING	illegal mode_string argument
LGP$_ILL_NPTS	npts is less than or equal to 0
LGP$_ILL_NROW	nrow is less than 1
LGP$_ILL_PORT	output port is not recognized
LGP$_ILL_WSN	illegal workstation number
LGP$_ILL_XARRAY	illegal first value of xarray
LGP$_ILL_XDELTA	xdelta is out of range
LGP$_ILL_XLEN	xlen is out of range
LGP$_ILL_XLOW	xlow is not less than xhigh
LGP$_ILL_YDELTA	ydelta is out of range
LGP$_ILL_YLEN	ylen is out of range
LGP$_ILL_YLOW	ylow is not less than yhigh

Symbolic Status	Description
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use

Graphical Output

Figure 4–8 shows the output produced by the LGP$PLOT routine:

Figure 4–8: Sample Output of LGP$PLOT Routine



LGP$PLOT_LOG

The LGP$PLOT_LOG plotting routine plots an array of data and creates a scaled logarithmic or linear coordinate system in base 10 log for the x axis, the y axis, or both.

Format

LGP$PLOT_LOG *(ws_number, mode, [xarray], yarray, n, [xlabel], [ylabel], [status], [iline], [grid_control], [xcontrol], [ycontrol], [color], [title], [metaflag], [metafile_name])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

mode
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying a scaled logarithmic or linear coordinate system for either the x axis, the y axis, or both.

Use the following values to specify the four possible scaling conditions:

Value	Scaling Condition
0	linear x axis and linear y axis
1	logarithmic x axis and linear y axis
2	linear x axis and logarithmic y axis
3	logarithmic x axis and logarithmic y axis

xarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the x-axis values to be plotted.

- If you default both the **xarray** argument and the **xcontrol** argument, the data is plotted at equal increments across the x axis. The x axis is not annotated.
- If you default the **xarray** argument, and you specify the **xcontrol** argument, the data is plotted at equal increments across the x axis independently of the x-axis annotation.
- If you specify the **xarray** argument and you default the **xcontrol** argument, the data is plotted at equal increments across the x axis. The x-axis annotation is linear or logarithmic, depending on the **mode** argument value you select.
- If you specify both the **xarray** argument and the **xcontrol** argument, the data is plotted beginning with the **xcontrol** start position, with the spacing determined by the single value contained in the **xarray** argument. The x-axis annotation is determined by the values contained in the **xcontrol** argument.

yarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the y-axis values.

n
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the following three values: **npts**, **nsize**, and **nsets**. You must specify all three values, even when the **yarray** argument is one-dimensional.

The first entry, **npts**, specifies the number of data values in each line segment. The value must be greater than or equal to 0. No plotting action is taken if **npts** equals 0 or 1.

The second entry, **nsize**, specifies the total size of the array that you declared to hold the data values. Specify the total size of the declared array whether or not you are using all of the points in the array. For example, if you declared an array of size 100, but are plotting only the first 50 points stored in that array, **npts** is equal to 50, but **nsize** must be specified as 100. The value of **nsize** must be greater than or equal to **npts**.

The third entry, **nsets**, specifies the number of data sets to be drawn. Each set is one line or curve or other element on the graph. The value must be greater than or equal to 1.

xlabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the x-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

ylabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the y-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

iline
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the type of line to draw. The value of **iline** can be one of the following:

Value	Line Type
1	Solid line
2	Dashed line
3	Dotted line
4	Dashed-dotted line

The default value is 1. Any integer outside the range (1 to 4) generates an error.

grid_control
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference, array reference**
An array of length two. The first value of **grid_control** specifies the number of grid lines per major increment on the x axis. The second value of **grid_control** specifies the number of grid lines per major increment on the y axis. The value of **grid_control** for the x axis and the y axis can be one of the following:

Value	Number of Grid Lines
0	No grid lines
1	One grid line for each number label on the axis
2	Two grid lines for each number label on the axis
	.
	.
	.
100	100 grid lines for each number label on the axis

The default value is 0. Any number outside the range (0 to 100) generates an error.

xcontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following x-axis parameters:

The first value, **xlen**, specifies the length of the x axis in inches.

The second value, **xlow**, specifies the first x-axis annotation value. This value is the minimum x-axis value.

The third value, **xhi**, specifies the last x-axis annotation value. This value is the maximum x-axis value.

The fourth value, **xdelta**, specifies the increment at which numerical values are annotated on the x axis. If this value is 0, the x axis is not annotated. For logarithmic scaling, the **xdelta** value is multiplied by itself to produce each successive value on the x axis.

If the **xcontrol** argument is defaulted and you specify the **xarray** argument, the x-axis annotation is determined by the logarithmic scaling algorithm.

In the case of linear scaling with the LGP$PLOT routine or the LGP$PLOTM routine, **xdelta** and **ydelta** are constants added to each successive annotation value to generate the next annotation value. For example, if **xlow** is 3 and **xdelta** is 2, the following linear-axis annotation is generated:

```
3, 5, 7, 9, ...
```

In the case of logarithmic scaling with the LGP$PLOT_LOG routine, **xdelta** and **ydelta** are constants that multiply each successive annotation value to generate the next annotation value. For example, if **xlow** is 1 and **xdelta** is 10, the following log 10 logarithmic-axis annotation is generated:

```
1, 10, 100, 1000, ...
```

For an additional example of logarithmic scaling, if **xlow** is 5 and **xdelta** is 5, the following logarithmic-axis annotation is generated:

```
5, 25, 125, 625, ...
```

Note that **xcontrol** specifies the size of the graph and its associated annotation values. It does not prevent data from being plotted outside these limits.

ycontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following y-axis parameters:

The first value, **ylen**, specifies the length of the y axis in inches.

The second value, **ylow**, specifies the first y-axis annotation value. This value is the minimum y-axis value.

The third value, **yhi**, specifies the last y-axis annotation value. This value is the maximum y-axis value.

The fourth value, **ydelta**, specifies the increment at which numerical values are annotated on the y axis. If this value is 0, the y axis is not annotated.

If the **ycontrol** argument is defaulted, the y-axis annotation is determined by the logarithmic autoscaling algorithm, and the y-axis length is determined by your workstation type. See Section 1.3, Plotting Device Support, for the y-axis default length for your workstation type.

Note that **ycontrol** specifies the size of the graph and its associated annotation values. It does not prevent data from being plotted outside these limits.

See the **xcontrol** argument description for information on the **xdelta** and **ydelta** values and scaling.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

title
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the graph title up to 72 characters in length. The title is automatically centered above the graph. If the **title** argument is omitted, a title is not printed.

metaflag
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying where to output the plot. The value of **metaflag** can be one of the following:

Value	Output is directed to:
0	Screen or plotter only
1	Metafile only
2	Screen or plotter and a metafile

Any integer outside this range (0 to 2) generates an error. The default is 0.

metafile_name
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the name of the metafile. The default metafile name is LGPMETAFILE.DAT.

Description

Use the LGP$PLOT_LOG routine to create a scaled logarithmic or linear coordinate system for the x axis, the y axis, or both. In a program, this routine must precede the other plotting routines that do not create an axis system. This routine also plots one data set.

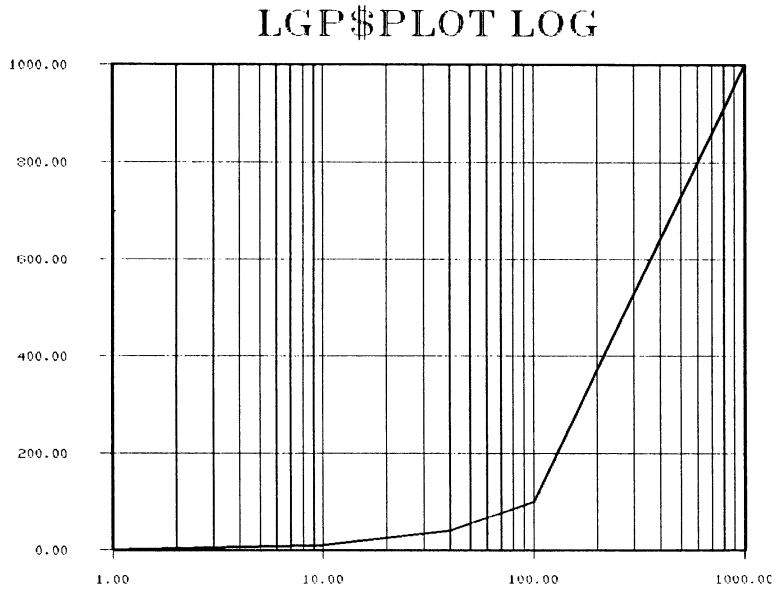
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_IGRID	igrid is out of range
LGP$_ILL_ILINE	iline is not 1, 2, 3, or 4
LGP$_ILL_LOG_DATA	Data in array is less than or equal to 0
LGP$_ILL_LOG_XLOW	xlow is not less than xhigh or less than or equal to 0
LGP$_ILL_LOG_YLOW	ylow is not less than yhigh or less than or equal to 0
LGP$_ILL_METAFLAG	metaflag is not 0, 1, or 2
LGP$_ILL_NPTS	npts is less than or equal to 0
LGP$_ILL_NROW	nrow is less than 1
LGP$_ILL_PORT	output port is not recognized
LGP$_ILL_WSN	illegal workstation number
LGP$_ILL_XARRAY	illegal first value for xarray
LGP$_ILL_XDELTA	xdelta is out of range
LGP$_ILL_XLEN	xlen is out of range
LGP$_ILL_YDELTA	ydelta is out of range
LGP$_ILL_YLEN	ylen is out of range
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use

Graphical Output

Figure 4–9 shows the output produced by the LGP$PLOT_LOG routine:

Figure 4–9: Sample Output of LGP$PLOT_LOG Routine



LGP$PLOT_METAFILE

The LGP$PLOT_METAFILE plotting routine creates a plot from data previously stored in a metafile.

Format LGP$PLOT_METAFILE *(ws_number, filename, [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

filename
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the name of the metafile to plot, up to 72 characters in length.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Use this routine to produce a plot from a previously stored metafile. The metafile contains the x and y axes as well as the plotted data.

You do not need to call the LGP$PLOT routine before you use this routine.

You should not use LGP$TERMINATE_PLOT with this routine, because the metafile has already been terminated by the program that created it.

Condition Values

Symbolic Status	Description
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use

LGP$PLOT_3D

The LGP$PLOT_3D plotting routine plots a three-dimensional graph on an axis system previously set up using LGP$SET_3D_GRAPH.

Format

LGP$PLOT_3D *(ws_number, [xarray], [yarray], zarray, nx, ny, [color], [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

xarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the x-axis values to be plotted. If this argument is defaulted, it is assumed that the x data are equally spaced.

yarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the y-axis values to be plotted. If this argument is defaulted, it is assumed that the y data are equally spaced.

zarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the z-axis values to be plotted.

nx
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of data points, or columns, contained in each array. The value of **nx** must be greater than or equal to 3.

ny
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of data points, or rows, contained in each array. The value of **ny** must be greater than or equal to 3.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See

Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

This routine allows you to plot a three-dimensional surface. You can plot multiple surfaces by making multiple LGP$PLOT_3D calls.

Each data array is a two-dimensional array of size (**nx**, **ny**). The x, y, z position of each point is determined by the entries in **xarray**, **yarray**, and **zarray**. If one of the coordinate arrays is defaulted, the plotted data is equispaced along the defaulted axis.

