DEC GKS FORTRAN Binding Reference Manual

Order Number: AA–MJ29A–TE

April 1989

This manual is a reference to the DEC GKS FORTRAN Binding functions. It contains information about the DEC GKS FORTRAN Binding control, output, output attribute, transformation, segment, input, inquiry, metafile, and error-handling functions.

Revision/Update Information:	This is a new manual.
Operating System and Version:	VMS Version 4.7 or higher. ULTRIX Version 3.0 or higher. VAXstation requirement: VAXstation Windowing Software Version 3.1 or higher, or DECwindows Version 1.0.
Software Version:	DEC GKS Version 4.0

digital equipment corporation
maynard, massachusetts

First Printing March 1984
Revised November 1984, May 1986, March 1987, and April 1989

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.

The postpaid Reader's Comments forms at the end of this document request your critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

ALL–IN–1	EduSystem	RT
DEC	IAS	ULTRIX
DEC/CMS	MASSBUS	UNIBUS
DEC/MMS	PDP	VAX
DECnet	PDT	VAXcluster
DECmate	P/OS	VMS
DECsystem–10	Professional	VT
DECSYSTEM–20	Q–bus	Work Processor
DECUS	Rainbow	
DECwriter	RSTS	digital™
DIBOL	RSX	

ZK4632

Contents

Chapter 8 Input Functions

Index

Examples

Tables

Preface

Manual Objectives

This manual provides reference to the DEC Graphical Kernel System (GKS) FORTRAN binding functions. DEC GKS is a level 2c GKS implementation. For more information concerning GKS implementation levels, refer to Chapter 1, Introduction.

NOTE

Before reading this manual, you should review the DEC GKS release notes by typing the following:

```
$ HELP  GKS  RELEASE_NOTES RETURN
```

Intended Audience

This manual is intended for experienced application programmers who need to reference information (binding syntax and brief argument descriptions) concerning the DEC GKS FORTRAN binding functions. Readers should be familiar with the FORTRAN programming language and the DIGITAL Command Language (DCL). (For more information concerning DCL, refer to the *VAX/VMS DCL Dictionary*.)

This manual is not tutorial in nature. New users who need tutorial information and moderately experienced users needing programming suggestions should refer to the *DEC GKS User Manual*.

Document Structure

This manual is contained in one volume, with the following information contained in its chapters:

- Chapter 1, Introduction, provides an introduction to the DEC GKS FORTRAN binding, including information about function syntax, data types, passing mechanism, and manual organization.
- Chapter 2, Compiling, Linking, and Running DEC GKS Programs on VMS, provides VMS specific information to compile, link, and edit your programs.
- Chapter 3, Compiling, Linking, and Running DEC GKS Programs on ULTRIX, provides ULTRIX specific information to compile, link, and edit your programs.
- Chapter 4, Control Functions, provides information concerning the establishment of the DEC GKS and workstation environments.
- Chapter 5, Output Functions, provides information concerning the generation of output primitives.
- Chapter 6, Output Attributes, provides information concerning the generation of output attributes.
- Chapter 7, Transformation Functions, provides information concerning the normalization and workstation transformations.
- Chapter 8, Input Functions, provides information concerning input.
- Chapter 9, Segment Functions, provides information concerning the storage of output primitives in segments.
- Chapter 10, Metafile, Error, and Utility Functions, provides information concerning long-term storage of graphical images, error-handling, and utility functions.
- Chapter 11, Inquiry Functions, provides information concerning the acquisition of DEC GKS and workstation status information.
- Appendix A, DEC GKS Function Names and FORTRAN Binding Function Names, provides information concerning all DEC GKS FORTRAN and DEC GKS$ function name cross references.
- Appendix B, DEC GKS FORTRAN Constants, provides information concerning all DEC GKS FORTRAN constants.
- Appendix C, DEC GKS Error Messages, provides information concerning all DEC GKS FORTRAN error messages.

Associated Documents

You may find the following documents useful when using DEC GKS:

- *DEC GKS User Manual*—For programmers who need tutorial information or guides to programming technique.
- *DEC GKS Reference Manual*—For programmers who need encyclopedic information about GKS functions.
- *DEC GKS GKS$ Binding Reference Manual*—For programmers who need specific syntax and argument descriptions for the GKS$ Binding.
- *DEC GKS C Binding Reference Manual*—For programmers who need specific syntax and argument descriptions for the C Binding.
- *Building a DEC GKS Workstation Handler System*—For programmers who need to build a DEC GKS workstation graphics handler.
- *Building a DEC GKS Device Handler System*—For programmers who need to provide support for a device unsupported by the DEC GKS graphics handlers.
- *DEC GKS Device Specifics Reference Manual*—For programmers who need device-specific information.
- *DEC GKS Installation Guide*—For system managers who install the DEC GKS software, including the Run-Time installation, on either VMS or ULTRIX operating systems.

Conventions

Convention	Meaning
RETURN	The symbol RETURN represents a single stroke of the RETURN key on a terminal.
$ RUN GKSPROG RETURN	In interactive examples, the user's response to a prompt is printed in red; system prompts are printed in black.
INTEGER X . . . X = 5	A vertical ellipsis indicates that not all of the text of a program or program output is illustrated. Only relevant material is shown in the example.

Convention	Meaning
option, . . .	A horizontal ellipsis indicates that additional arguments, options, or values can be entered. A comma that precedes the ellipsis indicates that successive items must be separated by commas.
[output-source, . . .]	Square brackets, in function synopses and a few other contexts, indicate that a syntactic element is optional.

Chapter 1

Introduction

The Graphical Kernel System (GKS) is a set of graphics functions that can be used by numerous types of graphics applications to produce two-dimensional pictures on graphics output devices. GKS is defined by the ANSI X3.124-1985 and the ISO 7942-1985 standards. DEC GKS adheres to both standards. When this manual refers to the GKS standard, the reference applies to both standards.

The FORTRAN binding provides the GKS functions and data types in both a natural and efficient manner using FORTRAN facilities, without violating the style or design philosophy of FORTRAN.

1.1 GKS Levels

The GKS standard defines levels of a GKS implementation that address the most common classes of graphic devices and application needs. The levels are determined primarily by input and output capabilities. The output level values are represented by the characters m, 0, 1, and 2. The input level values are represented by the characters a, b, and c.

The DEC GKS software is a level 2c implementation, incorporating all of the GKS output capabilities (level 2) and all of the input capabilities (level c). This manual uses the term DEC GKS when describing the 2c level DEC GKS product.

1.2 Programming Considerations

The specific method for using DEC GKS software depends on the features and conventions of each VAX language. This section discusses general issues that must be considered when using any VAX language with DEC GKS.

NOTE

Some of the VAX languages have language-specific requirements for using DEC GKS. For a complete discussion, you should refer to Appendix F, Language-Specific Programming Information, in the *DEC GKS Reference Manual*, before coding your programs. For a discussion of the capabilities of each of the DEC GKS supported physical devices, refer to the appropriate device-specific chapter in the *DEC GKS Device Specifics Reference Manual*.

1.2.1 Online Help

DEC GKS provides an online HELP library. To access this information, type the following:

```
$ HELP GKS RETURN
```

Before using the DEC GKS software, you should review the release notes for information pertinent to the current release. To review the release notes, type the following:

```
$ HELP GKS RELEASE_NOTES RETURN
```

1.2.2 Supported Languages

DEC GKS supports the following languages:

- VAX FORTRAN
- VAX C
- CC (Portable C)

1.2.3 Capabilities of Supported Devices

In many applications, you may wish to write completely device-independent programs. In this way, you can run your programs using different devices without having to rewrite your programs. The *DEC GKS User Manual* outlines the procedure for device-independent programming using DEC GKS.

However, you may wish to review the range of capabilities of the DEC GKS supported devices, or you may wish to write device-dependent subroutines within your application. In any instance, it is helpful to review the device-specific appendixes in this manual before you begin coding your application. The device-dependent appendixes contain information concerning predefined bundle index representations, color capabilities, initial input values, bit masks as workstation type values, supported escape functions for that particular device, and similar information.

1.2.4 Calling Sequences

Each DEC GKS function requires a specific calling sequence. The calling sequence indicates the elements included in the language statement that calls the function, and the order of those elements. The three elements are the following:

- **Call Type**
  High-level VAX languages call DEC GKS functions with CALL statements or function references. For example, when using FORTRAN, you can use a CALL statement to call DEC GKS functions.
- **Function Identifier**
  All DEC GKS function names begin with the prefix GKS$. FORTRAN binding names begin with the uppercase letter G, and C binding names begin with a lowercase g. The remainder of the name indicates the operation performed by the function. Refer to the *DEC GKS C Binding Reference Manual* for information concerning the C binding function names.
- **Argument List**
  Arguments that are passed to DEC GKS functions must be listed in the order shown in the syntax descriptions contained in this manual. See Section 3.2 for more information concerning the function description format used in this manual. The various language binding functions may have an argument list that is different from the corresponding GKS$ function.

1.2.5 Constants and Include Files

DEC GKS constants are symbolic names that are syntactically equivalent to literal integer constants. These constants are used in the following ways:

- As arguments to DEC GKS functions.
- As literal values to which you can compare a returned value from an inquiry function (for example, you can compare the return value, from a call to the function GKS$INQ_WS_TYPE, to the constant GKS$K_VT125).
- As literal completion status codes to which you can compare a function return value.

NOTE

Constants (defines) for the bindings are in the binding specific include files.

Many DEC GKS functions use constants as arguments, as shown by the following function call:

```
GKS$CLEAR_WS( 1, GKS$K_CLEAR_ALWAYS)
```

You can compare one of the completion status codes to a function return value, as follows:

```
   .
   .
   .
IF ( GKS$_SUCCESS = GKS$ACTIVATE_WS( 1 ) )
   .
   .
   .
```

Most DEC GKS constants begin with the prefix GKS$K and are defined in a definition file. All DEC GKS completion status code constants begin with the prefix GKS$_ERROR_ or DECGKS$_ERROR_NEG_ and are defined in a separate definition file. All DEC GKS bit mask constants begin with the prefix GKS$M_.

You can either specify a literal value as an argument to a DEC GKS function, or you can include the language definition files and use a defined constant name instead. The use of constants adds to program legibility and program documentation.

To review the list of DEC GKS constants, refer to Appendix B, DEC GKS Constants, in the *DEC GKS Reference Manual*. To review the list of DEC GKS completion status code constants, refer to Appendix C, DEC GKS Error Messages in this manual.

1.2.6 Returning a Data Record

The DEC GKS FORTRAN binding does not return data records. This restriction conforms with the GKS Standard. Use the GKS$ function with FORTRAN if you want to return the data record.

1.2.7 Packing and Unpacking Records

In the FORTRAN binding, a data record is defined to be an array containing character strings of 80 characters (data type CHARACTER*80). The binding functions use an integer value to determine the dimension of the array of strings. When using a data record as an output argument, the dimension of the data record is needed as a separate argument. In this manner, DEC GKS determines how much space exists in which to write the data record. DEC GKS places the length of the data actually written to the data record in another argument.

Depending on the needs of the data record, you may need to store information of several data types (integer, real, character string) in a data record. The FORTRAN binding provides the functions GPREC and GUREC to pack this information into the array of character strings, and to unpack the information so that you can place the data into variables of appropriate data types.

Simply, you pass the integers in one array, the real numbers in another array, and an array of character string lengths in a third array, to GPREC, and the function packs and writes the information to a data record. The function GUREC accepts a data record, and then unpacks the information into an array of integers, an array of real numbers, and an array of character string lengths.