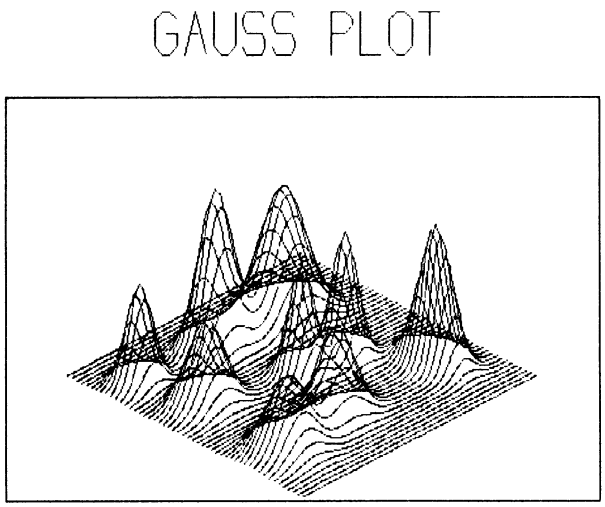
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_NX	nx less than 3
LGP$_ILL_NY	ny less than 3
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

Graphical Output

Figure 4–10 shows the output produced by the LGP$PLOT_3D routine:

Figure 4–10: Sample Output of LGP$PLOT_3D Routine



LGP$PLOTC

The LGP$PLOTC plotting routine plots a data set iteratively in a two-dimensional space previously defined by the LGP$PLOT routine or the LGP$SET_GRAPH routine.

Format

LGP$PLOTC *(ws_number, yarray, n, [n_start], [n_step], [iline], [color], [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

yarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the y-axis values. The array can be either one-dimensional or multidimensional. Whether the array is one-dimensional or multidimensional is determined by the values you supply to the **n** argument.

n
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the following three values: **npts**, **nsize**, and **nsets**. You must specify all three values, even when the **yarray** argument is one-dimensional.

The first entry, **npts**, specifies the number of data values in each line segment. The value must be greater than or equal to 0. No plotting action is taken if **npts** equals 0 or 1.

The second entry, **nsize**, specifies the total size of the array that you declared to hold the data values. Specify the total size of the declared array whether or not you are using all of the points in the array. For example, if you declared an array of size 100, but are plotting only the first 50 points stored in that array, **npts** is equal to 50, but **nsize** must be specified as 100. The value of **nsize** must be greater than or equal to **npts**.

The third entry, **nsets**, specifies the number of data sets to be drawn. Each set is one line or curve or other element on the graph. The value must be greater than or equal to 1.

n_start
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the data point at which plotting is to start. This value must be greater than or equal to 1 and less than **npts**. The default value is 1.

n_step
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the incremental value added to **n_start** after each data point is plotted. For example, if the value of **npts** is equal to 10, and the value of **n_start** is equal to 1, you can plot all data points (from 1 to 10, inclusive), by specifying **n_step** equal to 1.

The value of this argument must be less than or equal to **npts - n_start**. The default value is 1.

iline
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the type of line to draw. The value of **iline** can be one of the following:

Value	Line Type
1	Solid line
2	Dashed line
3	Dotted line
4	Dashed-dotted line

The default value is 1. Any integer outside the range (1 to 4) generates an error.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Use the LGP$PLOTC routine to update a plot dynamically by erasing old data and displaying new data as received in real time, or as produced by an iterative numerical calculation.

LGP$K_SCREENDIV in the plotting attribute list specifies how the plotting display window is divided during real-time plotting. The default value is 5.0, which divides the plotting display window into 5 line segments during real-time plotting.

LGP$K_ERASE_COND in the plotting attribute list specifies how the plotting display window is erased during real-time plotting. The default value is 0.0, which erases each individual line segment and displays updated real-time data after the out-of-date real-time data in that line segment is erased.

The plotting and erasing occur in sequence for each specified line segment of each channel plotted. For example, two line segments are to be plotted. The display window is divided into 5 line segments (LGP$K_SCREENDIV is 5.0). The screen erases individual line segments and plots updated real-time data in each segment after the out-of-date data is erased (LGP$K_ERASE_COND is 0.0).

During real time plotting, the first line segment of channel 1 is plotted; the first line segment of channel 2 is plotted; the second line segment of channel 1 is plotted; the second line segment of channel 2 is plotted, and so forth, until the 5 line segments for each of the 2 channels is plotted. Then, the first line segment of channel 1 is erased, and new data is plotted in that line segment. The first line segment of channel 2 is erased, and new data is plotted in that line segment. This continues until all the real-time data from each channel is plotted.

See Chapter 2 for information about the plotting attribute list supplied by VAXlab and the routines you use to modify this list.

You must call the LGP$PLOT or LGP$SET_GRAPH routine before you call LGP$PLOTC. The axis coordinates are determined by the LGP$PLOT or LGP$SET_GRAPH routine.

Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_ILINE	iline is not 1, 2, 3, or 4
LGP$_ILL_N_START	n_start is greater than npts or less than 1
LGP$_ILL_N_STEP	n_step is greater than npts - n_start or less than 1
LGP$_ILL_NPTS	npts is less than or equal to 0
LGP$_ILL_NROW	nrow is less than 1
LGP$_ILL_PROC_PLOTC	improper calling order
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use
LGP$_WSN_NOT_IN_USE	workstation number not in use

LGP$PLOTM

The LGP$PLOTM plotting routine plots additional data contained in an array to an existing graph.

Format

LGP$PLOTM *(ws_number, mode_string, [xarray], yarray, n, [status] , [iline], [color])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

mode_string
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
A character string containing four characters of value IXSY, IXMY, EXSY, or EXMY. The first letter of the string is either **I** (implicit) or **E** (explicit), specifying an implicit or explicit x array. The second letter is always **X** (x array). The third letter is either **S** (single) or **M** (multiple), specifying a one-dimensional or a multidimensional array. The fourth letter is always **Y**.

The character string is not case sensitive. You can specify the character string in either uppercase or lowercase letters.

xarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the x-axis values to be plotted.

If **mode_string** specifies an explicit x array (EXSY or EXMY), the **xarray** argument contains **npts** data values in the x array. **If an explicit x array is specified, the xarray argument is required.**

If **mode_string** specifies an implicit x array (IXSY or IXMY), the single value in the **xarray** argument is equal to the x value between data points, and the first x value of data is assumed to be 0. **If an implicit x array is specified, the xarray argument is optional.**

When **mode_string** specifies an implicit x array, the plotting of data across the x axis and the x-axis annotation vary according to the following argument specifications:

- If you default both the **xarray** argument and the **xcontrol** argument, the data is plotted at equal increments across the x axis. The x axis is not annotated.
- If you default the **xarray** argument, and you specify the **xcontrol** argument, the data is plotted at equal increments across the x axis independently of the x-axis annotation.
- If you specify the **xarray** argument, and you default the **xcontrol** argument, the data is plotted at equal increments across the x axis. The x-axis annotation is determined by the autoscaling algorithm.
- If you specify both the **xarray** argument and the **xcontrol** argument, the data is plotted beginning with x = 0, with the spacing determined by the single value contained in the **xarray** argument. The x-axis annotation is determined by the values contained in the **xcontrol** argument.

yarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array containing the independent y-axis values. The array can be either one-dimensional (IXSY, EXSY), or multidimensional (IXMY or EXMY).

n
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the following three values: **npts**, **nsize**, and **nsets**. If a one-dimensional array is specified (IXSY, EXSY), then **n** need only be a single longword containing **npts**. If a two-dimensional array is specified (IXMY, EXMY), then you must specify both **nsize** and **nsets**.

The first entry, **npts**, specifies the number of data values in each line segment. The value must be greater than or equal to 0. No plotting action is taken if **npts** equals 0 or 1.

The second entry, **nsize**, specifies the total size of the array that you declared to hold the data values. Specify the total size of the declared array whether or not you are using all of the points in the array. For example, if you declared an array of size 100, but are plotting only the first 50 points stored in that array, **npts** is equal to 50, but **nsize** must be specified as 100. The value of **nsize** must be greater than or equal to **npts**.

The third entry, **nsets**, specifies the number of data sets to be drawn. Each set is one line or curve or other element on the graph. The value must be greater than or equal to 1.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

iline
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the type of line to draw. The value of **iline** can be one of the following:

Value	Line Type
1	Solid line
2	Dashed line
3	Dotted line
4	Dashed-dotted line

The default value is 1. Any integer outside the range (1 to 4) generates an error.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

Description

This routine plots additional data sets in two dimensions. You must call the LGP$PLOT, LGP$PLOT_LOG, or LGP$SET_GRAPH routine before you call LGP$PLOTM. The axis coordinates are determined by the LGP$PLOT or LGP$SET_GRAPH routine.

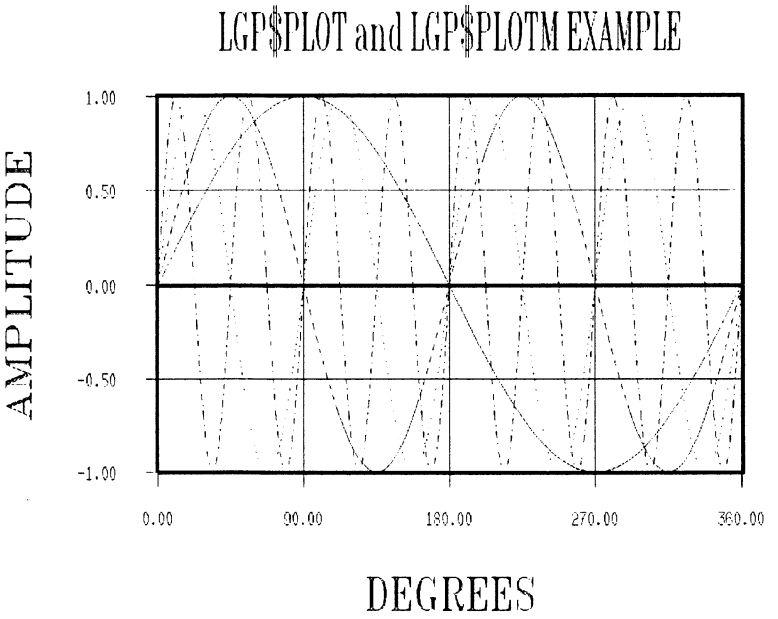
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_ILINE	iline is not 1, 2, 3, or 4
LGP$_ILL_MODE_STRING	illegal mode_string argument
LGP$_ILL_NPTS	npts is less than or equal to 0
LGP$_ILL_NROW	nrow is less than 1
LGP$_ILL_PROC_PLOTM	improper calling order
LGP$_ILL_XARRAY	illegal first value in xarray
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use
LGP$_WSN_NOT_IN_USE	workstation number not in use

Graphical Output

Figure 4–11 shows the output produced by the LGP$PLOTM routine after a call to LGP$PLOT:

Figure 4–11: Sample Output of LGP$PLOTM Routine



LGP$POINT

The LGP$POINT plotting routine plots a data point or an array of data points.

Format

LGP$POINT *(ws_number, xpos, ypos, [size], [igon], [npts], [status], [color], [ishade])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

xpos
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the x-axis value of a point. By specifying the **npts** argument, you can plot many points with one routine call. If you specify the **npts** argument larger than 1, the **xpos** argument is an array.

ypos
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the y-axis value of a point. By specifying the **npts** argument, you can plot many points with one routine call. If you specify the **npts** argument larger than 1, the **ypos** argument is an array.

size
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the pseudoradius of the object in inches. The pseudoradius is the dimension of the object as measured from the object's center of gravity to the farthest point on its perimeter, specifically:

Igon	Object	Pseudoradius
1	Dot	Not applicable
2	Plus	Distance from center to end
3	Asterisk	Distance from center to end
4	Circle	Circle radius
5	Diagonal	Distance from center to end
6	Triangle	Distance from center to a corner
7	Diamond	Distance from center to a corner
8	Square	Distance from center to a corner
9	Star	Distance from center to end
10	Hexagon	Distance from center to a corner

igon
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the type of data point to be drawn. See the definition of the **size** argument above for the legal values for this argument.

Any integer outside this range (1-10) generates an error. The default is 1 (dot).

npts
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of data points to be plotted. The default value is 1.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

ishade
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying how to shade a point. The value of **ishade** can be one of the following.

Value	Interior Shading
0	Interior unfilled
1	Interior filled
2	Interior cross-hatched
3	Interior filled with background color

The default value is 0. Any integer outside the range (0 to 3) generates an error.

Description

You use this routine to plot data points on graphs. You must call the LGP$PLOT, LGP$PLOT_LOG, or LGP$SET_GRAPH routine to set up the axis system before you call LGP$POINT.

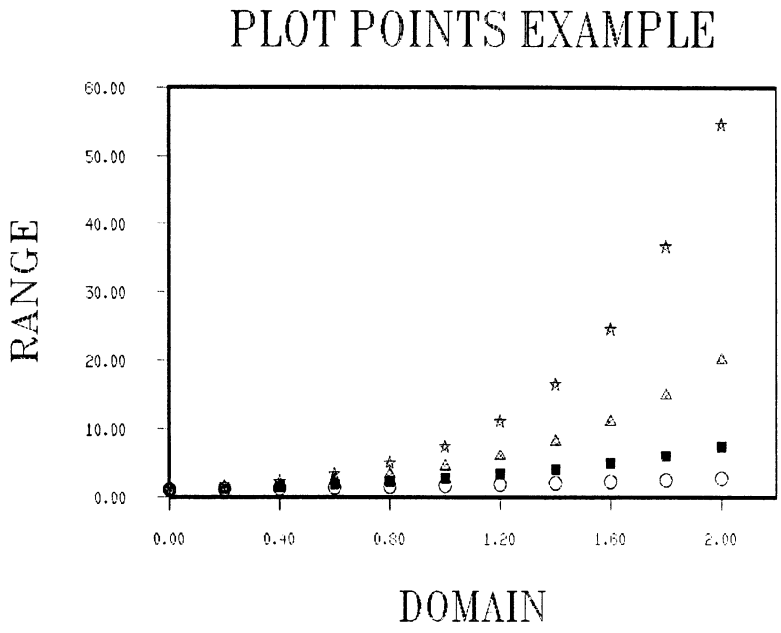
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_IGON	igon is out of range
LGP$_ILL_ISHADE	ishade is out of range
LGP$_ILL_NPTS	npts is less than or equal to 0
LGP$_ILL_PROC_POINT	improper calling order
LGP$_ILL_SIZE	size is out of range
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use
LGP$_WSN_NOT_IN_USE	workstation number not in use

Graphical Output

Figure 4–12 shows the output produced by the LGP$POINT routine:

Figure 4–12: Sample Output of LGP$POINT Routine



LGP$PRINT_SCREEN

The LGP$PRINT_SCREEN plotting routine copies the plot currently displayed on the terminal screen to a hardcopy device.

Format LGP$PRINT_SCREEN *([size])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

size
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the size of the hardcopy plot. **This argument is used only with the VT125 terminal.** If the value is 0, the hardcopy plot is the same size as the screen plot. If the value is nonzero, the hardcopy plot is twice the size of the screen image. The default value is 0.

Description

Use this routine to produce hardcopy plots directly from the terminal screen to an LA12, LA34, LA50, LA75, LA100, or LA210 device.

You cannot use this routine to produce hardcopy plots on the VAXlab-supported plotters. This routine works only when the printer is connected directly to the terminal printer port. It does not work when the printer is connected to another VAX serial port.

Unlike the VT240, VT241, VT330, and VT340 terminals, the VT125 terminal does not support a function key to produce a screen dump. This routine provides a screen dump capability.

NOTE

You can use this routine only with the VT125, VT240, VT241, VT330, and VT340 terminals.

LGP$PUT_TEXT

The LGP$PUT_TEXT routine writes text to a plot by writing a string to a specified starting location on the plot.

Format

LGP$PUT_TEXT *(ws_number, xpos, ypos, string, [size], [angle], [text_path], [font], [color], [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

xpos
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the starting x-axis position (relative to the lower lefthand corner) of the text string in inches.

ypos
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the starting y-axis position (relative to the lower lefthand corner) of the text string in inches.

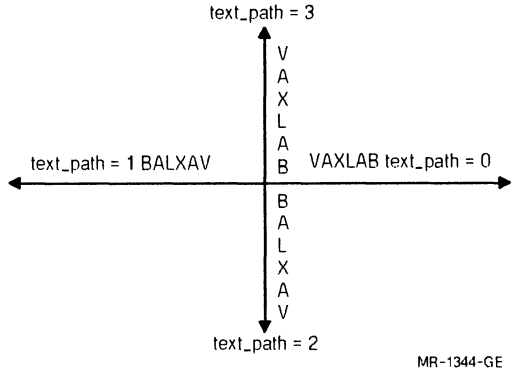
string
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
The text to be written on the plot, up to 72 characters in length.

size
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the height of the text string in inches. The value of this argument specifies the actual height of the text characters from the bottom to the top of the characters, even when the characters are written vertically on the plot. The default value is 0.15 inches.

angle
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the angle of rotation, in degrees, at which the text string is to be drawn on the plot with respect to the x axis. The value of this argument can be positive, negative, or 0 degrees. The default value is 0 degrees.

text_path
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the way in which the characters are assembled. The value of this argument can be 0 to 3.

The following figure illustrates the effect of each of the values of **text_path**.



font
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the font type used to write the text. Acceptable values for this argument are -32768 and 32768, inclusive. See the *DEC GKS Reference Manual* for a description of the font types available and the codes you use to specify the appropriate font types for each of the different devices. Any integer outside the range generates an error.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Use this routine to write a text string on a plot. Note that **xpos** and **ypos** are relative to the lower lefthand corner of the axes.

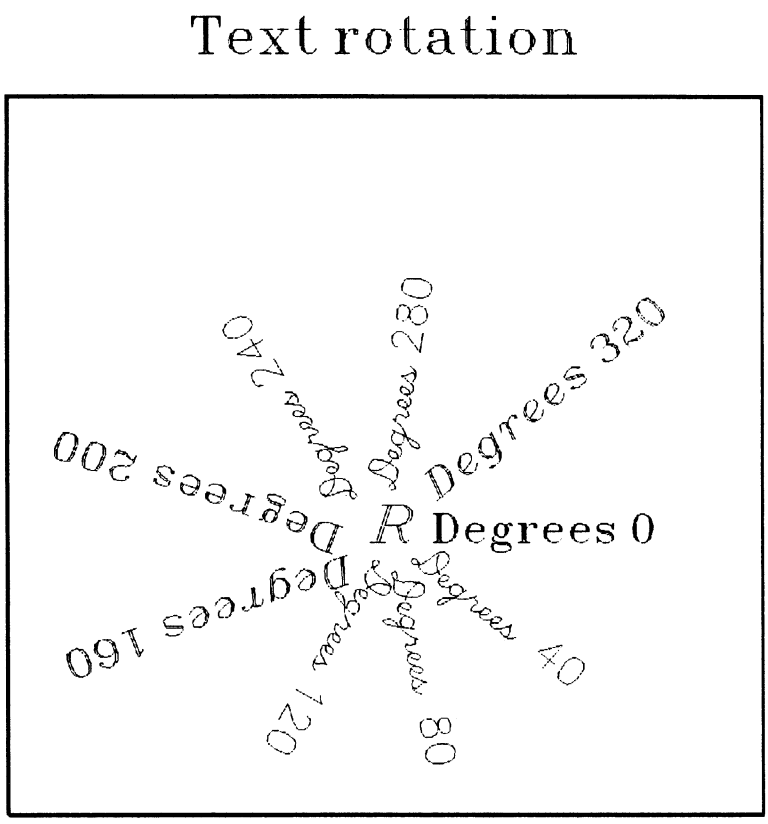
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_FONT	font number is out of range
LGP$_ILL_PUT_TEXT	LGP$PUT_TEXT was called before LGP$PLOT
LGP$_ILL_SIZE	size is out of range
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_NOT_IN_USE	workstation number not in use

Graphical Output

Figure 4–13 shows the output produced by the LGP$PUT_TEXT routine:

Figure 4–13: Sample Output of LGP$PUT_TEXT Routine



LGP$SCALE

The LGP$SCALE routine accesses the autoscaling algorithm used in a linear plot. The values returned are the minimum data point and the size of the increment needed for the number of divisions you specified. This routine does not produce a plot.

Format LGP$SCALE *(array, [n], ndiv, fval, increm, [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

array
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the array of data values to be searched for the maximum and minimum values. If the maximum and minimum values are known, specify the **npts** argument as 2 (the default value).

n
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference, array reference**
An array of length three specifying the following values: **npts**, **nsize**, and **nsets**. You must specify all three values, even when the **yarray** argument is one-dimensional.

The first entry, **npts**, specifies the number of data values in each line segment. The default value is 2. Any integer less than 2 generates an error.

The second entry, **nsize**, specifies the total size of the array you declared. Specify the total size of the array whether or not you are using all of the declared area. The value must be greater than or equal to **npts**.

The third entry, **nsets**, specifies the number of data sets to be scaled. Each set is one line or curve or other element on the graph. The value must be greater than or equal to 1.

ndiv
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of divisions to be made on the axis. Any integer less than 2 generates an error.

fval
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference**
An argument returning the first value of the plot. This value is determined by searching the data sets.

increm
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference**
An argument returning the plotting-increment value. The value is determined by the formula:

(max - fval) / ndiv

The maximum value **max** can be calculated by the formula:

fval + (ndiv * increm)

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Using the LGP$SCALE routine, you can access the autoscaling algorithm used by the LGP$PLOT routine. LGP$SCALE is called when you want to specify a value other than the default value for the length element of the **xcontrol** or **ycontrol** arrays of the LGP$PLOT routine. Create the **xcontrol** or **ycontrol** arrays with the length wanted and the values returned by this routine.

Condition Values

Symbolic Status	Description
LGP$_ILL_NDIV	ndiv is less than 2
LGP$_ILL_S_NPTS	value of npts in LGP$SCALE is less than 2
LGP$_ILL_S_ROW	value of nrow in LGP$SCALE is less than 1
LGP$_ILL_S_SIZE	value of size in LGP$SCALE is less than npts
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

LGP$SCALE_LOG

The LGP$SCALE_LOG routine accesses the autoscaling algorithm used in a logarithmic plot. The values returned are the minimum data point and the size of the increment needed for the number of divisions you specified. This routine does not produce a plot.

Format

LGP$SCALE_LOG *(array, [n], ndiv, fval, increm, [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

array
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the array of data values to be searched for the maximum and minimum values. If the maximum and minimum values are known, specify the **npts** argument as 2 (the default value).

n
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference, array reference**
An array of length three specifying the following values: **npts**, **nsize**, and **nsets**. You must specify all three values, even when the **yarray** argument is one-dimensional.

The first entry, **npts**, specifies the number of data values in each line segment. The default value is 2. Any integer less than 2 generates an error.

The second entry, **nsize**, specifies the total size of the array you declared. Specify the total size of the array whether or not you are using all of the declared area. The value must be greater than or equal to **npts**.