Example 1–1 shows how you pack a data record to initialize a choice input device with a prompt and echo type of 3, and how to unpack the information written in the output data record.

Example 1–1: Packing and Unpacking a Data Record

```
C     This program packs a data array for a choice device.
      IMPLICIT NONE

C     Notice that you need to include a different definitions file
C     when you use the FORTRAN binding.
      INCLUDE 'SYS$LIBRARY:GKSDEFS.BND'
      INTEGER WS_ID, PROMPT_ECHO_TYPE, ERROR_STATUS, INPUT_MODE,
     * ECHO_FLAG, XFORM, MAX_DATA, DATA_WRITTEN, INPUT_STATUS,
     * INITIAL_CHOICE, NUM_CHOICES, EMPTY_INT( 1 ), DEVICE_NUM,
     * ZERO, NUM_STRINGS, OP_MODE, LENGTH_STRINGS( 3 ), INPUT_CHOICE
      REAL EMPTY_REAL( 1 ), ECHO_AREA( 4 )
      CHARACTER*80 DATA_RECORD( 10 ), STRINGS( 3 ),
     * STRINGS_2( 3 )
      DATA WS_ID / 1 /, ZERO / 0 /, DEVICE_NUM / 1 /,
     * NUM_STRINGS / 3 /, PROMPT_ECHO_TYPE / 1 /

      CALL GOPKS( 6 )
      CALL GOPWK( WS_ID, GCONID, GV240C)
      CALL GACWK( WS_ID )

C     DATA_RECORD can only hold ten character strings.
      MAX_DATA = 10

      CALL GQCHS( WS_ID, DEVICE_NUM, MAX_DATA, ERROR_STATUS,
     * OP_MODE, ECHO_FLAG, INPUT_STATUS, INITIAL_CHOICE, PROMPT_ECHO_TYPE,
     * ECHO_AREA, DATA_WRITTEN, DATA_RECORD )

C     Create the strings and pack and new data record.
      STRINGS( 1 ) = 'The'
      STRINGS( 2 ) = 'Unforgettable'
      STRINGS( 3 ) = 'Fire'
      LENGTH_STRINGS( 1 ) = LEN( 'The' )
      LENGTH_STRINGS( 2 ) = LEN( 'Unforgettable' )
      LENGTH_STRINGS( 3 ) = LEN( 'Fire' )

(1)   CALL GPREC( ZERO, EMPTY_INT, ZERO, EMPTY_REAL,
     * NUM_STRINGS, LENGTH_STRINGS, STRINGS, MAX_DATA,
     * ERROR_STATUS, DATA_WRITTEN, DATA_RECORD )

C     Initialize.
      CALL GINCH( WS_ID, DEVICE_NUM, INPUT_STATUS,
     * INITIAL_CHOICE, PROMPT_ECHO_TYPE, ECHO_AREA( 1 ),
     * ECHO_AREA( 2 ), ECHO_AREA( 3 ), ECHO_AREA( 4 ),
     * DATA_WRITTEN, DATA_RECORD )

C     Set mode.
      CALL GSCHM( WS_ID, DEVICE_NUM, OP_MODE, ECHO_FLAG )

C     Request input.
      CALL GRQCH( WS_ID, DEVICE_NUM, INPUT_STATUS,
     * INPUT_CHOICE )
```

(continued on next page)

Example 1–1 (Cont.): Packing and Unpacking a Data Record

```
C     Unpack the data record into a new variable.
❷     CALL GUREC( DATA_WRITTEN, DATA_RECORD, ZERO, ZERO,
     * NUM_STRINGS, ERROR_STATUS, ZERO, EMPTY_INT, ZERO,
     * EMPTY_REAL, NUM_STRINGS, LENGTH_STRINGS, STRINGS_2 )

C     Check to see if the strings were unpacked into the new variable.
      WRITE(6,*) STRINGS_2( 1 )
      WRITE(6,*) STRINGS_2( 2 )
      WRITE(6,*) STRINGS_2( 3 )

      CALL GDAWK( WS_ID )
      CALL GCLWK( WS_ID )
      CALL GCLKS()
      END
```

The following numbers correspond to the numbers in the previous example:

❶ This call to GPREC packs a data record for a choice input device. As arguments to this function, you pass an array with zero integers, an array with zero real numbers, and an array with three character strings to be used to label the choices. The description of GINCH in this chapter specifies the different types of acceptable data records for a choice input device.

❷ You normally do not need to unpack a data record as this example illustrates, but this call to the function GUREC illustrates the process. After the program requests input, the unpacked character strings are written to the screen.

After the program requests choice input, it writes the following character strings to the screen.

```
$ RUN EXAMPLE_1 RETURN
The
Unforgettable
Fire
```

NOTE

The input data record component descriptions require knowledge of the GKS standard data record format. You can review the GKS standard data records by referring to Chapter 8, Input Functions, in the *DEC GKS Reference Manual*. For information concerning which devices support which prompt and echo types, refer to the *DEC GKS Device Specifics Reference Manual*.

1.3 Binding Function Names

The FORTRAN binding specifies a set of standard function identifiers. Each of the DEC GKS function identifiers that use the GKS$ interface correspond to one or more of the FORTRAN binding identifiers. In Appendix A, Table A–1 matches the DEC GKS$ function names to the FORTRAN binding function names.

Note that all of the names begin with the letter G. If you are writing a GKS program that uses the FORTRAN binding, avoid using the letter G as the first letter of the name of any of your own subroutines, functions, variables, or the name of your main program.

The FORTRAN binding names abbreviate the names commonly found in the GKS standard. For instance, all functions begin with the letter G (for GKS); the FORTRAN binding abbreviation for ACTIVATE is AC; the binding abbreviation for WORKSTATION is WK. Consequently, the FORTRAN binding function name for ACTIVATE WORKSTATION is GACWK.

The GKS FORTRAN binding includes two functions that do not correspond to DEC GKS functions: GPREC and GUREC. The functions GPREC and GUPREC pack and unpack data records for use by the FORTRAN binding functions, in particular the input functions. For detailed information concerning GPREC and GUPREC, refer to Section 1.2.7.

1.4 FORTRAN Binding Function Syntax

The following command section lists the following for each FORTRAN binding function:

- FORTRAN binding function descriptive name
- Valid DEC GKS operating states during which you can use the function
- Function syntax listing
- Brief description of the arguments

The argument description identifies the data type, indicates whether the argument is an input or output parameter, and provides a brief description of the argument. To identify the data type, the argument descriptions use the following:

- Integer—to specify an INTEGER*4
- Real—to specify a REAL*4

- String—to specify a CHARACTER*80
- Strdat—to specify a CHARACTER*80 (LENGTH), where (LENGTH) is the actual dimension of the array of strings

An I or O indicates an input or output argument, respectively. An M indicates a modifiable argument.

All arguments are passed by reference unless the description specifies that the argument is passed by descriptor.

Appendix C lists DEC GKS error messages, the DEC GKS error number, the VMS completion status codes, and appropriate user action to correct the error.

1.5 FORTRAN Binding Function Organization

The FORTRAN binding functions are divided into functional categories that correspond to the following chapters:

- Control functions
- Output functions
- Attribute functions
- Transformation functions
- Input functions
- Segment functions
- Metafile functions
- Error-handling functions
- Inquiry functions
  - GKS description table functions
  - Workstation description table functions
  - GKS state list functions
  - Workstation state list functions
  - Segment functions
  - Pixel functions

For quick reference, you can use either of the following means to locate a FORTRAN binding function description.

- If you know the functional category of the function you want to find, go to that chapter or section and look for the descriptive name in alphabetical order.

  For example, you know that GOPWK is in the control category. Go to the Control Functions section and look for Open Workstation in alphabetical order, or use the Table of Contents and look up Open Workstation under Control Functions to the page number.

  Appendix A lists the DEC GKS functions according to category and includes the corresponding FORTRAN binding function names.
- Look up the function in the index.

  If you are not sure which functional category a function belongs to, you can look up either the binding name (for example, GOPWK) or the descriptive name (Open Workstation) in the index for page number.

1.5.1 Standard Escape/GDP Data Records

When calling the functions GESC or GGDP (generalized drawing primitive), you may need to pass a data record. DEC GKS has a standard escape/GDP data record that contains up to three integer components and four array addresses.

The DEC GKS standard escape/GDP data record is as follows:

Position	Data Type	Description
1	Integer	Number of integer values passed in the data record.
2	Integer	Number of real values passed in the data record.
3	Integer	Number of string addresses passed in the data record.
4	Integer (address)	Address of array of integers with exactly as many elements as the number specified in component number 1.
5	Integer (address)	Address of array of real numbers with exactly as many elements as the number specified in component number 2.
6	Integer (address)	Address of array of string lengths with exactly as many elements as the number specified in component number 3.
7	Integer (address)	Address of array of string addresses with exactly as many elements as the number specified in component number 3.

For the FORTRAN binding, you will not directly create a data record of this format. Since a data record is defined to be an array containing character strings of 80 characters, it must first be packed using the GPREC function, as described in Section 1.2.7. This packed record will be input to GESC, which will then create a data record of the above format, hereafter referred to as the *DEC GKS data record.*

To use an escape or GDP data record, you need to perform the following tasks:

1. Find the GDP or escape description in Appendix I, DEC GKS GDPs and Escapes, in the *DEC GKS Reference Manual.*
2. Determine the contents of the DEC GKS data record, if one is required, and use GPREC to pack the data record.

   The correspondence between the GPREC parameters and the DEC GKS data record is as follows:

GPREC parameter	DEC GKS data record
i_entries	Position 1
int_array	Position 4
r_entries	Position 2
real_array	Position 5
st_entries	Position 3
st_lengths	Position 6
str_array	Position 7

3. GPREC provides output parameters describing the packed data record and the number of dimensions of the packed data record. Use these parameters as input parameters to the GESC or GGDP functions.

After performing their task, some escape functions pass information back to you by use of an output data record. This output data record is also a packed data record. You may wish to use the GUREC function to unpack those values. Each of the device-specific chapters in the *DEC GKS Device Specifics Reference Manual* describes its level of support for the various escapes and GDPs provided by DEC GKS. For more information, refer to Appendix I, DEC GKS GDPs and Escapes, in the *DEC GKS Reference Manual*, or to the appropriate device specific chapter.

NOTE

Remember that the DEC GKS input data records have a format that is completely different from the DEC GKS standard escape/GDP data record format. To review the GKS standard input data records, refer to Chapter 8, Input Functions, in the *DEC GKS Reference Manual*. To review the actual data records required by the DEC GKS graphics handlers, refer to Appendix J, DEC GKS Specific Input Values, in the *DEC GKS Reference Manual*.

Chapter 2

Compiling, Linking, and Running DEC GKS Programs on VMS

The DEC GKS functions that begin with the prefix GKS$ are designed to be used on one of Digital's systems. Those functions meet the *functional* GKS standard. In other words, they perform the necessary tasks as designated by the GKS standard.

However, these functions are in no way meant to meet a *syntactical* standard. For instance, the DEC GKS function GKS$CELL_ARRAY might have a different number of arguments than the cell array function in another GKS implementation. As a result, programs written using the GKS$ interface are not easily transportable; you have to edit the function names, and quite possibly the number and order of function arguments.

To provide a syntactical standard so that programs can be transportable between GKS implementations, the GKS standard defines the FORTRAN binding for users of the FORTRAN language. The FORTRAN binding standardizes the GKS function names, and the number and order of function parameters. Consequently, you can use a program written using the DEC GKS FORTRAN binding on any GKS implementation.

2.1 Compiling, Linking, and Running

To use the FORTRAN binding, you compile and execute programs using the FORTRAN binding functions just as you would any other program. Use the compile command that is appropriate for FORTRAN, and use the RUN command to execute the image.

To link your programs that use the FORTRAN binding, you need to use the following command line:

```
$ LINK  file_name.OBJ,  SYS$LIBRARY:GKSFORBND/LIBRARY[RETURN]
```

In the example, you link your DEC GKS program with the object library containing the DEC GKS FORTRAN binding function code.

If you choose to make reference to the FORTRAN binding constant values, you need to include the appropriate definitions file into your source program. There is a separate definitions file for the GKS$ constants and the FORTRAN binding constants. For specific information concerning the inclusion of definition files, refer to Chapter 1, Introduction to DEC GKS, in the *DEC GKS Reference Manual*. To review the list of FORTRAN binding constants, refer to Appendix B, DEC GKS Constants, in the *DEC GKS Reference Manual*.

If you create a user-defined error handler, you need to link your program differently. See Section 2.2 for more information.

2.2 Using User-Defined Error-Handling Functions

If you choose to handle errors in a different manner from the way in which the GKS FORTRAN binding handles errors, you can replace the function GERHND with a function of your own. In this way, you can control data storage, message generation, and execution of an error-handler in the manner most suited to your application.

If you do replace GERHND with an external function of your own, you need to link your object files as follows:

```
$ LINK  file_name.OBJ, error_handler.OBJ, SYS$LIBRARY:GKSFORBND/LIBRARY[RETURN]
```

The error handler file name must appear before the FORTRAN binding library in the LINK command line.

When working with the error-handling functions, you may need to specify the name of the FORTRAN binding function that caused the error. The error-handling functions specify that the function identifiers must be integers.

The FORTRAN binding defines a series of identifiers, which are defined to be an integer value, that correspond to each of the FORTRAN binding function names. To specify the FORTRAN binding function identifier, specify the name of the FORTRAN binding function name, but replace the initial letter G (for GKS) with the letter E (for Error).

For example, to specify the function GOPWK (OPEN WORKSTATION) as the function that generated the error, pass the function identifier EOPWK as an argument, as follows:

```
          .
          .
          .
C     Log an error generated by the function GOPWK.
      GERLOG( error_number, EOPWK, error_file )
          .
          .
          .
```

Chapter 3

Compiling, Linking, and Running DEC GKS Programs on ULTRIX

The DEC GKS functions that begin with the prefix GKS$ are designed to be used on a DIGITAL system. Those functions meet the *functional* GKS standard. In other words, they perform the necessary tasks as designated by the GKS standard.

However, these functions are in no way meant to meet a *syntactical* standard. For instance, the DEC GKS function GKS$CELL_ARRAY might have a different number of arguments than the cell array function in another GKS implementation. As a result, programs written using the GKS$ interface are not easily transportable; you have to edit the function names, and quite possibly the number and order of function arguments.

Use the FORTRAN binding, and approved ISO and ANSI standards, for tranportability.

3.1 ULTRIX Programming Considerations

The specific method for using DEC GKS software depends on the features and conventions of each VAX language. This section discusses general issues that must be considered when using any VAX language with DEC GKS. For a discussion of the capabilities of each of the DEC GKS supported physical devices, refer to the appropriate device-specific chapter in the *DEC GKS Device Specifics Reference Manual.*

3.2 Including Definition Files

You use DEC GKS software primarily by placing calls to DEC GKS functions in your program. However, when using DEC GKS, you need statements in your program other than calls to GKS functions. The specific statements that are needed depend on the VAX language you use. (For more information, refer to Appendix F, Language-Specific Programming Information, in the *DEC GKS Reference Manual.*)

DEC GKS constants and their values must be made available to all programs using DEC GKS regardless of the language you use. All high-level languages that use DEC GKS have a method for inserting an external file into the program source code stream at compile time. Incorporating an external file is the method for making DEC GKS constants available.

Your installation kit has been supplied with files that contain DEC GKS constants and separate files that contain DEC GKS completion status code constants. You incorporate these files into your program with a statement that is appropriate to the language you are using.

Any FORTRAN program that uses the FORTRAN binding functions should contain the following statement:

```
INCLUDE '/usr/include/GKS/gksdefs.bnd'
```

The language definition files located in /usr/include/GKS are as follows:

- gksdefs.h for VAX C and CC (GKS$ binding)
- gks.h for VAX C and CC (C binding)
- gksdefs.bnd for VAX FORTRAN using the FORTRAN binding functions

The completion status code definition files located in usr/include/GKS are as follows:

- gksmsgs.h for VAX C

Each file includes comments that describe the exact method for using a given definition file.

3.2.1 Compiling and Linking FORTRAN Programs

A program that uses DEC GKS function calls should be compiled and executed as any other program. Use the compile command that is appropriate to the language you are using. To run an executable program, type the executable file name that you specified.

To compile and link a DEC GKS FORTRAN binding program, use the following syntax:

```
fort -o application application.for\ RETURN
   -lGKSFORBND -lddif -dwt -lcursesX -lc -lXll -lm -lc RETURN
```

NOTE

The \ RETURN convention indicates that you type the backslash character \, press Return, and then type text on the next line of the screen.

3.2.2 Environment Variables and DEC GKS Programming

In many DEC GKS programs, the execution of your application appears as follows:

```
❶       CALL GKS$OPEN_GKS( stderr )

❷       CALL GKS$OPEN_WS( 1, GKS$K_CONID_DEFAULT,
       * GKS$K_WSTYPE_DEFAULT )

        CALL GKS$ACTIVATE_WS( 1 )
           .
           .
           .
  C     Release the DEC GKS and workstation environments.
        CALL GKS$DEACTIVATE_WS( 1 )
        CALL GKS$CLOSE_WS( 1 )
        CALL GKS$CLOSE_GKS()
```

The following numbers correspond to the numbers in the previous example:

❶ In this call to GKS$OPEN_GKS, the name stderr is the only argument to the function. This argument tells DEC GKS where to write generated error messages.

If you pass the name stderr (or the value 0), DEC GKS writes the error messages to the specified location. By default, stderr goes to the device /dev/tty, which translates to your process' default device connection (error messages appear on your terminal's display surface).

If you choose, you can specify a path name as an argument to GKS$OPEN_GKS. In this way, you have a permanent record of generated error messages for use during program debugging.

❷ The constant GKS$K_CONID_DEFAULT (or the value 0) tells DEC GKS to evaluate the environment variable GKSconid in order to determine the name of the device connection. The constant GKS$K_WSTYPE_DEFAULT tells GKS to evaluate the environment variable GKSwstype in order to determine the name of the workstation type.

Consequently, you can use the setenv command to your shell to define the environment variables to be the connection and type with which you are working, as follows:

```
csh> setenv  GKSconid  /dev/tty RETURN
csh> setenv  GKSwstype  13
csh> # VT241 Color RETURN
csh> application RETURN
   .
   .
   .
csh> setenv  GKSconid  /dev/tt00 RETURN
csh> setenv  GKSwstype  12
csh> # VT125 Black and White RETURN
csh> application RETURN
   .
   .
   .
```

There may be times when you do not wish to define the DEC GKS environment variables. In this case, or if you define an invalid value, DEC GKS translates several environment variables in the following order:

1. If the environment variable GKSconid is undefined, DEC GKS uses /dev/tty for output.
2. If the environment variable GKSwstype is undefined, then DEC GKS sets the device type to be GKS$K_VT240BW (the value 14, a black and white VT240).

The ability to define GKSconid and GKSwstype provides device independency. For more information concerning device-independent DEC GKS programs, refer to the *DEC GKS User Manual*.

3.2.2.1 Specifying Bit Masks as Workstation Type Values

You have the option of specifying the workstation type value in either a hexadecimal, octal, or decimal longword value. In most cases, it is sufficient to specify the type value in decimal.

However, some of the DEC GKS supported devices allow you to pass a *bit mask* in the first word of the longword workstation type value. For example, the following workstation type specifies default values for the DIGITAL LVP16 plotter:

```
csh> setenv  GKSwstype   51 RETURN
```

The following decimal workstation type specifies to DEC GKS to use the LVP16 plotter in landscape mode, with a paper size of 11 x 17:

```
csh> setenv  GKSwstype   %x131123 RETURN
```

For a complete list of all of the available bit masks for a given device, refer to the *DEC GKS Device Specifics Reference Manual.*

Chapter 4

Control Functions

The control functions establish the DEC GKS and workstation environments, and control the workstation surface in a variety of ways. The following list presents the control functions by category:

Category	GKS Functions
GKS Environment	GOPKS, GCLKS
Workstation Environment	GOPWK, GACWK, GDAWK, GCLWK
Display Surface Control	GCLRWK, GRSGWK, GSDS, GUWK
Additional Control	GESC, GMSG, GMSGS

ACTIVATE WORKSTATION

Operating States: GWSOP, GWSAC

Syntax

GACWK *(workstation_id)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier

CLEAR WORKSTATION

Operating States: GWSOP, GWSAC

Syntax

GCLRWK *(workstation_id, control_flag)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
control_flag	integer	I	Control flag (GCONDI, GALWAY)

CLOSE GKS

Operating States: GGKOP

Syntax

GCLKS *( )*

CLOSE WORKSTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GCLWK *(workstation_id)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier

DEACTIVATE WORKSTATION

Operating States: GWSAC

Syntax

GDAWK *(workstation_id)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier

ESCAPE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GESC *(fun_id, dim_idr, idr, max_odr, len_odr, odr)*

Arguments

Argument	Data Type	I/O	Description
fun_id	integer	I	Function identifier
dim_idr	integer	I	Dimension of the input data record
idr	strdata	I	Input data record
max_odr	integer	I	Maximum size of the output data record, in bytes
len_odr	integer	O	Number of array elements in the output data record
odr	strdata	O	Output data record

NOTE

If you pass the value 0 as the argument max_odr (maximum size of the output data record), the function GESC checks for errors, returns the size of the output data record in max_odr (in bytes), but does *not* perform the escape. You can use this functionality to check the size of the returned output data record (in bytes) without performing the escape action.

MESSAGE (FORTRAN 77)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GMSG *(workstation_id, message)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
message	string	I	Text of the message

MESSAGE (FORTRAN 77 subset)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GMSGS *(workstation_id, l_message, message)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
l_message	integer	I	Length of the message in characters, in bytes
message	string	I	Text of the message

OPEN GKS

Operating States: GGKCL

Syntax

GOPKS *(err_file [, buffer])*

Arguments

Argument	Data Type	I/O	Description
err_file	integer	I	Unit number of the error file.
buffer	integer	I	Amount of memory units available to DEC GKS (optional argument).