The third entry, **nsets**, specifies the number of data sets to be scaled. Each set is one line or curve or other element on the graph. The value must be greater than or equal to 1.

ndiv
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **write only**
mechanism: **by reference**
An argument specifying the number of divisions to be made on the axis. Any integer less than 2 generates an error.

fval
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference**
An argument returning the first value of the plot. This value is determined by searching the data sets.

increm
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference**
An argument returning the plotting-increment value from the expression **fval** + **increm** ** (increment number - 1). This number is always a power of ten, for example:

```
10, 100, 10,000, ...
```

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Using the LGP$SCALE_LOG routine, you can access the autoscaling algorithm used by the VAXlab plotting routine LGP$PLOT_LOG. LGP$SCALE_LOG is called when you want to specify a value other than the default value for the length element of the **xcontrol** or **ycontrol** arrays of the LGP$PLOT_LOG routine. Create the **xcontrol** or **ycontrol** arrays with the length wanted and the values returned by this routine.

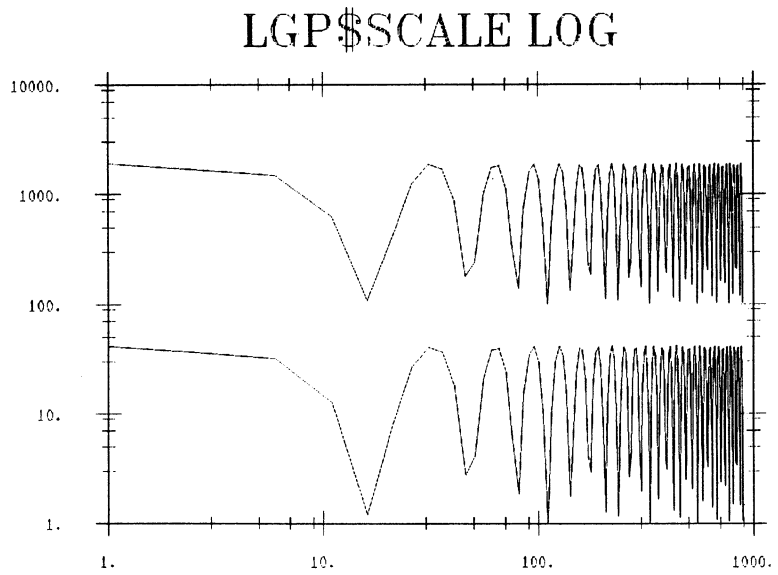
Condition Values

Symbolic Status	Description
LGP$_ILL_LOG_DATA	Data in the array is less than or equal to 0
LGP$_ILL_S_NPTS	value of npts in LGP$SCALE is less than 2
LGP$_ILL_S_ROW	value of nrow in LGP$SCALE is less than 1
LGP$_ILL_S_SIZE	value of size in LGP$SCALE is less than npts
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

Graphical Output

The example program LGP_SCALE_LOG.C uses LGP$SCALE_LOG, passes the values returned to LGP$PLOT_LOG, and produces the output shown in Figure 4–14:

Figure 4–14: Sample Output of LGP$SCALE_LOG Routine



LGP$SET_GRAPH

The LGP$SET_GRAPH routine sets up an x, y coordinate system for two-dimensional plotting.

Format

LGP$SET_GRAPH *(ws_number, xcontrol, ycontrol, [graph_type], [mode], [title], [xlabel], [ylabel], [grid_control], [status], [metaflag], [metafile_name])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

xcontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following x-axis parameters:

The first value, **xlen**, specifies the length of the x axis in inches.

The second value, **xlow**, specifies the first x-axis annotation value.

The third value, **xhi**, specifies the last x-axis annotation value. This value is the maximum x-axis value.

The fourth value, **xdelta**, specifies the increment at which numerical values are annotated on the x axis. If this value is 0, the x axis is not annotated.

ycontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following y-axis parameters:

The first value, **ylen**, specifies the length of the y axis in inches.

The second value, **ylow**, specifies the first y-axis annotation value.

The third value, **yhi**, specifies the last y-axis annotation value. This is the maximum y-axis value.

The fourth value, **ydelta**, specifies the increment at which numerical values are annotated on the y axis. If this value is 0, the y axis is not annotated.

graph_type
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the type of graph to be created. This feature is not currently implemented. The type of graph created is a linear graph.

mode
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying a scaled logarithmic or linear coordinate system for either the x axis, the y axis, or both.

Use the following values to specify the four possible scaling conditions. The default is 0.

Value	Scaling Condition
0	linear x axis and linear y axis
1	logarithmic x axis and linear y axis
2	linear x axis and logarithmic y axis
3	logarithmic x axis and logarithmic y axis

title
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the graph title up to 72 characters in length. The title is automatically centered above the graph. If the **title** argument is omitted, a title is not printed.

xlabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the x-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

ylabel
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the y-axis label up to 72 characters in length. If this argument is omitted, no label is printed.

grid_control
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An array of length two. The first value of **grid_control** specifies the number of grid lines per major increment on the x axis. The second value of **grid_control** specifies the number of grid lines per major increment on the y axis. The value of **grid_control** for the x axis and the y axis can be one of the following:

Value	Number of Grid Lines
0	No grid lines
1	One grid line for each number label on the axis
2	Two grid lines for each number label on the axis
	.
	.
	.
100	100 grid lines for each number label on the axis

Any integer outside the range (0 to 100) generates an error.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

metaflag
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying where to output the plot. The value of **metaflag** can be one of the following:

Value	Output is directed to:
0	Screen or plotter only
1	Metafile only
2	Screen or plotter and a metafile

Any integer outside this range (0 to 2) generates an error. The default is 0.

metafile_name
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the name of the metafile. The default metafile name is LGPMETAFILE.DAT.

Description

The LGP$SET_GRAPH routine sets up an axis system for two-dimensional plotting. This routine does not plot data.

After setting up an axis system with the LGP$SET_GRAPH routine, use LGP$PLOTM, LGP$PLOTC, LGP$POINT, LGP$STNDEV, LGP$HIST, or LGP$PUT_TEXT to plot data.

NOTE

Do not use LGP$SET_GRAPH if you are using LGP$PLOT or LGP$PLOT_LOG.

Condition Values

Symbolic Status	Description
LGP$_ILL_IGRID	igrid is out of range
LGP$_ILL_METAFLAG	metaflag is not 0, 1, or 2
LGP$_ILL_PORT	output port is not recognized
LGP$_ILL_WSN	illegal workstation number
LGP$_ILL_XDELTA	xdelta is out of range
LGP$_ILL_XLEN	xlen is out of range
LGP$_ILL_XLOW	xlow is not less than xhigh
LGP$_ILL_YDELTA	ydelta is out of range
LGP$_ILL_YLEN	ylen is out of range
LGP$_ILL_YLOW	ylow is not less than yhigh
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use

LGP$SET_3D_GRAPH

The LGP$SET_3D_GRAPH routine sets up a graph for three-dimensional plotting. Later, plot the graph using LGP$PLOT_3D.

Format

LGP$SET_3D_GRAPH *(ws_number, [xcontrol], [ycontrol], zmin, zmax, [status], [rot], [tilt], [title], [metaflag], [metafile_name])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

xcontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following x-axis parameters:

The first value, **xlen**, specifies the length of the x axis in inches.

The second value, **xlow**, specifies the minimum x value to be plotted.

The third value, **xhigh**, specifies the maximum x value to be plotted.

The fourth value, **xdelta**, specifies the increment at which numerical values are annotated on the x axis. This feature is not currently implemented. If you do not specify a value, you may declare your array to be either of length three or of length four.

ycontrol
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length four specifying the following y-axis parameters:

The first value, **ylen**, specifies the length of the y axis in inches.

The second value, **ylow**, specifies the minimum y value to be plotted.

The third value, **yhigh**, specifies the maximum y value to be plotted.

The fourth value, **ydelta**, specifies the increment at which numerical values are annotated on the y axis. This feature is not currently implemented. If you do not specify a value, you may declare your array to be either of length three or of length four.

zmin
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the minimum z value to be plotted. Values less than **zmin** are not plotted. The value of **zmin** must be less than the value of **zmax**.

zmax
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the maximum z value to be plotted. Values greater than **zmax** are not plotted. The value of **zmax** must be greater than the value of **zmin**.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

rot
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the angle, in degrees, at which the base of the plot (x, y plane) is rotated with respect to its origin. Positive values represent clockwise rotation viewed by looking down the positive z axis toward the origin. Negative values represent counterclockwise rotation.

tilt
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the angle, in degrees, at which the front of the plot (x, z plane) is rotated with respect to its origin. Positive values represent clockwise rotation viewed by looking down the positive x axis toward the origin. Negative values represent counterclockwise rotation.

title
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the graph title up to 72 characters in length. The title is automatically centered above the graph. If the **title** argument is omitted, a title is not printed.

metaflag
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying where to output the plot. The value of **metaflag** can be one of the following:

Value	Output is directed to:
0	Screen or plotter only
1	Metafile only
2	Screen or plotter and a metafile

Any integer outside this range (0 to 2) generates an error. The default is 0.

metafile_name
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the name of the metafile. The default metafile name is LGPMETAFILE.DAT.

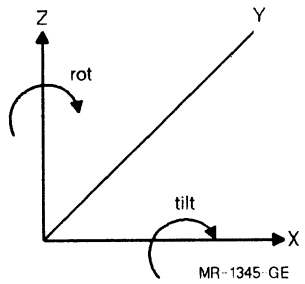
Description

This routine sets up an axis system for plotting data given in three-dimensional format. Use LGP$PLOT_3D to plot the graph.

NOTE

Do not use LGP$SET_3D_GRAPH if you are using LGP$3D_SIMPLE.

The following figure shows the effect of the **rot** and **tilt** arguments:



Condition Values

Symbolic Status	Description
LGP$_ILL_METAFLAG	metaflag is not 0, 1, or 2
LGP$_ILL_PORT	output port is not recognized
LGP$_ILL_WSN	illegal workstation number
LGP$_ILL_ZMAX	zmin is greater than or equal to zmax
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use

LGP$SPLINE_QHC

The LGP$SPLINE_QHC routine generates the spline values for smooth curve plotting and accesses the smooth curve-fitting algorithm to interpolate points between given input values. Optionally, this routine also provides the coefficients for the polynomial equation produced by the routine for smooth curve fitting.

Format

LGP$SPLINE_QHC *([xarray], yarray, n, n_subint, [x_spl_segment], [y_spl_segment], [coef], [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

xarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array specifying the independent (x) values used for calculating either the spline minisegments, the coefficients used for calculating the minisegments, or both. If this argument is defaulted, the routine assumes that the independent variable value (x) is equispaced with starting value 1 and incremental value 1.

yarray
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array specifying the dependent (y) values used for calculating either the spline minisegments, the coefficients used for calculating the minisegments, or both.

n
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the length of the data set. This value must be greater than or equal to 2, and less than 512.

n_subint
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of minisegments to be calculated between each (x,y) data pair. This value must be greater than or equal to 1, and less than or equal to 100. This value does not affect the **coef** array.

x_spl_segment
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference, array reference**
An array ((n-1)*n_subint+1) in length that returns the spline data as the x value of the minisegments. This argument is optional but, if you choose to include it in the argument list, also include the **y_spl_segment** argument, and default the **coef** argument.

y_spl_segment
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference, array reference**
An array ((n-1)*n_subint+1) in length that returns the spline data as the y value of the minisegments. This argument is optional but, if you choose to include it in the argument list, also include the **x_spl_segment** argument, and default the **coef** argument.

coef
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference, array reference**
A two-dimensional array (n-1,4) in length that returns the polynomial coefficients of each minisegment. This argument is optional but, if you choose to include it in the argument list, default both the **x_spl_segment** and **y_spl_segment** arguments.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Use this routine for data interpolation. For more information, see *Numerical Recipes*, Press, Flannery, Teukolsky, and Vetterling, Cambridge University Press, 1987.