OPEN WORKSTATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GOPWK *(workstation_id, con_id, workstation_type)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
con_id	integer	I	Connection identifier
workstation_ type	integer	I	Workstation type

REDRAW ALL SEGMENTS ON WORKSTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRSGWK *(workstation_id)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier

SET DEFERRAL STATE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSDS *(workstation_id, def_mode, reg_mode)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
def_mode	integer	I	Deferral mode (GASAP, GBNIG, GBNIL, GASTI)
reg_mode	integer	I	Implicit regeneration mode (GSUPPD, GALLOW)

UPDATE WORKSTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GUWK *(workstation_id, reg_flag)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
ref_flag	integer	I	Implicit regeneration flag (GPOSTP, GPERFO)

Chapter 5

Output Functions

The DEC GKS output functions generate the basic components, or **primitives**, of all graphical pictures. The output functions are divided into the following categories:

Category	GKS Functions
Draw connected lines.	GPL
Mark locations with symbols.	GPM
Draw text.	GTX, GTXS
Fill a polygon.	GFA
Color cells of a rectangle.	GCA
Draw generalized drawing primitive.	GGDP

CELL ARRAY

Operating States: GWSAC, GSGOP

Syntax

GCA *(spx, spy, dpx, dpy, dim_x, dim_y, scol, srow, ncols, nrows, cindex)*

Arguments

Argument	Data Type	I/O	Description
spx	real	I	Starting point X value in world coordinates
spy	real	I	Starting point Y value in world coordinates
dpx	real	I	Diagonal point X value in world coordinates
dpy	real	I	Diagonal point Y value in world coordinates
dim_x	integer	I	X dimension of the cell array
dim_y	integer	I	Y dimension of the cell array
scol	integer	I	Starting column of the color array
srow	integer	I	Starting row of the color array
ncols	integer	I	Number of color array columns to traverse
nrows	integer	I	Number of color array rows to traverse
cindex	integer	I	Two-dimensional color index array dim_x by dim_y

FILL AREA

Operating States: GWSAC, GSGOP

Syntax

GFA *(number_of_points, px, py)*

Arguments

Argument	Data Type	I/O	Description
number_of_ points	integer	I	Number of points
px	real	I	Array of X world coordinate points of dimension npoints
py	real	I	Array of Y world coordinate points of dimension npoints

GENERALIZED DRAWING PRIMITIVE

Operating States: GWSAC, GSGOP

Syntax

GGDP *(number_of_points, px, py, gdp_id, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
number_of_ points	integer	I	Number of points
px	real	I	Array of X world coordinate points of dimension npoints
py	real	I	Array of Y world coordinate points of dimension npoints
gdp_id	integer	I	GDP identifier
dim_dr	integer	I	Dimension of the data record
dr	strdata	I	The data record

POLYLINE

Operating States: GWSAC, GSGOP

Syntax

GPL *(number_of_points, px, py)*

Arguments

Argument	Data Type	I/O	Description
number_of_ points	integer	I	Number of points
px	real	I	Array of X world coordinate points of dimension npoints
py	real	I	Array of Y world coordinate points of dimension npoints

POLYMARKER

Operating States: GWSAC, GSGOP

Syntax

GPM *(number_of_points, px, py)*

Arguments

Argument	Data Type	I/O	Description
number_of_ points	integer	I	Number of points
px	real	I	Array of X world coordinate points of dimension npoints
py	real	I	Array of Y world coordinate points of dimension npoints

TEXT (FORTRAN 77)

Operating States: GWSAC, GSGOP

Syntax

GTX *(px, py, text)*

Arguments

Argument	Data Type	I/O	Description
px	real	I	X starting position in world coordinates
py	real	I	Y starting position in world coordinates
text	string	I	The text string

TEXT (FORTRAN 77 subset)

Operating States: GWSAC, GSGOP

Syntax

GTXS *(px, py, ltext, text)*

Arguments

Argument	Data Type	I/O	Description
px	real	I	X starting position in world coordinates
py	real	I	Y starting position in world coordinates
ltext	integer	I	Length of the text string (bytes) in characters
text	string	I	The text string

Chapter 6

Output Attributes

The DEC GKS output attribute functions affect the appearance of generated output primitives. The following list presents the output attribute functions by category:

Category	GKS Functions
Fill Area Attributes	GSFACI, GSFAI, GSFAIS, GSFASI, GSPARF, GSPA
Polyline Attributes	GSPLCI, GSPLI, GSLN, GSLWSC
Polymarker Attributes	GSPMCI, GSPMI, GSMKSC, GSMK
Text Attributes	GSTXAL, GSTXCI, GSCHXP, GSTXFP, GSCHH, GSTXI, GSTXP, GSCHSP, GSCHUP
Aspect Source Flags	GSASF
Representations	GSCR, SFAR, GSPAR, GSPLR, GSPMR, GSTXR

SET ASPECT SOURCE FLAGS

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSASF *(flags)*

Arguments

Argument	Data Type	I/O	Description
flags	integer	I	Array of length 13 containing the aspect source flags (GBUNDL, GINDIV), as follows: 1. Linetype 2. Linewidth scale factor 3. Polyline color index 4. Marker type 5. Marker size scale factor 6. Polymarker color index 7. Text font and precision 8. Character expansion factor 9. Character spacing 10. Text color index 11. Fill area interior style 12. Fill area style index 13. Fill area color index

SET CHARACTER EXPANSION FACTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSCHXP *(efactor)*

Arguments

Argument	Data Type	I/O	Description
efactor	real	I	Character expansion factor

SET CHARACTER HEIGHT

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

NOTE

This function is GKS$SET_TEXT_HEIGHT in the *DEC GKS Reference Manual*, also referred to as SET TEXT HEIGHT.

Syntax

GSCHH *(height)*

Arguments

Argument	Data Type	I/O	Description
height	real	I	Character height in world coordinates

SET CHARACTER SPACING

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

NOTE

This function is called SET TEXT SPACING in the *DEC GKS Reference Manual*.

Syntax

GSCHSP *(spacing)*

Arguments

Argument	Data Type	I/O	Description
spacing	real	I	Character spacing

SET CHARACTER UP VECTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

NOTE

This function is called SET TEXT UP VECTOR in the *DEC GKS Reference Manual.*

Syntax

GSCHUP *(xvector, yvector)*

Arguments

Argument	Data Type	I/O	Description
xvector	real	I	X world coordinate vector value
yvector	real	I	Y world coordinate vector value

SET COLOR REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSCR *(workstation_id, cindex, red_i, green_i, blue_i)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
cindex	integer	I	Color index
red_i	real	I	Red intensity
green_i	real	I	Green intensity
blue_i	real	I	Blue intensity

SET FILL AREA COLOR INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSFACI *(cindex)*

Arguments

Argument	Data Type	I/O	Description
cindex	integer	I	Color index

SET FILL AREA INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSFAI *(index)*

Arguments

Argument	Data Type	I/O	Description
index	integer	I	Fill area index

SET FILL AREA INTERIOR STYLE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSFAIS *(style)*

Arguments

Argument	Data Type	I/O	Description
style	integer	I	Fill area interior style (GHOLLO, GSOLID, GPATTR, GHATCH)

SET FILL AREA REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSFAR *(workstation_id, index, style, sindex, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
index	integer	I	Fill area index value
style	integer	I	Interior style (GHOLLO, GSOLID, GPATTR, GHATCH)
sindex	integer	I	Style index
cindex	integer	I	Fill area color index

SET FILL AREA STYLE INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSFASI *(sindex)*

Arguments

Argument	Data Type	I/O	Description
sindex	integer	I	Fill area style index

SET LINETYPE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSLN *(ltype)*

Arguments

Argument	Data Type	I/O	Description
ltype	integer	I	Line type (GLSOLI, GLDASH, GLDOT, GLDASD). Nonstandard line types available.

SET LINEWIDTH SCALE FACTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSLWSC *(lwidth)*

Arguments

Argument	Data Type	I/O	Description
lwidth	real	I	Linewidth scale factor

SET MARKER SIZE SCALE FACTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSMKSC *(sfactor)*

Arguments

Argument	Data Type	I/O	Description
sfactor	real	I	Polymarker size scale factor

SET MARKER TYPE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSMK *(mtype)*

Arguments

Argument	Data Type	I/O	Description
mtype	integer	I	Polymarker type (GPOINT, GPLUS, GAST, GOMARK, GXMARK). Nonstandard marker types available.

SET PATTERN REFERENCE POINT

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSPARF *(px, py)*

Arguments

Argument	Data Type	I/O	Description
px	real	I	X pattern reference point in world coordinates
py	real	I	Y pattern reference point in world coordinates

SET PATTERN REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSPAR *(workstation_id, pindex, dim_x, dim_y, scol, srow, ncols, nrows, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
pindex	integer	I	Pattern index value
dim_x	integer	I	X dimension of the pattern color array
dim_y	integer	I	Y dimension of the pattern color array
scol	integer	I	Starting column of the color array
srow	integer	I	Starting row of the color array
ncols	integer	I	Number of color array columns to traverse
nrows	integer	I	Number of color array rows to traverse
cindex	integer	I	Two dimension color index array dim_x by dim_y

SET PATTERN SIZE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSPA *(px, py)*

Arguments

Argument	Data Type	I/O	Description
px	real	I	Pattern size in X world coordinates
py	real	I	Pattern size in Y world coordinates

SET PICK IDENTIFIER

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSPKID *(pick_id)*

Arguments

Argument	Data Type	I/O	Description
pick_id	integer	I	Pick identifier

SET POLYLINE COLOR INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSPLCI *(cindex)*

Arguments

Argument	Data Type	I/O	Description
cindex	integer	I	Color index

SET POLYLINE INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSPLI *(pindex)*

Arguments

Argument	Data Type	I/O	Description
pindex	integer	I	Polyline index

SET POLYLINE REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSPLR *(workstation_id, pindex, ltype, lwidth, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
pindex	integer	I	Polyline index value
ltype	integer	I	Linetype (GLSOLI, GLDASH, GLDOT, GLDASD. Non-standard types available.)
lwidth	real	I	Linewidth scale factor
cindex	integer	I	Color index

SET POLYMARKER COLOR INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSPMCI *(cindex)*

Arguments

Argument	Data Type	I/O	Description
cindex	integer	I	Polymarker color index value

SET POLYMARKER INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSPMI *(pindex)*

Arguments

Argument	Data Type	I/O	Description
pindex	integer	I	Polymarker index value

SET POLYMARKER REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSPMR *(workstation_id, pindex, mtype, sfactor, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
pindex	integer	I	Polymarker index
mtype	integer	I	Polymarker type (GPOINT, GPLUS, GAST, GOMARK, GXMARK. Nonstandard marker types available.)
sfactor	real	I	Polymarker size scale factor
cindex	integer	I	Polymarker color index

SET TEXT ALIGNMENT

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSTXAL *(halign, valign)*

Arguments

Argument	Data Type	I/O	Description
halign	integer	I	Horizontal alignment (GAHNOR, GALEFT, GACENT, GARITE)
valign	integer	I	Vertical alignment (GAVNOR, GATOP, GACAP, GAHALF, GABASE, GABOTT)

SET TEXT COLOR INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSTXCI *(cindex)*

Arguments

Argument	Data Type	I/O	Description
cindex	integer	I	Color index

SET TEXT FONT AND PRECISION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSTXFP *(font, precision)*

Arguments

Argument	Data Type	I/O	Description
font	integer	I	Text font
precision	integer	I	Text precision (GSTRP, GCHARP, GSTRKP)

SET TEXT INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSTXI *(tindex)*

Arguments

Argument	Data Type	I/O	Description
tindex	integer	I	Text index value

SET TEXT PATH

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSTXP *(text_path)*

Arguments

Argument	Data Type	I/O	Description
text_path	integer	I	Text path (GRIGHT, GLEFT, GUP, GDOWN)

SET TEXT REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSTXR *(workstation_id, tindex, font, precision, efactor, spacing, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
tindex	integer	I	Text index value
font	integer	I	Text font
precision	integer	I	Text precision (GSTRP, GCHARP, GSTRKP)
efactor	real	I	Character expansion factor
spacing	real	I	Character spacing
cindex	integer	I	Text color index

Chapter 7

Transformation Functions

The DEC GKS transformation functions allow you to compose a picture, to control how much of the picture is seen on the workstation surface, and to control how much of the workstation surface is used to display the picture. The following list presents the transformation functions by category:

Category	GKS Functions
Normalization	GKS$SELECT_XFORM, GKS$SET_CLIPPING, GKS$SET_VIEWPORT, GKS$SET_VIEWPORT_PRIORITY, GKS$SET_WINDOW
Workstation	GKS$SET_WS_VIEWPORT, GKS$SET_WS_WINDOW

SELECT NORMALIZATION TRANSFORMATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSELNT *(xform)*

Arguments

Argument	Data Type	I/O	Description
xform	integer	I	Normalization transformation number

SET CLIPPING INDICATOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSCLIP *(flag)*

Arguments

Argument	Data Type	I/O	Description
flag	integer	I	Clipping flag (GNCLIP, GCLIP)

SET VIEWPORT

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSVP *(xform, xmin, xmax, ymin, ymax)*

Arguments

Argument	Data Type	I/O	Description
xform	integer	I	Normalization transformation number
xmin	real	I	X minimum NDC value
xmax	real	I	X maximum NDC value
ymin	real	I	Y minimum NDC value
ymax	real	I	Y maximum NDC value

SET VIEWPORT INPUT PRIORITY

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSVPIP *(xform, ref_xform, rel_prior)*

Arguments

Argument	Data Type	I/O	Description
xform	integer	I	Normalization transformation number
ref_xform	integer	I	Reference normalization transformation number
rel_prior	integer	I	Relative priority (GHIGHR, GLOWER)

SET WINDOW

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GSWN *(xform, xmin, xmax, ymin, ymax)*

Arguments

Argument	Data Type	I/O	Description
xform	integer	I	Normalization transformation number
xmin	real	I	X minimum world coordinate value
xmax	real	I	X maximum world coordinate value
ymin	real	I	Y minimum world coordinate value
ymax	real	I	Y maximum world coordinate value

SET WORKSTATION VIEWPORT

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSWKVP *(workstation_id, xmin, xmax, ymin, ymax)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
xmin	real	I	X minimum device coordinate value
xmax	real	I	X maximum device coordinate value
ymin	real	I	Y minimum device coordinate value
ymax	real	I	Y maximum device coordinate value

SET WORKSTATION WINDOW

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSWKWN *(workstation_id, xmin, xmax, ymin, ymax)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
xmin	real	I	X minimum NDC value
xmax	real	I	X maximum NDC value
ymin	real	I	Y minimum NDC value
ymax	real	I	Y maximum NDC value

Chapter 8

Input Functions

The DEC GKS input functions allow an application program to accept input from a user. The following list presents the input functions by category:

Category	GKS Functions
Initialization	GINCH, GINLC, GINPK, GINST, GINSK, GINVL
Mode Control	GSCHM, GSLCM, GSPKM, GSSTM, GSSKM, GSVLM
Request Mode	GRQCH, GRQLC, GRQPK, GRQST, GRQSK, GRQVL
Sample Mode	GSMCH, GSMLC, GSMPK, GSMST, GSMSK, GSMVL
Event Mode	GWAIT, GFLUSH, GTCH, GGTLC, GGTPK, GGTST, GGTSK, GGTVL

AWAIT EVENT

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GWAIT *(time_out, workstation_id, in_class, dev_num)*

Arguments

Argument	Data Type	I/O	Description
time_out	real	I	Time-out value (in seconds)
workstation_id	integer	O	Workstation identifier
in_class	integer	O	Input class (GLOCAT, GSTROK, GVALUA, GCHOIC, GPICK, GSTRIN, GNCLAS)
dev_num	integer	O	Logical input device number

FLUSH DEVICE EVENTS

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GFLUSH *(workstation_id, in_class, dev_num)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
in_class	integer	I	Input class (GLOCAT, GSTROK, GVALUA, GCHOIC, GPICK, GSTRIN)
dev_num	integer	I	Logical input device number

GET CHOICE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GGTCH *(in_status, ch_num)*

Arguments

Argument	Data Type	I/O	Description
in_status	integer	O	Input status flag (GOK, GNCHOI)
ch_num	integer	O	Choice number

GET LOCATOR

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GGTLC *(xform, pos_x, pos_y)*

Arguments

Argument	Data Type	I/O	Description
xform	integer	O	Normalization tranformation used
pos_x	real	O	X world coordinate position of locator
pos_y	real	O	Y world coordinate position of locator

GET PICK

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GGTPK *(in_status, segment_name, pick_id)*

Arguments

Argument	Data Type	I/O	Description
in_status	integer	O	Input status flag (GOK, GNPICK)
segment_ name	integer	O	Picked segment
pick_id	integer	O	Chosen pick identifier

GET STRING (FORTRAN 77)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GGTST *(num_char, cstring)*

Arguments

Argument	Data Type	I/O	Description
num_char	integer	O	Number of characters in the string
cstring	string	O	Character string

GET STRING (FORTRAN 77 subset)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GGTST *(num_char, cstring)*

Arguments

Argument	Data Type	I/O	Description
num_char	integer	O	Number of characters in the string
cstring	string	O	Character string

GET STROKE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GGTSK *(max_pts, xform, num_pts, px, py)*

Arguments

Argument	Data Type	I/O	Description
max_pts	integer	I	Maximum number of points that can be stored
xform	integer	O	Normalization transformation used
num_ points	integer	O	Number of points returned
px	real	O	Array containing X world coordinate stroke values
py	real	O	Array containing Y world coordinate stroke values

GET VALUATOR

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GGTVL *(value)*

Arguments

Argument	Data Type	I/O	Description
value	real	O	Returned value

INITIALIZE CHOICE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GINCH *(workstation_id, dev_num, in_status, in_choice, p_e_type, xmin, xmax, ymin, ymax, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Choice device number
in_status	integer	I	Initial input status (GOK, GNCHOI)
in_choice	integer	I	Initial choice number
p_e_type	integer	I	Prompt/echo type
xmin	real	I	X minimum device coordinate value
xmax	real	I	X maximum device coordinate value
ymin	real	I	Y minimum device coordinate value
ymax	real	I	Y maximum device coordinate value
dim_dr	integer	I	Dimension of data record
dr	strdata	I	Data record array of dimension dim_dr

When packing a data record using the function GPREC (see Section 1.2.7 or GPREC in this section for more information), the GPREC arguments needed for a prompt and echo type of 2 are as follows:

Component	Description
i_entries	Number of choice alternatives
int_array	Array of prompts
r_entries	The value 0
real_array	Dummy component
st_entries	The value 0
st_lengths	Dummy component
str_array	Not used

The GPREC arguments needed for a prompt and echo types of -1, 1, 3, or 4 are as follows:

Component	Description
i_entries	The value 0
int_array	Dummy component
r_entries	The value 0
real_array	Dummy component
st_entries	Number of choice prompt strings
st_lengths	Array of lengths of the choice prompt strings
str_array	Array of choice prompt strings

The GPREC arguments needed for a prompt and echo type of 5 are as follows:

Component	Description
i_entries	The value 1
int_array	The segment name
r_entries	The value 0
real_array	Dummy component
st_entries	The value 0
st_lengths	Dummy component
str_array	Dummy component

INITIALIZE LOCATOR

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GINLC *(workstation_id, dev_num, xform, px, py, p_e_type, xmin, xmax, ymin, ymax, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Locator device number
xform	integer	I	Translation normalization transformation number
px	real	I	Initial X position in world coordinates
py	real	I	Initial Y position in world coordinates
p_e_type	integer	I	Prompt/echo type
xmin	real	I	X minimum device coordinate value
xmax	real	I	X maximum device coordinate value
ymin	real	I	Y minimum device coordinate value
ymax	real	I	Y maximum device coordinate value
dim_dr	integer	I	Dimension of data record
dr	strdata	I	Data record array of dimension dim_dr

For prompt and echo types 1, 2, 3, and 6, you do not have to pack a data record using GPREC for this function. Define dim_dr to be the value 0, and then pass a null string for the data record.

When packing a data record using the function GPREC (see Section 1.2.7 or GPREC in this section for more information), the GPREC arguments needed for a prompt and echo type of 4 are as follows:

Component	Description
i_entries	If the attribute control flag is GSPEC, pass the value 7. If the attribute control flag is GCURNT, pass the value 1.
int_array	Pass the attribute control flag. If the flag is GSPEC, then also pass the values line type Attribute Control Flag (ASF), line width scale factor ASF, polyline color index ASF, polyline index, line type, and the polyline color index.
r_entries	If the attribute control flag is GSPEC, pass the value 1. If the attribute control flag is GCURNT, pass the value 0.
real_array	If the attribute control flag is GSPEC, pass the value of the line width scale factor. If the attribute control flag is GCURNT, pass a dummy value.
st_entries	The value 0.
st_lengths	Dummy component.
str_array	Dummy component.

If you use a prompt and echo type of 5, and if the polyline/fill area flag = GPLINE, then you need to specify the following arguments to GPREC:

Component	Description
i_entries	If the attribute control flag is GSPEC, pass the value 8. If the attribute control flag is GCURNT, pass the value 2.
int_array	Pass the values for the polyline/fill area flag and the attribute control flag. If the flag is GSPEC, then also pass the values line type Attribute Control Flag (ASF), line width scale factor ASF, polyline color index ASF, polyline index, line type, and the polyline color index.
r_entries	If the attribute control flag is GSPEC, pass the value 1. If the attribute control flag is GCURNT, pass the value 0.

Component	Description
real_array	If the attribute control flag is GSPEC, pass the value of the line width scale factor. If the attribute control flag is GCURNT, pass a dummy value.
st_entries	The value 0.
st_lengths	Dummy component.
str_array	Dummy component.

If you use a prompt and echo type of 5, and if the polyline/fill area flag = GFILLA, then you need to specify the following arguments to GPREC:

Component	Description
i_entries	If the attribute control flag is GSPEC, pass the value 8. If the attribute control flag is GCURNT, pass the value 2.
int_array	Pass the values for the polyline/fill area flag and the attribute control flag. If the flag is GSPEC, then also pass the values fill area interior style Attribute Source Flag (ASF), fill area style index ASF, fill area color ASF, fill area index, fill area interior style, and the fill area style index.
r_entries	The value 0.
real_array	Dummy component.
st_entries	The value 0.
st_lengths	Dummy component.
str_array	Dummy component.

INITIALIZE PICK

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GINPK *(workstation_id, dev_num, istatus, isegment, i_pick_id, p_e_type, xmin, xmax, ymin, ymax, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Pick device number
istatus	integer	I	Initial pick status (GOK, GNPICK)
isegment	integer	I	Initially picked segment
i_pick_id	integer	I	Initially picked pick identifier
p_e_type	integer	I	Prompt/echo type
xmin	real	I	X minimum device coordinate value
xmax	real	I	X maximum device coordinate value
ymin	real	I	Y minimum device coordinate value
ymax	real	I	Y maximum device coordinate value
dim_dr	integer	I	Dimension of data record
dr	strdata	I	Data record array of dimension dim_dr

You do not have to pack a data record using GPREC for this function. Define dim_dr to be the value 0 and then pass a null string.

INITIALIZE STRING (FORTRAN 77)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GINST *(workstation_id, dev_num, lstring, istring, p_e_type, xmin, xmax, ymin, ymax, buf_len, i_cur_pos, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier.
dev_num	integer	I	String device number.
lstring	integer	I	Length of initial string; must be >= 0. The number of characters actually used is the minimum of lstring and the length of istring.
istring	string	I	Initial string.
p_e_type	integer	I	Prompt/echo type.
xmin	real	I	X minimum device coordinate value.
xmax	real	I	X maximum device coordinate value.
ymin	real	I	Y minimum device coordinate value.
ymax	real	I	Y maximum device coordinate value.
buf_len	integer	I	Length of string buffer.
i_cur_pos	integer	I	Initial cursor position in initial string.
dim_dr	integer	I	Dimension of data record.
dr	strdata	I	Data record array of dimension dim_dr.