Condition Values

Symbolic Status	Description
LGP$_ILL_ARRAY	input arrays contain invalid data
LGP$_ILL_N	n is out of range
LGP$_ILL_N_SUBINT	n_subint is out of range
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

LGP$STNDEV

The LGP$STNDEV plotting routine plots a standard deviation marker.

Format

LGP$STNDEV *(ws_number, xpos, ypos, stndev, [npts], [status], [color])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

xpos
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the x-axis position of a marker. By specifying the **npts** argument, you can plot many standard deviation markers with one routine call. If **npts** is greater than one, **xpos** is an array.

ypos
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the y-axis position of a marker. By specifying the **npts** argument, you can plot many standard deviation markers with one routine call. If **npts** is greater than one, **ypos** is an array.

stndev
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An argument specifying the magnitude of the standard deviation. By specifying the **npts** argument, you can plot many standard deviation markers with one routine call.

npts
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of standard deviation markers to be plotted. The default value is 1. Any integer less than 1 generates an error.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

Description

Use the LGP$STNDEV routine to plot standard deviation markers. You must call the LGP$PLOT routine before you call LGP$STNDEV.

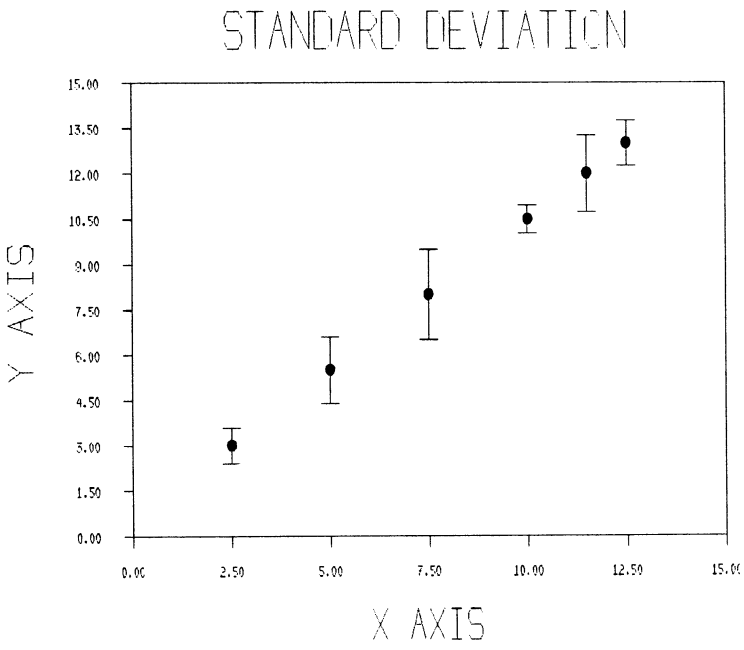
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_NPTS	npts is less than or equal to 0
LGP$_ILL_PROC_STNDEV	improper calling order
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use
LGP$_WSN_NOT_IN_USE	workstation number not in use

Graphical Output

Figure 4–15 shows the output produced by the LGP$STNDEV routine:

Figure 4–15: Sample Output of LGP$STNDEV Routine



LGP$3D_SIMPLE

The LGP$3D_SIMPLE plotting routine plots a three-dimensional array.

Format LGP$3D_SIMPLE *(ws_number, array3, nx, ny, zmin, zmax, [mode], [xlen], [ylen], [status], [rot], [tilt], [color], [title], [metaflag], [metafile_name])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually.

array3
VMS Usage: **floating_point**
type: **f_floating**
access: **read only**
mechanism: **by reference, array reference**
A two-dimensional array specifying the z values to be plotted. The array must contain a minimum of 3 x 3 points to a maximum of 1024 x 1024

points. The x and y values are assumed to consist of equispaced data values.

nx
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of columns contained in the **array3** argument. The value of **nx** must be greater than or equal to 3.

ny
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the number of rows contained in the **array3** argument. The value of **ny** must be greater than or equal to 3.

zmin
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the minimum z value to be plotted. Values less than **zmin** are not plotted. The value of **zmin** must be less than the value of **zmax**.

zmax
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the maximum z value to be plotted. Values greater than **zmax** are not plotted. The value of **zmax** must be greater than the value of **zmin**.

mode
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting mode. If **mode** is 0, no mesh is drawn. If **mode** is nonzero, a mesh is drawn. The default value is 0.

xlen
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the size of the x,y projection of the 3-D plot in the x direction. The default value is 5 inches.

ylen
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the size of the x,y projection of the 3-D plot in the y direction. See Section 1.3, Plotting Device Support, for the y-axis default length for your terminal type.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

rot
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the angle, in degrees, at which the base of the plot (x-y plane) is rotated with respect to its origin. Positive values

represent clockwise rotation viewed by looking down the positive z axis toward the origin. Negative values represent counterclockwise rotation.

tilt
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the angle, in degrees, at which the front of the plot (x-z plane) is rotated with respect to its origin. Positive values represent clockwise rotation viewed by looking down the positive x axis toward the origin. Negative values represent counterclockwise rotation.

color
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference, array reference**
An array of length three containing three values (between 0.0 and 1.0, inclusive) representing the red, green, and blue intensities, respectively. Any number outside the range generates an error. See Section 1.8, Producing Multicolor Plots on the Terminal Screen, for more information about producing multicolor plots.

title
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the graph title up to 72 characters in length. The title is automatically centered above the graph. If the **title** argument is omitted, a title is not printed.

metaflag
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying where to output the plot. The value of **metaflag** can be one of the following.

Value	Output is directed to:
0	Screen or plotter only
1	Metafile only
2	Screen or plotter and a metafile

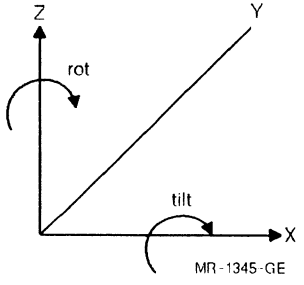
Any integer outside this range (0 to 2) generates an error. The default is 0.

metafile_name
VMS Usage: **char_string**
type: **character string**
access: **read only**
mechanism: **by descriptor**
An argument specifying the name of the metafile. The default metafile name is LGPMETAFILE.DAT.

Description

This routine plots data given in three-dimensional format with hidden lines removed. This routine draws an axis system.

The following figure shows the effect of the **rot** and **tilt** arguments:



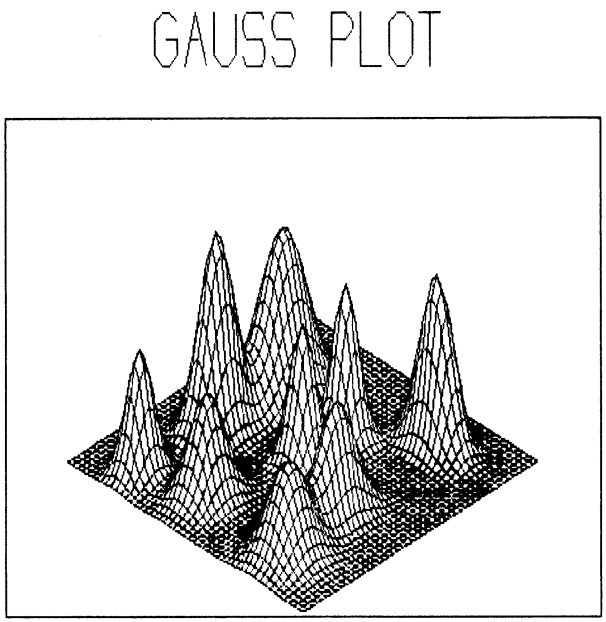
Condition Values

Symbolic Status	Description
LGP$_ILL_COLOR_ARRAY	color entry is out of range
LGP$_ILL_METAFLAG	metaflag is not 0, 1, or 2
LGP$_ILL_NX	nx less than 3
LGP$_ILL_NY	ny less than 3
LGP$_ILL_PORT	output port is not recognized
LGP$_ILL_WSN	illegal workstation number
LGP$_ILL_ZMAX	zmin is greater than or equal to zmax
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_IN_USE	workstation number is in use

Graphical Output

Figure 4–16 shows the output produced by the LGP$3D_SIMPLE routine:

Figure 4–16: Sample Output of LGP$3D_SIMPLE Routine



LGP$TABLE_MODIFY

The LGP$TABLE_MODIFY routine modifies values in the plotting attribute list.

Format LGP$TABLE_MODIFY *(tablentry, value, [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

tablentry
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the symbolic name of the attribute to be modified in the plotting attribute list. See Table 2–1, Plotting Attribute List Values, for a list of the VAXlab-supplied plotting attributes.

value
VMS Usage: **floating_point**
type: **F_floating**
access: **read only**
mechanism: **by reference**
An argument specifying the new attribute value. Any value outside the minimum and maximum values specified in the plotting attribute list generates an error, and the current attribute value remains unchanged. See Table 2–1, Plotting Attribute List Values, for a list of the VAXlab-supplied plotting attributes.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

The plotting attribute list contains default values for each of the plotting attributes associated with plotting, such as line thickness, fonts, and number of significant digits. Use this routine to modify the list.

Use the LGP$TABLE_RESET routine to restore the default values.

Condition Values

Symbolic Status	Description
LGP$_ILL_AT_LIST	attribute is reserved by DIGITAL
LGP$_ILL_AT_VALUE	attribute value out of range
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

LGP$TABLE_READ_VALUE

The LGP$TABLE_READ_VALUE routine returns the current values of the Plotting Attribute List.

Format

LGP$TABLE_READ_VALUE *(symbolic_name, value, [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

symbolic_name
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the symbolic name of the plotting attribute whose value you want returned. See Table 2–1, Plotting Attribute List Values, for a list of the VAXlab-supplied plotting attributes.

value
VMS Usage: **floating_point**
type: **F_floating**
access: **write only**
mechanism: **by reference**
An argument returning the current value of the plotting attribute specified in the **symbolic_name** argument. See Table 2–1 for a list of the VAXlab-supplied plotting attributes.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Use this plotting routine to examine values contained in the plotting attribute list.

Condition Values

Symbolic Status	Description
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success

LGP$TABLE_RESET

The LGP$TABLE_RESET routine restores the VAXlab-supplied default values to the plotting attribute list.

Format LGP$TABLE_RESET *(None)*

Returns

VMS Usage:	**cond_value**
type:	**longword (unsigned)**
access:	**write only**
mechanism:	**by value**

Arguments

None.

Description

Use this routine to restore the default values of the plotting attribute list previously modified with the LGP$TABLE_MODIFY routine.

LGP$TERMINATE_PLOT

The LGP$TERMINATE_PLOT routine signals the end of plotting information, clears the plotting screen, and closes the open workstation window.