You do not have to pack a data record using GPREC for this function. Define dim_dr to be the value 0 and then pass a null string.

INITIALIZE STRING (FORTRAN 77 subset)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GINST *(workstation_id, dev_num, lstring, istring, p_e_type, xmin, xmax, ymin, ymax, buf_len, i_cur_pos, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	String device number
lstring	integer	I	Length of initial string
istring	integer	I	Initial string
p_e_type	integer	I	Prompt/echo type
xmin	real	I	X minimum device coordinate value
xmax	real	I	X maximum device coordinate value
ymin	real	I	Y minimum device coordinate value
ymax	real	I	Y maximum device coordinate value
buf_len	integer	I	Length of string buffer
i_cur_pos	integer	I	Initial cursor position in initial string
dim_dr	integer	I	Dimension of data record
dr	strdata	I	Data record array of dimension dim_dr

You do not have to pack a data record using GPREC for this function. Define dim_dr to be the value 0 and then pass a null string.

INITIALIZE STROKE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GINSK *(workstation_id, dev_num, xform, num_pts, px, py, p_e_type, xmin, xmax, ymin, ymax, buf_len, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Stroke device number
xform	integer	I	Initial normalization transformation number
num_pts	integer	I	Number of points in initial stroke
px	real	I	Array of X world coordinate values in initial stroke
py	real	I	Array of Y world coordinate values in initial stroke
p_e_type	integer	I	Prompt/echo type
xmin	real	I	X minimum device coordinate value
xmax	real	I	X maximum device coordinate value
ymin	real	I	Y minimum device coordinate value
ymax	real	I	Y maximum device coordinate value
buf_len	integer	I	Input buffer size
dim_dr	integer	I	Dimension of data record
dr	strdata	I	Data record array of dimension dim_dr

When packing a data record using the function GPREC (see Section 1.2.7 or GPREC in this section for more information), the GPREC arguments, needed for prompt and echo types of 1 or 2, are as follows:

Component	Description
i_entries	The value 0
int_array	Dummy component
r_entries	The value 3
real_array	The X, Y, and time intervals
st_entries	The value 0
st_lengths	Dummy component
str_array	Dummy component

The GPREC arguments needed for a prompt and echo type of 3 are as follows:

Component	Description
i_entries	If the attribute control flag is GSPEC, pass the value 5. If the attribute control flag is GCURNT, pass the value 1.
int_array	Pass the value for the attribute control flag. If the flag is GSPEC, then also pass the values marker size scale factor Attribute Source Flag (ASF), polymarker color index ASF, polymarker index, and the polymarker color index.
r_entries	If the attribute control flag is GSPEC, pass the value 4. If the attribute control flag is GCURNT, pass the value 3.
real_array	Pass the values for the X, Y, and time intervals. If the attribute control flag is GSPEC, then also pass the marker size scale factor.
st_entries	The value 0.
st_lengths	Dummy component.
str_array	Dummy component.

The GPREC arguments needed for a prompt and echo type of 4 are as follows:

Component	Description
i_entries	If the attribute control flag is GSPEC, pass the value 7. If the attribute control flag is GCURNT, pass the value 1.
int_array	Pass the values for the attribute control flag. If the flag is GSPEC, then also pass the values line type Attribute Control Flag (ASF), line width scale factor ASF, polyline color index ASF, polyline index, line type, and the polyline color index.
r_entries	If the attribute control flag is GSPEC, pass the value 4. If the attribute control flag is GCURNT, pass the value 3.
real_array	Pass the values X, Y, and time intervals. If the attribute control flag is GSPEC, then also pass the line width scale factor.
st_entries	The value 0.
st_lengths	Dummy component.
str_array	Dummy component.

INITIALIZE VALUATOR

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GINVL *(workstation_id, dev_num, ivalue, p_e_type, xmin, xmax, ymin, ymax, low_val, high_val, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Valuator device number
ivalue	real	I	Initial value
p_e_type	integer	I	Prompt/echo type
xmin	real	I	X minimum device coordinate value
xmax	real	I	X maximum device coordinate value
ymin	real	I	Y minimum device coordinate value
ymax	real	I	Y maximum device coordinate value
low_val	real	I	Minimum value
high_val	real	I	Maximum value
dim_dr	integer	I	Dimension of data record
dr	strdata	I	Data record array of dimension dim_dr

You do not have to pack a data record using GPREC for this function. Define dim_dr to be the value 0 and then pass a null string.

REQUEST CHOICE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRQCH *(workstation_id, dev_num, in_status, ch_num)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
in_status	integer	O	Input status flag (GNONE, GOK, GNCHOI)
ch_num	integer	O	Choice number

REQUEST LOCATOR

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRQLC *(workstation_id, dev_num, in_status, xform, pos_x, pos_y)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
in_status	integer	O	Input status flag (GOK, GNONE)
xform	integer	O	Normalization tranformation used
pos_x	real	O	X world coordinate position of locator
pos_y	real	O	Y world coordinate position of locator

REQUEST PICK

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRQPK *(workstation_id, dev_num, in_status, segment_name, pick_id)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
in_status	integer	O	Input status flag (GNONE, GOK, GNPICK)
segment_name	integer	O	Picked segment
pick_id	integer	O	Chosen pick identifier

REQUEST STRING (FORTRAN 77)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRQST *(workstation_id, dev_num, in_status, num_char, cstring)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
in_status	integer	O	Input status flag (GNONE, GOK)
num_char	integer	O	Number of characters in the string
cstring	string	O	Character string

REQUEST STRING (FORTRAN 77 subset)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRQST *(workstation_id, dev_num, in_status, num_char, cstring)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
in_status	integer	O	Input status flag (GNONE, GOK)
num_char	integer	O	Number of characters in the string
cstring	string	O	Character string

REQUEST STROKE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRQSK *(workstation_id, dev_num, max_pts, in_status, xform, num_pts, px, py)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
max_pts	integer	I	Maximum number of points that can be stored
in_status	integer	O	Input status flag (GNONE, GOK)
xform	integer	O	Normalization transformation used
num_points	integer	O	Number of points returned
px	real	O	Array containing X world coordinate stroke values
py	real	O	Array containing Y world coordinate stroke values

REQUEST VALUATOR

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRQVL *(workstation_id, dev_num, in_status, value)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
in_status	integer	O	Input status flag (GNONE, GOK)
value	real	O	Returned value

SAMPLE CHOICE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSMCH *(workstation_id, dev_num, in_status, ch_num)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
in_status	integer	O	Input status flag (GOK, GNCHOI)
ch_num	integer	O	Choice number

SAMPLE LOCATOR

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSMLC *(workstation_id, dev_num, xform, pos_x, pos_y)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
xform	integer	O	Normalization tranformation used
pos_x	real	O	X world coordinate position of locator
pos_y	real	O	Y world coordinate position of locator

SAMPLE PICK

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSMPK *(workstation_id, dev_num, in_status, segment_name, pick_id)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
in_status	integer	O	Input status flag (GOK, GNPICK)
segment_name	integer	O	Picked segment
pick_id	integer	O	Chosen pick identifier

SAMPLE STRING (FORTRAN 77)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSMST *(workstation_id, dev_num, num_char, cstring)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
num_char	integer	O	Number of characters in the string
cstring	string	O	Character string

SAMPLE STRING (FORTRAN 77 subset)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSMST *(workstation_id, dev_num, num_char, cstring)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
num_char	integer	O	Number of characters in the string
cstring	string	O	Character string

SAMPLE STROKE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSMSK *(workstation_id, dev_num, max_pts, xform, num_pts, px, py)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
max_pts	integer	I	Maximum number of points that can be stored
xform	integer	O	Normalization transformation used
num_points	integer	O	Number of points returned
px	real	O	Array containing X world coordinate stroke values
py	real	O	Array containing Y world coordinate stroke values

SAMPLE VALUATOR

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSMVL *(workstation_id, dev_num, value)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
value	real	O	Returned value

SET CHOICE MODE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSCHM *(workstation_id, dev_num, operating_mode, echo)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
operating_mode	integer	I	Input operating mode (GREQU, GSAMPL, GEVENT)
echo	integer	I	Echo flag (GNECHO, GECHO)

SET LOCATOR MODE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSLCM *(workstation_id, dev_num, operating_mode, echo)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
operating_mode	integer	I	Input operating mode (GREQU, GSAMPL, GEVENT)
echo	integer	I	Echo flag (GNECHO, GECHO)

SET PICK MODE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSPKM *(workstation_id, dev_num, operating_mode, echo)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
operating_mode	integer	I	Input operating mode (GREQU, GSAMPL, GEVENT)
echo	integer	I	Echo flag (GNECHO, GECHO)

SET STRING MODE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSSTM *(workstation_id, dev_num, operating_mode, echo)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
operating_mode	integer	I	Input operating mode (GREQU, GSAMPL, GEVENT)
echo	integer	I	Echo flag (GNECHO, GECHO)

SET STROKE MODE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSSKM *(workstation_id, dev_num, operating_mode, echo)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
operating_mode	integer	I	Input operating mode (GREQU, GSAMPL, GEVENT)
echo	integer	I	Echo flag (GNECHO, GECHO)

SET VALUATOR MODE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSVLM *(workstation_id, dev_num, operating_mode, echo)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
dev_num	integer	I	Device number
operating_mode	integer	I	Input operating mode (GREQU, GSAMPL, GEVENT)
echo	integer	I	Echo flag (GNECHO, GECHO)

Chapter 9

Segment Functions

The DEC GKS segment functions create, manipulate, and delete stored groups of output primitives called *segments*. The segment functions can be divided into the following categories:

Category	GKS Functions
Using Segments	GASGWK, GCLSG, GCSGWK, GCRSG, GDSG, GDSGWK, GINSG, RENSG
Primitive Attributes	GSPKID
Segment Attributes	GSDTEC, GSHLIT, GSSGP, GSVIS
Segment Transformations	GACTM, GEVTM, GSSGT

ACCUMULATE TRANSFORMATION MATRIX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GACTM *(matrix, fx, fy, x_vector, y_vector, rotate, scale_x, scale_y, wc_ndc, nmatrix)*

Arguments

Argument	Data Type	I/O	Description
matrix	real	I	2 x 3 Segment transformation matrix
fx	real	I	Fixed-point X value
fy	real	I	Fixed-point Y value
x_vector	real	I	The X value of the shift vector
y_vector	real	I	The Y value of the shift vector
rotate	real	I	Rotation in radians
scale_x	real	I	X scale factor
scale_y	real	I	Y scale factor
wc_ndc	integer	I	Coordinate switch (GWC, GNDC)
nmatrix	real	O	New transformation matrix (2 x 3); the new component values concatenated to the old matrix

ASSOCIATE SEGMENT WITH WORKSTATION

Operating States: GWSOP, GWSAC

Syntax

GASGWK *(workstation_id, segment_name)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
segment_name	integer	I	Segment name

CLOSE SEGMENT

Operating States: GSGOP

Syntax

GCLSG *( )*

COPY SEGMENT TO WORKSTATION

Operating States: GWSOP, GWSAC

Syntax

GCSGWK *(workstation_id, segment_name)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
segment_name	integer	I	Segment name

CREATE SEGMENT

Operating States: GWSAC

Syntax

GCRSG *(segment_name)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name

DELETE SEGMENT

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GDSG *(segment_name)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name

DELETE SEGMENT FROM WORKSTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GDSGWK *(workstation_id, segment_name)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
segment_name	integer	I	Segment name

EVALUATE TRANSFORMATION MATRIX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GEVTM *(fx, fy, x_vector, y_vector, rotate, scale_x, scale_y, wc_ndc, matrix)*

Arguments

Argument	Data Type	I/O	Description
fx	real	I	Fixed-point X value
fy	real	I	Fixed-point Y value
x_vector	real	I	The X value of the shift vector
y_vector	real	I	The Y value of the shift vector
rotate	real	I	Rotation in radians
scale_x	real	I	X scale factor
scale_y	real	I	Y scale factor
wc_ndc	integer	I	Coordinate switch (GWC, GNDC)
matrix	real	O	2 x 3 evaluated transformation matrix

INSERT SEGMENT

Operating States: GWSAC, GSGOP

Syntax

GINSG *(segment_name, matrix)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name
matrix	real	I	2 x 3 insertion transformation matrix

RENAME SEGMENT

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRENSG *(old_name, new_name)*

Arguments

Argument	Data Type	I/O	Description
old_name	integer	I	Segment's previous name
new_name	integer	I	Segment's new name

SET DETECTABILITY

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSDTEC *(segment_name, detect)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name
detect	integer	I	Detectability (GUNDET, GDETEC)

SET HIGHLIGHTING

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSHLIT *(segment_name, high)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name
high	integer	I	Highlighting (GNORMAL, GHILIT)

SET SEGMENT PRIORITY

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSSGP *(segment_name, priority)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name
priority	real	I	Segment priority

SET VISIBILITY

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSVIS *(segment_name, visible)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name
visible	integer	I	Segment visibility (GINVIS, GVISI)

SET SEGMENT TRANSFORMATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GSSGT *(segment_name, matrix)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name
matrix	real	I	2 x 3 transformation matrix

Chapter 10

Metafile, Error, and Utility Functions

The DEC GKS metafile functions provide a mechanism for long-term storage, communication, and reproduction of a graphical image. Metafiles created by an application can be used by other applications on other computer systems to reproduce a picture. When you store picture information in a *metafile*, you store specific information concerning the output primitives contained in the picture, the corresponding output attributes, and other information that may be needed to reproduce the picture.

The DEC GKS error-handling functions provide a method for you to control the generation of messages to the user, and a method of exit when a DEC GKS function call generates an error. The following list presents the DEC GKS error-handling functions:

- GKS$EMERGENCY_CLOSE
- GKS$ERROR_HANDLER
- GKS$LOG_ERROR
- GKS$SET_ERROR_HANDLER

GET ITEM TYPE FROM GKSM

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GGTITM *(workstation_id, item_type, len_dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
item_type	integer	O	Item type
len_dr	integer	O	Length of the data record in bytes

INTERPRET ITEM

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GIITM *(item_type, len_dr, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
item_type	integer	I	Item type
len_dr	integer	I	Total length of the data record in bytes
dim_dr	integer	I	Dimension of the data record array
dr	strdat	I	The item data record

READ ITEM FROM GKSM

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GRDITM *(workstation_id, len_dr, len_buf, dr_buf)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier.
len_dr	integer	I	Length of the item data record (in bytes), which can be obtained in a call to GGTITM.
len_buf	integer	I	Maximum dimension of returned data record array. If the dimension is too small to receive the data record, DEC GKS truncates the data record to fit into your buffer.
dr_buf	strdat	O	Your item data record.

WRITE ITEM TO GKSM

Operating States: GWSAC, GSGOP

Syntax

GWITM *(workstation_id, item_type, len_dr, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
item_type	integer	I	Item type
len_dr	integer	I	Length of the data record in bytes
dim_dr	integer	I	Dimension of the data record array
dr	strdat	I	Data record

EMERGENCY CLOSE GKS

Operating States: GGKCL, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GECLKS *( )*

ERROR HANDLING

Operating States: GGKCL, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GERHND *(error_number, fun_id, error_file)*

Arguments

Argument	Data Type	/O	Description
error_ number	integer	I	Error number
fun_id	integer	I	Function identifier
error_file	integer	I	Error file

When specifying function identifiers such as the fun_id argument, specify the name of the FORTRAN binding function name, but replace the initial letter G with the letter E. For instance, to specify the function GOPWK (OPEN WORKSTATION), pass the function identifier EOPWK as an argument to this function.

ERROR LOGGING

Operating States: GGKCL, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GERLOG *(error_number, fun_id, error_file)*

Arguments

Argument	Data Type	I/O	Description
error_ number	integer	I	Error number
fun_id	integer	I	Function identifier
error_file	integer	I	Error file

When specifying function identifiers such as the fun_id argument, specify the name of the FORTRAN binding function name, but replace the initial letter G with the letter E. For instance, to specify the function GOPWK (OPEN WORKSTATION), pass the function identifier EOPWK as an argument to this function.

PACK DATA RECORD

Operating States: GGKCL, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GPREC *(i_entries, int_array, r_entries, real_array, st_entries, st_lengths, str_array, max_dr, error_status, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
i_entries	integer	I	Number of integer entries to be packed into the data record
int_array	integer	I	Array of dimension i_entries containing the integer values
r_entries	integer	I	Number of real entries to be packed into the data record
real_array	real	I	Array of dimension r_entries containing the real values
st_entries	integer	I	Number of string entries to be packed into the data record
st_lengths	integer	I	Array of dimension st_entries containing the number of characters in each of the strings
str_array	strdat	I	Array of dimension st_entries containing the character string data
max_dr	integer	I	Maximum dimension of the data record buffer
error_ status	integer	O	Error status
dim_dr	integer	O	Amount of the buffer used to pack the data record
dr	strdat	O	The data record buffer

The function GPREC, like the inquiry functions, does not generate an error. Rather, it reports error conditions through use of the error_status argument. If the function writes the value 0 to the error indicator, the function execution succeeded without encountering an error condition. See Section 1.2.7 for detailed information concerning GPREC and packed data records.

PACK DATA RECORD (FORTRAN 77 subset)

Operating States: GGKCL, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GPREC *(i_entries, int_array, r_entries, real_array, st_entries, st_lengths, str_array, max_dr, error_status, dim_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
i_entries	integer	I	Number of integer entries to be packed into the data record
int_array	integer	I	Array of dimension i_entries containing the integer values
r_entries	integer	I	Number of real entries to be packed into the data record
real_array	real	I	Array of dimension r_entries containing the real values
st_entries	integer	I	Number of string entries to be packed into the data record
st_lengths	integer	I	Array of dimension st_entries containing the number of characters in each of the strings
str_array	strdat	I	Array of dimension st_entries containing the character string data
max_dr	integer	I	Maximum dimension of the data record buffer
error_ status	integer	O	Error status
dim_dr	integer	O	Amount of the buffer used to pack the data record
dr	strdat	O	The data record buffer

The function GPREC, like the inquiry functions, does not generate an error. Rather, it reports error conditions through use of the error_status argument. If the function writes the value 0 to the error indicator, the function execution succeeded without encountering an error condition. See Section 1.2.7 for detailed information concerning GPREC and packed data records.

UNPACK DATA RECORD

Operating States: GGKCL, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GUREC *(dim_dr, dr, max_int, max_real, max_str, error_status, i_entries, int_array, r_entries, real_array, st_entries, st_lengths, str_array)*

Arguments

Argument	Data Type	I/O	Description
dim_dr	integer	I	Amount of the buffer containing the packed data record (number of array elements used)
dr	strdat	I	The packed data record buffer
max_int	integer	I	The maximum number of integer values that your declared array can hold
max_real	integer	I	The maximum number of real values that your declared array can hold
max_str	integer	I	The maximum number of strings that your declared array can hold
error_ status	integer	O	Error status
i_entries	integer	O	Number of integer entries to be packed into the data record
int_array	integer	O	Array of dimension i_entries containing the integer values
r_entries	integer	O	Number of real entries to be packed into the data record

Argument	Data Type	I/O	Description
real_array	real	O	Array of dimension r_entries containing the real values
st_entries	integer	O	Number of string entries to be packed into the data record
st_lengths	integer	O	Array of dimension st_entries containing the number of characters in each of the strings
str_array	strdat	O	Array of dimension st_entries containing the character string data

The function GUREC, like the inquiry functions, does not generate an error. Rather, it reports error conditions through use of the error_status argument. If the function writes the value 0 to the error indicator, the function execution succeeded without encountering an error condition. See Section 1.2.7 for detailed information concerning GPREC and packed data records.

UNPACK DATA RECORD (FORTRAN 77 subset)

Operating States: GGKCL, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GUREC *(dim_dr, dr, max_int, max_real, max_str, error_status, i_entries, int_array, r_entries, real_array, st_entries, st_lengths, str_array)*

Arguments

Argument	Data Type	I/O	Description
dim_dr	integer	I	Amount of the buffer containing the packed data record
dr	strdat	I	The packed data record buffer
max_int	integer	I	The maximum number of integer values that your declared array can hold
max_real	integer	I	The maximum number of real values that your declared array can hold
max_str	integer	I	The maximum number of strings that your declared array can hold
error_ status	integer	O	Error status
i_entries	integer	O	Number of integer entries packed into the data record
int_array	integer	O	Array of dimension i_entries containing the integer values
r_entries	integer	O	Number of real entries packed into the data record

Argument	Data Type	I/O	Description
real_array	real	O	Array of dimension r_entries containing the real values
st_entries	integer	O	Number of string entries packed into the data record
st_lengths	integer	O	Array of dimension st_entries containing the number of characters in each of the strings
str_array	strdat	O	Array of dimension st_entries containing the character string data

The function GUREC, like the inquiry functions, does not generate an error. Rather, it reports error conditions through use of the error_status argument. If the function writes the value 0 to the error indicator, the function execution succeeded without encountering an error condition. See Section 1.2.7 for detailed information concerning GPREC and packed data records.

Chapter 11

Inquiry Functions

The DEC GKS inquiry functions allow you to obtain current and/or default values for the operating state, output function attributes, deferral and regeneration modes, transformations, segments, and device capabilities. DEC GKS writes the values from the state lists and description tables to the inquiry function arguments.