Format LGP$TERMINATE_PLOT *(ws_number, [clear_screen], [status])*

Returns

VMS Usage: **cond_value**
type: **longword (unsigned)**
access: **write only**
mechanism: **by value**

Arguments

ws_number
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying the plotting workstation number. By specifying the plotting workstation number, you can reference multiple plots individually. The plotting screen is cleared when only one plot is on the screen. When more than one plot is on the screen, you can specify which plot to erase by using this argument.

clear_screen
VMS Usage: **longword_signed**
type: **longword integer (signed)**
access: **read only**
mechanism: **by reference**
An argument specifying whether or not to clear the workstation window. A zero specifies not clearing the workstation window. A nonzero value specifies clearing the workstation window. The default value is 0.

This argument is valid only for plots sent to terminal screens. Do not use this argument when producing hardcopy plots.

status
VMS Usage: **longword_unsigned**
type: **longword (unsigned)**
access: **write only**
mechanism: **by reference**
An argument returning the status of the operation. If the **status** argument is omitted and an error occurs, the message is directed to both SYS$OUTPUT and SYS$ERROR.

Description

Use this routine to clear plots off the display screen and to close open workstations. To keep a plot on the terminal screen for a predetermined period of time, use the LIB$WAIT routine described in the *VMS RTL Library (LIB$) Manual*.

You also use this routine to signal the end of plotting data when an LA12, LA34, LA50, LA75, LA100, LA210, or LN03 PLUS printer is individually programmed to produce a hardcopy plot.

You should not need use this routine to terminate the plot of a metafile, because the metafile has already been terminated by the program that created it.

Condition Values

Symbolic Status	Description
LGP$_ILL_WSN	illegal workstation number
LGP$_MAND_ARG	mandatory argument is missing
LGP$_SUCCESS	success
LGP$_WSN_NOT_IN_USE	workstation number not in use

Chapter 5

Error Messages

This chapter describes the LGP error-handling facility, explains the error messages, and gives suggested recovery procedures.

5.1 Overview

VAXlab Software Library provides an LGP error message facility. When you execute an image that results in an error, the system locates the error message associated with the error and directs it to the devices or files defined as SYS$ERROR and SYS$OUTPUT. The LGP routines use the same standards as the VMS Run-Time Library and System Services for passing back status information about routine calls.

The VMS Run-Time Library and System Services return a status value which is passed back to the user program through a longword variable when the routine is called as a function. A successful operation returns a success status value with bit 0 set (true). An unsuccessful operation returns one of the LGP symbolic values with bit 0 clear (false).

In addition, you can use an optional argument in each routine call list to obtain the routine status. Again, a successful operation returns bit 0 set (true).

If you call the routine as either a function or a subroutine, when an error condition exists and you do not include the optional status argument in the routine call, a message is directed to both SYS$OUTPUT and SYS$ERROR.

The LGP error code values are defined in symbolic status definition files, which must be included in your LGP programs. See Section 1.6, Including Symbolic Status Definition Files, for more information.

5.2 Checking Routine Call Status

A user program can check the status of routine calls in the following three ways:

1. By testing for status after each operation and, upon receipt of any condition other than success, signaling the condition value to the device or file defined as SYS$ERROR.

```
C Set up a coordinate system with value.  Include the status argument
C in the routine call argument list.

        INTEGER*4 STATUS
        .
        .
        .
        CALL LGP$PLOT(1,STRING,,0,N,LABELX,LABELY,STATUS, , ,
       -        XCONTROL,YCONTROL,,LABELM)
        IF (.NOT. STATUS)  CALL LIB$SIGNAL(%VAL(STATUS))
```

2. By testing status after each operation for a specific condition value.

```
      INCLUDE 'SYS$LIBRARY:LGPDEF.FOR  ! Get symbolic status definitions
      INTEGER*4 STATUS
      .
      .
      .
C Set up the coordinate axis and labels. The LGP plot routine returns
C the status.

      STATUS = CALL LGP$PLOT(1,MODE,,POS_ARRAY,1,X_LABEL,Y_LABEL, , ,
      - X_CONTROL,Y_CONTROL,,LABEL)

        IF (STATUS .NE. LGP$_SUCCESS) THEN
           CALL LIB$SIGNAL(%VAL(STATUS))
        ENDIF
```

This program segment tests if status is not equal to success. If status is not equal to success, the routine signals the status. If status equals success, program execution continues.

NOTE

If your program is coded to check for specific condition values after one or more operations, you must include the symbolic status definition file appropriate for your programming language. The symbolic status definition files supplied by VAXlab for error codes are listed in Table 1–3. If you do not include the symbolic status definition file, your program will not recognize these values.

3. By omitting the **STATUS** argument. If you omit the **STATUS** argument, and an error occurs, the message is directed to both SYS$ERROR and SYS$OUTPUT.

```
CALL LGP$PLOT(1,MODE,,POS_ARRAY,1,X_LABEL,Y_LABEL, , ,
- X_CONTROL,Y_CONTROL,,LABEL)
```

NOTE

Most plotting-routine error conditions are reported through the above mechanisms. However, under extreme conditions, such as passing a number on the verge of underflow or overflow, you may receive a GKS error or a VMS system error.

5.3 List of Error Messages

This section presents the LGP symbolic status values (error messages) with an explanation of each condition and the appropriate user action to recover from each condition.

LGP$_AXIS_RESIZED, axis too big, resized

Explanation: The x- and y-axis lengths you specified are too large for the device you are using. As a result, the **xlen** and **ylen** values reset to the default.

The **xlen** value of the **xcontrol** argument specifies the length of the x axis in inches. The **ylen** value of the **ycontrol** argument specifies the length of the y axis in inches.

User Action: Reduce the **xlen**, **ylen** values. Use the device-specific x, y axis values provided in Table 1–2.

LGP$_ILL_ARRAY, input arrays contain invalid data

Explanation: The data input to the **xarray** or **yarray** argument of the LGP$SPLINE_QHC routine is not in ascending order.

User Action: Change the **xarray** or **yarray** data so that all values are in ascending order. See the LGP$SPLINE_QHC reference description in Chapter 4 for more information.

LGP$_ILL_ARRAY_SIZE, choice array size is negative or 0

Explanation: You input a value less than or equal to 0 for the **choice_array_size** argument of the LGP$GET_CHOICE routine.

User Action: Change the **choice_array_size** argument to a value greater than 0.

LGP$_ILL_AT_LIST, attribute is reserved by DIGITAL

Explanation: You either did one of the following:

- Used an incorrect symbolic name for a plotting attribute

- Forgot to include the symbolic status definition file in your routine call program

User Action: Check the symbolic name used in your routine call against Table 1–2 and edit the symbolic name if it is incorrect. Include the appropriate symbolic status definition file in your program.

LGP$_ILL_AT_VALUE, attribute value out of range

Explanation: You entered an attribute value that is outside of the range specified in the attribute list.

User Action: See Table 2–1 for a list of LGP attributes and their corresponding value ranges. Adjust the program's attribute values accordingly.

LGP$_ILL_COLOR_ARRAY, color entry is out of range

Explanation: You entered a color array value that is outside the specified range.

The **color** array contains three numbers (between 0.0 and 1.0, inclusive) that specify the red, green, and blue intensities. Any number outside of this range generates an error.

User Action: Specify **color** array values between 0.0 and 1.0. See the *DEC GKS Reference Manual* for a color chart and a list of the possible **color** array values.

LGP$_ILL_ECHO_AREA, invalid echo area specified

Explanation: You entered an incorrect **echo_area** value for the LGPGET_CHOICE, LGPGET_STRING, or LGP$GET_VALUE routine.

User Action: Change the value of the **echo_area** argument for the routine you are using. See the appropriate reference description in Chapter 4 for more information about argument values.

LGP$_ILL_FONT, font number is out-of-range

Explanation: You entered an incorrect font value for the specific device you are using. The font values are between -32768 and 32768, inclusive.

User Action: Enter the device-specific font value. See the *DEC GKS Reference Manual, Volume II* for device-specific font values.

LGP$_ILL_H_XLOW, xlow is greater than or equal to xhi

Explanation: The starting value for a histogram element, **xlow**, is greater than or equal to the ending value, **xhi**.

In the LGP$HIST routine, the **xlow** array argument specifies the leftmost axis position of a histogram element. The **xhi** array argument specifies the rightmost position of the histogram element.

User Action: The starting value, **xlow**, must be less than the ending value, **xhi**. Adjust the **xlow** and **xhi** array values accordingly.

LGP$_ILL_IGON, igon is out of range

Explanation: You specified an invalid value for the **igon** argument in LGP$POINT.

The **igon** argument specifies the type of point to be drawn. The valid argument range is between 1 and 10, inclusive.

User Action: See the LGP$POINT reference description in Chapter 4 for more information about argument values.

LGP$_ILL_IGRID, igrid is out of range

Explanation: You specifed an invalid **igrid** argument value.

The **igrid** argument specifies the number of grid lines per major increment. The valid argument range is between 0 and 100, inclusive, where 0 specifies no grid lines and 100 specifies 100 grid lines.

User Action: Specify a valid **igrid** value.

LGP$_ILL_ILINE, iline is not 1, 2, 3, or 4

Explanation: You specified an invalid value for the **iline** argument.

The **iline** argument specifies the type of line to draw. Legal values are between 1 and 4, inclusive.

User Action: See the appropriate reference description in Chapter 4 for more information about argument values.

LGP$_ILL_INPUT_WS, workstation does not allow input

Explanation: You tried to perform an input function with a workstation that does not allow input.

User Action: Make sure you defined the logical name GKS$WSTYPE to be a workstation that is not output-only. See the *DEC GKS Reference Manual* and Table 1–3 in this guide for information about supported workstation types.

LGP$_ILL_ISHADE, ishade is out of range

Explanation: You specified an invalid **ishade** argument entry in the LGP$HIST or LGP$POINT routine.

The **ishade** argument specifies how to shade a histogram. The legal values are between 0 and 3.

User Action: See the LGP$HIST or LGP$POINT reference description in Chapter 4 for more information about argument values.

LGP$_ILL_LOG_DATA, Data in array is less than or equal to 0

Explanation: You entered a value less than or equal to 0 for the **array** argument of the LGP$SCALE_LOG routine.

User Action: Adjust the **array** argument to a value greater than 0.

LGP$_ILL_LOG_XLOW, xlow is not less than xhigh or less than or equal to 0.

Explanation: Your entry for the **xlow** value of the **xcontrol** argument in LGP$PLOT_LOG is greater than or equal to the value of **xhigh**, or **xlow** is less than or equal to 0

User Action: Specify an **xlow** value that is lower than the **xhigh** value.

LGP$_ILL_LOG_YLOW, ylow is not less than yhigh or less than or equal to 0

Explanation: Your entry for the **ylow** value of the **ycontrol** argument in LGP$PLOT_LOG is greater than or equal to the value of **yhigh**, or **ylow** is less than or equal to 0

User Action: Specify an **ylow** value that is lower than the **yhigh** value.

LGP$_ILL_METAFLAG, metaflag is not 0, 1, or 2

Explanation: You specified a value outside the valid **metaflag** range of 0, 1, or 2.

User Action: See the appropriate reference description in Chapter 4 for more information about argument values.

LGP$_ILL_MODE_STRING, illegal 'mode_string' argument

Explanation: You specified an invalid **mode_string** argument value for the LGP$PLOT or LGP$PLOTM routine.

User Action: Use one of the following **mode_string** values:

- IXSY
- IXMY
- EXSY
- EXMY

See the LGP$PLOT or LGP$PLOTM reference description in Chapter 4 for more information about argument values.

LGP$_ILL_N, n is out of range

Explanation: You specified an invalid value for the **n** argument of the LGP$SPLINE_QHC routine.

The **n** argument specifies the length of the dataset.

User Action: The **n** argument value must be between 2 and 512, inclusive. Adjust the **n** argument entry accordingly. See the LGP$SPLINE_QHC reference description in Chapter 4 for more information about argument values.

LGP$_ILL_N_START, n_start is greater than npts or less than 1

Explanation: You specified an invalid value for the **n_start** argument of the LGP$PLOTC routine.

The **n_start** argument specifies the data point at which plotting is to start. This value must be greater than or equal to 1 and less than the value of the **npts** argument. The default value is 1.

User Action: Specify an **n_start** value that is greater than or equal to 1 and less than the value of the **npts** argument.

LGP$_ILL_N_STEP, n_step is greater than npts - n_start or less than 1

Explanation: You specified an invalid value for the **n_step** argument of the LGP$PLOTC routine.

The **n_step** argument specifies the incremental value added to **n_start** after each data point is plotted. The value of this argument must be less than or equal to **npts** - **n_start**. The default value is 1.

User Action: Specify an **n_step** value that is less than or equal to **npts** - **n_start**.

LGP$_ILL_N_SUBINT, n_subint is out of range

Explanation: You specified an invalid value for the **n_subint** argument of the LGP$SPLINE_QHC routine.

The **n_subint** argument contains the number of minisegments to be calculated between each x, y data pair. This value must be between 1 and 100, inclusive.

User Action: Specify a value for the **n_subint** argument that is between 1 and 100, inclusive.

LGP$_ILL_NDIV, ndiv is less than 2

Explanation: You specified a value for the **ndiv** argument of the LGP$SCALE routine that is less than 2.

The **ndiv** argument specifies the number of divisions to be made on the axis. Any number less than 2 generates an error.

User Action: Specify an **ndiv** value that is greater than or equal to 2.

LGP$_ILL_NPTS, npts is less than or equal to 0

Explanation: You specified a value for the **npts** entry of the **n** argument that is less than or equal to 0.

The **npts** entry specifies the number of data values in each line segment. The value must be greater than or equal to 0. No plotting action takes place if **npts** equals 0 or 1.

User Action: Specify an **npts** value that is greater than 0.

LGP$_ILL_NSETS, nsets is less than 1

Explanation: You specified a value for the **nsets** entry of the **n** argument that is less than 1.

The **nsets** entry specifies the number of rows to be plotted. The value must be greater than or equal to 1.

User Action: Specify an **nsets** value that is greater than or equal to 1.

LGP$_ILL_NSIZE, nsize is less than 1

Explanation: You specified a value of less than 1 for the **nsize** entry of the **n** argument.

The **nsize** value specifies the number of columns to be plotted when multiple rows of data are to be plotted. The value must be greater than or equal to **npts**. The value of **npts** must be greater than or equal to 0. No plotting action takes place if **npts** equals 0 or one.

User Action: Specify an **nsize** value that is greater than or equal to **npts**.

LGP$_ILL_NX, nx less than 3

Explanation: You specified a value for the **nx** argument that is less than 3.

The **nx** argument specifies the number of columns contained in **array3**. The value of the **nx** argument must be greater than or equal to 3.

User Action: Specify an **nx** value that is greater than or equal to 3. See the appropriate reference description in Chapter 4 for more information about argument values.

LGP$_ILL_NY, ny less than 3

Explanation: You specified a value for the **ny** argument that is less than 3.

The **ny** argument specifies the number of rows contained in **array3**. The value of **ny** must be greater than or equal to 3.

User Action: Specify an **ny** value that is greater than or equal to 3. See the appropriate reference description in Chapter 4 for more information about argument values.

LGP$_ILL_PEN_NUMBER, illegal pen number

Explanation: You specified an invalid value for the **pnum** argument of the LGP$MAP_PEN routine.

The **pnum** argument specifies the pen number. The value of **pnum** must be between 1 and 16, inclusive.

User Action: Specify a **pnum** value that is between 1 and 16, inclusive. See the LGP$MAP_PEN reference description in Chapter 4 for more information about argument values.

LGP$_ILL_PORT, output port is not recognized

Explanation: LGP does not recognize the foreign device you are attempting to use.

User Action: To use a foreign device with LGP, you must install the appropriate device handler. See Chapter 1 for information on foreign device support and LGP device support.

LGP$_ILL_PORT_LEN, length of port string is illegal

Explanation: You used the LGP$CHANGE_PLOTTING_LOGICALS routine to define the logical name GKS$CONID. Your entry for **port** argument is outside of the specified limit.

The **port** argument defines the logical name GKS$CONID. This string value must be less than or equal to 31 characters in length.

User Action: Adjust the string value of the **port** argument accordingly.

LGP$_ILL_PROC_HIST, improper calling order

Explanation: You called the LGP$HIST routine to plot a histogram bar before calling the LGP$PLOT routine or the LGP$SET_GRAPH routine to set up an axis system for your graph.

User Action: First, call LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph. Then, call LGP$HIST to plot a histogram bar on the graph. See Chapter 1 for information about the LGP plotting routine call hierarchy. See the LGP$HIST reference description in Chapter 4 for more information.

LGP$_ILL_PROC_PLOTC, improper calling order

Explanation: You called LGP$PLOTC to plot a dataset iteratively before calling LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph.

User Action: First, call LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph. Then, call LGP$PLOTC to plot a dataset iteratively on your graph. See Chapter 1 for information about the LGP plotting routine call hierarchy. See the LGP$PLOTC reference description in Chapter 4 for more information.

LGP$_ILL_PROC_PLOTM, improper calling order

Explanation: You called LGP$PLOTM to plot an additional dataset before calling LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph.

User Action: First, call LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph. Then, call LGP$PLOTM to plot an additional dataset on your graph. See Chapter 1 for information about the LGP plotting routine call hierarchy. See the LGP$PLOTM reference description in Chapter 4 for more information about argument values.

LGP$_ILL_PROC_POINT, improper calling order

Explanation: You called LGP$POINT to plot a data point or a group of data points before calling LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph.

User Action: First, call LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph. Then, call LGP$POINT to plot a data point or a group of data points on your graph. See Chapter 1 for information about the LGP plotting routine call hierarchy. See the LGP$POINT reference description in Chapter 4 for more information.

LGP$_ILL_PROC_STNDEV, improper calling order

Explanation: You called LGP$STNDEV to plot a standard deviation marker before calling LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph.

User Action: First, call LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph. Then, call LGP$STNDEV to plot a standard deviation marker on your graph. See Chapter 1 for information about the LGP plotting routine call hierarchy. See the LGP$STNDEV reference description in Chapter 4 for more information.

LGP$_ILL_PUT_TEXT, improper calling order

Explanation: You called LGP$PUT_TEXT to write text to a plot before calling LGP$PLOT or LGP$SET_GRAPH to set up an axis system for your graph.

User Action: First, call any top-level routine to set up an axis system for your graph. Then, call LGP$PUT_TEXT to write a text string to your graph. See Chapter 1 for information about the LGP plotting routine call hierarchy. See the LGP$PUT_TEXT reference description in Chapter 4 for more information about argument values.

LGP$_ILL_S_NPTS, npts is less than 2

Explanation: Your entry for the **npts** value of the **n** array argument in LGP$SCALE or LGP$SCALE_LOG is less than 2.

The **npts** argument specifies the number of independent variable values in the array (default 2). Any number less than 2 generates an error.

User Action: Replace the current value of **npts** with a value of 2 or more. See the LGP$SCALE or LGP$SCALE_LOG reference description in Chapter 4 for more information about argument values.

LGP$_ILL_S_NSETS, nsets is less than 1

Explanation: Your entry for the **nsets** value of the **n** array argument in LGP$SCALE or LGP$SCALE_LOG is less than 1.

The **nsets** value specifies the number of rows in the array when multiple rows of data are to be plotted (default 1). Any number less than 1 generates an error.

User Action: Replace the current value of **nsets** with a value of 1 or more. See the LGP$SCALE or LGP$SCALE_LOG reference description in Chapter 4 for more information about argument values.

LGP$_ILL_S_NSIZE, nsize is less than npts

Explanation: Your entry for the **nsize** value of the **n** array argument in LGP$SCALE or LGP$SCALE_LOG is less than **npts**.

The **nsize** entry specifies the total size of the array you declared. The value must be greater than or equal to **npts**. A value less than **npts** generates an error.

User Action: Replace the current value of **nsize** with a value greater than or equal to **npts**. See the LGP$SCALE or LGP$SCALE_LOG reference description in Chapter 4 for more information.

LGP$_ILL_SIZE, size is out of range

Explanation: Your entry for the **size** argument of LGP$POINT or LGP$PUT_TEXT is less than or equal to 0.

User Action: Replace the current value of the **size** argument with a value greater than 0. See the LGP$POINT and LGP$PUT_TEXT reference description in Chapter 4 for more information about argument values.

LGP$_ILL_STNDEV, stndev is less than or equal to 0

Explanation: Your entry for the **stndev** array argument of the LGP$STNDEV routine is less than or equal to 0.

The **stndev** array argument specifies the magnitude of the standard deviation.

User Action: Replace the current value of the **stndev** array argument with a value greater than 0. See the LGP$STNDEV reference description in Chapter 4 for more information about argument values.

LGP$_ILL_TEXT_PATH, illegal text_path argument

Explanation: You entered an invalid value for the **text_path** argument of the LGP$PUT_TEXT routine.

When you call LGP$PUT_TEXT to write text to a plot, the **text_path** argument specifies the way in which the characters are displayed on the plot. The value of this argument can be 0 to 3.

User Action: Replace the current **text_path** value with a value from 0 to 3. See the LGP$PUT_TEXT reference description in Chapter 4 for more information.

LGP$_ILL_WKSTN_SIZE, LGP$WKSTN_XXX_SIZE is less or equal to 0

Explanation: While defining logicals for a foreign device, you entered a value less than or equal to 0 for LGP$WKSTN_XXX_SIZE.

User Action: Define LGP$WKSTN_XXX_SIZE as any number greater than 0. See Chapter 1 for information on foreign device support and LGP device support.

LGP$_ILL_WLIST_SIZE, wlist_size is less than or equal to 0

Explanation: You entered a value less than or equal to 0 for the **wslist_size** argument of the LGP$INFO routine.

When you call LGP$INFO to return information about a specified workstation number, the **wslist_size** argument specifies the size of the **WSLIST** argument.

User Action: Enter a value greater than 0 for the **wslist_size** argument. See the LGP$INFO reference description in Chapter 4 for more information about argument values.

LGP$_ILL_WSN, illegal workstation number

Explanation: You entered a value less than or equal to 0 for the **ws_number** argument.

The **ws_number** argument specifies the workstation number to which output is directed.

User Action: Replace the current **ws_number** value with a value greater than 0.

LGP$_ILL_WSTYPE, wstype is less then 0

Explanation: You defined the workstation type (GKS$WSTYPE) as a negative number.

User Action: Define the workstation type using the values that correspond to the specific LGP-supported device you are using. See Table 1-2 for a list of supported LGP devices and their corresponding workstation types.

LGP$_ILL_XARRAY, illegal first value is xarray

Explanation: You entered a negative value for the **xarray** argument of the LGP$PLOT routine.

When you call LGP$PLOT with an implicit x array (IXSY or IXMY), the single value in the **xarray** argument is equal to the x value between data points, and the first data point is assumed to be 0.

User Action: Specify a positive value for the **xarray** argument. See the LGP$PLOT reference description in Chapter 4 for more information about argument values.

LGP$_ILL_XDELTA, xdelta is out-of-range

Explanation: Your entry for the **xdelta** value of the **xcontrol** argument is larger than the range between the **xlow** and **xhigh** values.

The **xdelta** value of the **xcontrol** argument specifies the increment at which numerical values are annotated on the x axis. If this value is 0, the x axis is not annotated.

User Action: Specify a smaller value for **xdelta**, or expand the range between the **xlow** and the **xhigh** values.

LGP$_ILL_XLEN, xlen is out of range

Explanation: Your entry for the **xlen** value of the **xcontrol** argument is less than or equal to 0.

The **xlen** value specifies the length of the x axis in inches.

User Action: Specify an **xlen** value that is greater than 0.

LGP$_ILL_XLOW, xlow is not less than xhigh

Explanation: Your entry for the **xlow** value of the **xcontrol** argument is greater than or equal to the value of **xhigh**.

User Action: Specify an **xlow** value that is lower than the **xhigh** value.

LGP$_ILL_YDELTA, ydelta is out of range

Explanation: Your entry for the **ydelta** value of the **ycontrol** argument is larger than the range between the **ylow** and **yhigh** values.

The **ydelta** value of the **ycontrol** argument specifies the increment at which numerical values are annotated on the y axis. If this value is 0, the y axis is not annotated.

User Action: Specify a smaller **ydelta** value, or expand the range between **ylow** and **yhigh** values.

LGP$_ILL_YLEN, ylen is out of range

Explanation: Your entry for the **ylen** value of the **ycontrol** argument is less than or equal to 0.

The **ylen** value specifies the length of the y axis in inches.

User Action: Specify an **ylen** value that is greater than 0.

LGP$_ILL_YLOW, ylow is not less than yhigh

Explanation: Your entry for the **ylow** value of the **ycontrol** argument is greater than or equal to the value of **yhigh**.

User Action: Specify an **ylow** value that is lower than the **yhigh** value.

LGP$_ILL_ZMAX, zmin is greater than or equal to zmax

Explanation: In the LGP$3D_SIMPLE routine, the value of the **zmin** argument is greater than or equal to the value of the **zmax** argument.

The **zmin** argument specifies the minimum z value to be plotted. Values less than **zmin** are not plotted. The value of **zmin** must be less than the value of **zmax**.

User Action: Specify a value for the **zmin** argument that is less than the value of the **zmax** argument. See the LGP$3D_SIMPLE reference description in Chapter 4 for more information about argument values.

LGP$_MAND_ARG, mandatory argument is missing

Explanation: You omitted a required argument from the routine call argument list. The LGP facility does not provide a default value for the missing argument.

User Action: You must specify all mandatory arguments. See the appropriate reference description in Chapter 4 to determine which argument is missing. Edit your program to include the missing mandatory argument.

LGP$_SUCCESS, success

Explanation: LGP call completed successfully.

User Action: No user action necessary.

LGP$_WKSTN_SIZE_UNDEF, Logical for foreign device is not defined

Explanation: You neglected to define the logical name for the foreign device you are using.

User Action: See Section 1.5.1, Defining Logical Names for Foreign Devices, for more information.

LGP$_WSN_IN_USE, workstation number is in use

Explanation: The workstation number you specified is already in use.

User Action: Specify an open workstation number. See the appropriate reference description in Chapter 4 to assure that you entered the correct argument values.

LGP$_WSN_NOT_IN_USE, workstation number not in use

Explanation: The workstation number you specified has not been set up by a previous routine call.

User Action: See the appropriate reference description in Chapter 4 to assure that you entered the correct argument values. Specify an existing workstation number.

Chapter 6

Online Sample Programs

This chapter provides an overview of sample application programs showing how to use the VAXlab Laboratory Graphics Package. These programs are shipped with your VAXlab software kit and are placed online during the VAXlab software installation procedure. You can find the LGP sample programs in a directory with the logical name LGP$EXAMPLES. The logical name of this directory is defined in one of the VAXlab startup command files during installation.

The sample program file names include:

- The facility code, LGP
- A descriptive abbreviation for the LGP routine or task the sample program illustrates
- A file extension indicating the programming language in which each sample program is coded

For example, the sample program LGP_PLOT_3D.FOR shows how to generate a series of bivariate Gaussian curves in three dimensions and is written in VAX FORTRAN.

NOTE

Except where noted in the source files, the LGP sample programs are designed to be run on a VT241 terminal. When these programs run on other terminals, they may resize the axis to fit the display surface. In this case, the program still runs successfully, but displays the following informational message:

%LGP-I-AXIS_RESIZED, axis too big, resized

Table 6-1 lists the sample program names, the routines each sample program uses, and a brief description of what each sample program does.

Table 6-1: LGP Online Sample Programs

Routines Used	Description
	LGP_ASSIGN_PEN.FOR
LGP$MAP_PEN LGP$PLOT LGP$PLOTM	Shows how to plot data on the screen and write the plot to a metafile. The metafile can later be plotted to an output device as illustrated in sample program LGP_PLOT_METAFILE.FOR.
	LGP_AUTOSCALE.FOR
LGP$SCALE	Demonstrates the use of the autoscaling algorithm, LGP$SCALE. Given a set of numbers in an array, this program prints out the value of **fval** and the increment.
	LGP_CHANGE_LOGICALS.FOR
LGP$CHANGE_ PLOTTING_LOGICALS LGP$PLOT	Shows how to redefine the plotting logicals to produce a hardcopy plot on the LVP16.
	LGP_CONTOURM.FOR
LGP$CONTOUR LGP$CONTOURM LGP$TABLE_MODIFY	Shows how to generate a series of bivariate Gaussian curves and plot them in contour form.
	LGP_ERASE.C
LGP$_ERASE LGP$HIST LGP$PLOTM LGP$POINT	Shows how to plot a dataset, erase the dataset leaving the coordinate system, plot points to the graph, erase the points, and plot a histogram.

Table 6–1 (Cont.): LGP Online Sample Programs

Routines Used	Description
	LGP_GET_ROUTINES.FOR
LGP$GET_CHOICE LGP$GET_LOCATOR LGP$GET_STRING LGP$GET_STROKE LGP$GET_VALUE	Demonstrates the use of the five LGP$GET_x routines by showing how to plot a graph and how to add and label points on the graph.
	LGP_GKS.FOR
GKS$FILL_AREA LGP$PLOT GKS$POLYLINE LGP$TABLE_MODIFY	Shows how to plot a histogram using calls to both LGP and GKS.
	LGP_INFO.FOR
LGP$HIST LGP$INFO LGP$PLOT	Shows how to obtain plotting information about a routine that plots a simple histogram.
	LGP_LIO.FOR
LIO$ATTACH LIO$DETACH LGP$PLOT LGP$PLOTC LIO$READ LIO$SET LGP$TABLE_MODIFY	Shows how to use the synchronous user interface (LIO$READ) to read 256 A/D values and plot them using the LGP routines in a continuous real-time display.
	LGP_METAFILE.FOR
LGP$PLOT_METAFILE	Shows how to plot a metafile created by sample program LGP_ASSIGN_PEN.FOR.

Table 6–1 (Cont.): LGP Online Sample Programs

Routines Used	Description
LGP_MOD_ATTRIB.FOR	
LGP$TABLE_MODIFY LGP$TABLE_READ_VALUE	Shows how to read new attribute table values into an array and print out the values.
LGP_MULTIPLOT.FOR	
LGP$HIST LGP$POINT LGP$SET_GRAPH LGP$TABLE_MODIFY	Shows how to use several LGP routines to create multiple plots for simultaneous display on a display terminal screen.
LGP_PLOT_CONTOUR.FOR	
LGP$CONTOUR	Shows how to generate and plot a series of bivariate Gaussian curves in contour form.
LGP_PLOT_HIST.FOR	
LGP$HIST LGP$PLOT	Shows how to plot a histogram, given a set of values.
LGP_PLOT_LOG.C	
LGP$PLOT_LOG	Creates and scales a logarithmic coordinate system based on the data and plots the dataset.
LGP_PLOT_POINTS.C, LGP_PLOT_POINTS.FOR	
LGP$PLOT LGP$POINT	Shows how to plot the function f(x) = exp (0.5*x*k) for k=1,2,3,4, where k=1 (circles), k=2 (squares), k=3 (triangles), and k=4 (stars).
LGP_PLOT_SINE.FOR, LGP_PLOT_SINE.PAS	
LGP$PLOT	Shows how to plot the sine waves of .2π, π, .5π, and .25π.

Table 6–1 (Cont.): LGP Online Sample Programs

Routines Used	Description
	LGP_PLOT_3D.FOR
LGP$3D_SIMPLE	Shows how to generate and plot a series of bivariate Gaussian curves in three dimensions.
	LGP_PLOTC.FOR
LGP$PLOT LGP$PLOTC LGP$TABLE_MODIFY	Shows how to plot the Fourier expansion of a basic periodic function square wave, and produces an approximation of real-time graphical display on your terminal screen.
	LGP_PLOTM_SINE.FOR
LGP$PLOT LGP$PLOTM	Shows how to plot the sine waves of $.2\pi$, π, $.5\pi$, and $.25\pi$.
	LGP_SCALE_LOG.C
LGP$PLOT_LOG LGP$SCALE_LOG	Determines the proper scaling conditions for the x axis and y axis of a plot based on the data. The LGP$PLOT_LOG routine uses the logarithmic coordinate system created by the LGP$SCALE_LOG routine to plot the data.
	LGP_SHADE_CONTOUR.FOR
LGP$CONTOUR_SHADE LGP$TABLE_MODIFY	Shows how to generate a series of bivariate Gaussian curves and plot them in contour form with shading.
	LGP_SIMPLE.C
LGP$PLOT LGP$PLOTM LGP$POINT	Shows how to set up and plot a simple graph.

Table 6-1 (Cont.): LGP Online Sample Programs

Routines Used	Description
	LGP_STNDEV.FOR
LGP$PLOT LGP$POINT LGP$STNDEV	Shows how to plot a set of 10 points with their standard deviation values.
	LGP_3D.FOR
LGP$PLOT_3D LGP$SET_3D_GRAPH	Shows how to plot on multiple surfaces.
	LGP_WRITE_TEXT.FOR
LGP$PUT_TEXT	Shows how to rotate a text string and use a variety of font types and font colors. See Chapter 1 for a copy of this sample program and the output it produces.

Index

A

B

C

M

N

O

P

R

S

T

V

W

Reader's Comments

Guide to the VAXlab Laboratory Graphics Package
AA-KN00C-TE

Your comments and suggestions will help us improve the quality of our future documentation. Please note that this form is for comments on documentation only.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (product works as described)	☐	☐	☐	☐
Completeness (enough information)	☐	☐	☐	☐
Clarity (easy to understand)	☐	☐	☐	☐
Organization (structure of subject matter)	☐	☐	☐	☐
Figures (useful)	☐	☐	☐	☐
Examples (useful)	☐	☐	☐	☐
Index (ability to find topic)	☐	☐	☐	☐
Page layout (easy to find information)	☐	☐	☐	☐

What I like best about this manual: _____

_____

What I like least about this manual: _____

_____

My additional comments or suggestions for improving this manual:

_____

_____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____

Please indicate the type of user/reader that you most nearly represent:

☐ Administrative Support
☐ Computer Operator
☐ Educator/Trainer
☐ Programmer/Analyst
☐ Sales
☐ Scientist/Engineer
☐ Software Support
☐ System Manager
☐ Other (please specify) _____

Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

_____ Phone _____

10/87

Fold Here and Tape

digital™

Please
Affix Stamp
Here

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications
P.O. BOX 1001
MARLBOROUGH, MA 01752–9840

Fold Here

Cut Along Dotted Line