INQUIRE LEVEL OF GKS

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQLVKS *(error_status, level)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
level	integer	O	GKS level (LMA, LMB, LMC, L0A, L0B, L0C, L1A, L1B, L1C, L2A, L2B, L2C)

INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQMNTN *(error_status, max)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
max	integer	O	Maximum normalization transformations number supported

INQUIRE WORKSTATION MAXIMUM NUMBERS

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQWKM *(error_status, sim_open, sim_active, ws_w_seg)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
sim_open	integer	O	Maximum number of simultaneously open workstations supported
sim_active	integer	O	Maximum number of simultaneously active workstations supported
ws_w_seg	integer	O	Maximum number of workstations that can be associated with a segment

INQUIRE LIST element OF AVAILABLE WORKSTATION TYPES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQEWK *(element, error_status, num_types, relement)*

Arguments

Argument	Data Type	I/O	Description
element	integer	I	The element of the list of workstations you wish to have returned
error_ status	integer	O	Error status
num_types	integer	O	The total number of workstation types (elements)
relement	integer	O	The workstation type of the requested element of the list

INQUIRE COLOR FACILITIES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCF *(workstation_type, error_status, ncolors, color_flag, nindexes)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
ncolors	integer	O	Number of colors
color_flag	integer	O	Color availability flag (GMONOC, GCOLOR)
nindexes	integer	O	Number of predefined indexes

INQUIRE DEFAULT CHOICE DATA

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQDCH *(workstation_type, device_number, element, dim_dr, error_status, num_choi, num_types, relement, echo_area, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
device_number	integer	I	Device number
element	integer	I	Element of the available prompt and echo type list requested
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
num_choi	integer	O	Maximum number of choices
num_types	integer	O	Number of prompt and echo types in the list
relement	integer	O	Prompt and echo type of the requested element of the list
echo_area	real	O	Array of four device coordinates specifying the default echo area in the order XMIN, XMAX, YMIN, YMAX
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	Choice data record

INQUIRE DEFAULT DEFERRAL STATE VALUES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDDS *(workstation_type, error_status, def_mode, reg_mode)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_status	integer	O	Error status
def_mode	integer	O	Default deferral mode (GASAP, GBNIG, GBNIL, GASTI)
reg_mode	integer	O	Default regeneration mode (GSUPPD, GALLOW)

INQUIRE DEFAULT LOCATOR DEVICE DATA

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDLC *(workstation_type, device_number, element, dim_dr, error_status, px, py, num_types, relement, echo_area, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
device_number	integer	I	Device number
element	integer	I	Element of the available prompt and echo type list requested
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
px	real	O	Default initial X world coordinate locator position
py	real	O	Default initial Y world coordinate locator position
num_types	integer	O	Number of prompt and echo types in the list
relement	integer	O	Prompt and echo type of the requested element

Argument	Data Type	I/O	Description
echo_area	real	O	Array of four device coordinates specifying the default echo area in the order XMIN, XMAX, YMIN, YMAX
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	Locator data record

INQUIRE DEFAULT PICK DEVICE DATA

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDPK *(workstation_type, device_number, element, dim_dr, error_status, num_types, relement, echo_area, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
device_ number	integer	I	Device number
element	integer	I	Element of the available prompt and echo type list requested
dim_dr	integer	I	Dimension of the data record variable
error_ status	integer	O	Error status
num_types	integer	O	Number of prompt and echo types in the list
relement	integer	O	Prompt and echo type of the requested element
echo_area	real	O	Array of four device coordinates specifying the default echo area in the order XMIN, XMAX, YMIN, YMAX
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	Pick data record

INQUIRE DEFAULT STRING DEVICE DATA

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDST *(workstation_type, device_number, element, dim_dr, error_status, max_buf, num_types, relement, echo_area, len_buf, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
device_ number	integer	I	Device number
element	integer	I	Element of the available prompt and echo type list requested
dim_dr	integer	I	Dimension of the data record variable
error_ status	integer	O	Error status
max_buf	integer	O	Maximum string buffer size
num_types	integer	O	Number of prompt and echo types in the list
relement	integer	O	Prompt and echo type of the requested element

Argument	Data Type	I/O	Description
echo_area	real	O	Array of four device coordinates specifying the default echo area in the order XMIN, XMAX, YMIN, YMAX
len_buf	integer	O	Length of the string buffer
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	String data record

INQUIRE DEFAULT STROKE DEVICE DATA

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDSK *(workstation_type, device_number, element, dim_dr, error_status, max_buf, num_types, relement, echo_area, len_buf, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
device_ number	integer	I	Device number
element	integer	I	Element of the available prompt and echo type list requested
dim_dr	integer	I	Dimension of the data record variable
error_ status	integer	O	Error status
max_buf	integer	O	Maximum stroke buffer size
num_types	integer	O	Number of prompt and echo types in the list
relement	integer	O	Prompt and echo type of the requested element

Argument	Data Type	I/O	Description
echo_area	real	O	Array of four device coordinates specifying the default echo area in the order XMIN, XMAX, YMIN, YMAX
len_buf	integer	O	Length of the stroke buffer
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	Stroke data record

INQUIRE DEFAULT VALUATOR DEVICE DATA

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDVL *(workstation_type, device_number, element, dim_dr, error_status, def_value, num_types, relement, echo_area, low_val, high_val, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
device_ number	integer	I	Device number
element	integer	I	Element of the available prompt and echo type list requested
dim_dr	integer	I	Dimension of the data record variable
error_ status	integer	O	Error status
def_value	real	O	Default value
num_types	integer	O	Number of prompt and echo types in the list
relement	integer	O	Prompt and echo type of the requested element
echo_area	real	O	Array of four device coordinates specifying the default echo area in the order XMIN, XMAX, YMIN, YMAX

Argument	Data Type	I/O	Description
low_val	real	O	Minimum possible value
high_val	real	O	Maximum possible value
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	Valuator data record

INQUIRE DISPLAY SPACE SIZE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDSP *(workstation_type, error_status, units, px, py, ras_x, ras_y)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
units	integer	O	Device coordinate measurement units (GMETRE, GOTHU)
px	real	O	Maximum X coordinate value in device coordinates
py	real	O	Maximum Y coordinate value in device coordinates
ras_x	integer	O	Maximum X value in raster units
ras_y	integer	O	Maximum Y value in raster units

INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDSGA *(workstation_type, error_status, xform, vis_on_off, vis_off_on, highlight, priority, add_prim, delete)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
xform	integer	O	Segment transformation (GIRG, GIMM)
vis_on_off	integer	O	Visibility turned from ON to OFF (GIRG, GIMM)
vis_off_on	integer	O	Visibility turned from OFF to ON (GIRG, GIMM)
highlight	integer	O	Highlighting changes (GIRG, GIMM)
priority	integer	O	Priority changes (GIRG, GIMM)
add_prim	integer	O	Adding primitives to an open segment (GIRG, GIMM)
delete	integer	O	Segment deletion (GIRG, GIMM)

INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQDWKA *(workstation_type, error_status, pl_rep, pm_rep, t_rep, fa_rep, pat_rep, c_rep, ws_xforms)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
pl_rep	integer	O	Polyline representation changeable (GIRG, GIMM)
pm_rep	integer	O	Polymarker representation changeable (GIRG, GIMM)
t_rep	integer	O	Text representation changeable (GIRG, GIMM)
fa_rep	integer	O	Fill area representation changeable (GIRG, GIMM)
pat_rep	integer	O	Pattern representation changeable (GIRG, GIMM)
c_rep	integer	O	Color representation changeable (GIRG, GIMM)
ws_xforms	integer	O	Workstation transformation changeable (GIRG, GIMM)

INQUIRE FILL AREA FACILITIES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQFAF *(workstation_type, selement, helement, error_status, num_int, r_selement, num_hatch, r_helement, num_index)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
selement	integer	I	Requested element of the list of available fill interior styles
helement	integer	I	Requested element of the list of available fill hatch styles
error_ status	integer	O	Error status
num_int	integer	O	Number of available interior styles
r_selement	integer	O	Interior style of the requested element (GHOLLO, GSOLID, GPATTR, GHATCH)
num_hatch	integer	O	Number of available hatch styles
r_helement	integer	O	Hatch style of the requested element
num_index	integer	O	The number of predefined fill area index values

INQUIRE GENERALIZED DRAWING PRIMITIVE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQGDP *(workstation_id, gdp_id, error_status, num_atts, list_atts)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation type
gdp_id	integer	I	GDP identifier
error_status	integer	O	Error status
num_atts	integer	O	Number of attribute bundles used
list_atts	integer	O	Array of four elements containing the bundles used (GPLBND, GPMBND, GTXBND, GFABND)

INQUIRE LIST element OF AVAILABLE GENERALIZED DRAWING PRIMITIVES

Operating States: GGKOP, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQEGDP *(workstation_type, element, error_status, num_gdp, relement)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
element	integer	I	Requested element of the list of available GDPs
error_ status	integer	O	Error status
num_gdp	integer	O	Number of available GDPs
relement	integer	O	GDP identifier of the requested element

INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQLWK *(workstation_type, error_status, m_pline, m_pmark, m_text, m_fill, m_patt, m_color)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	I	Error status
m_pline	integer	O	Maximum polyline bundle entries
m_pmark	integer	O	Maximum polymarker bundle entries
m_text	integer	O	Maximum text bundle entries
m_fill	integer	O	Maximum fill area bundle entries
m_patt	integer	O	Maximum pattern indexes
m_color	integer	O	Maximum color indexes

INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQLI *(workstation_type, error_status, num_loc, num_stk, num_val, num_ch, num_pi, num_stri)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
num_loc	integer	O	Number of locator devices
num_stk	integer	O	Number of stroke devices
num_val	integer	O	Number of valuator devices
num_ch	integer	O	Number of choice devices
num_pi	integer	O	Number of pick devices
num_stri	integer	O	Number of string devices

INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQSGP *(workstation_type, error_status, num_pri)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
num_pri	integer	O	Number of supported segment priorities

INQUIRE PATTERN FACILITIES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPAF *(workstation_type, error_status, nindexes)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
nindexes	integer	O	Number of predefined pattern indexes

INQUIRE POLYLINE FACILITIES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPLF *(workstation_type, element, error_status, num_types, relement, num_widths, nom_width, min_width, max_width, nindexes)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
element	integer	I	Requested element of the list of available line types
error_ status	integer	O	Error status
num_types	integer	O	Number of line types available
relement	integer	O	Line type of the requested element
num_ widths	integer	O	Number of line widths
nom_width	real	O	Default size line supported by the graphics handler (nominal)
min_width	real	O	Smallest line width possible
max_width	real	O	Largest line width possible
nindexes	integer	O	Number of predefined polyline index values

INQUIRE POLYMARKER FACILITIES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPMF *(workstation_type, element, error_status, num_types, relement, num_sizes, nom_size, min_size, max_size, nindexes)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
element	integer	I	Requested element of the list of available marker types
error_ status	integer	O	Error status
num_types	integer	O	Number of marker types available
relement	integer	O	Marker type of the requested element
num_sizes	integer	O	Number of available marker sizes
nom_size	real	O	Default size marker size supported by the graphics handler (nominal)
min_size	real	O	Smallest marker size possible
max_size	real	O	Largest marker size possible
nindexes	integer	O	Number of predefined polymarker index values

INQUIRE PREDEFINED COLOR REPRESENTATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPCR *(workstation_type, cindex, error_status, red_i, green_i, blue_i)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
cindex	integer	I	Color index
error_ status	integer	O	Error status
red_i	real	O	Red intensity
green_i	real	O	Green intensity
blue_i	real	O	Blue intensity

INQUIRE PREDEFINED FILL AREA REPRESENTATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPFAR *(workstation_type, findex, error_status, int_style, sindex, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
findex	integer	I	Fill area index
error_ status	integer	O	Error status
int_style	integer	O	Fill area interior style (GHOLLO, GSOLID, GPATTR, GHATCH)
sindex	integer	O	Fill area style index
cindex	integer	O	Fill area color index

INQUIRE PREDEFINED PATTERN REPRESENTATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPPAR *(workstation_type, pindex, max_x_dim, max_y_dim, error_status, dim_x, dim_y, carray)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
pindex	integer	I	Pattern index
max_x_dim	integer	I	Maximum X dimension of the color array
max_y_dim	integer	I	Maximum Y dimension of the color array
error_ status	integer	O	Error status
dim_x	integer	O	Columns of the color array filled
dim_y	integer	O	Rows of the color array filled
carray	integer	O	Array of max_x_dim columns and max_y_dim rows, containing the color indexes comprising the pattern

INQUIRE PREDEFINED POLYLINE REPRESENTATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPPLR *(workstation_type, pindex, error_status, ltype, lwidth, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
pindex	integer	I	Polyline index
error_ status	integer	O	Error status
ltype	integer	O	Linetype
lwidth	real	O	Linewidth scale factor
cindex	integer	O	Polyline color index

INQUIRE PREDEFINED POLYMARKER REPRESENTATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPPMR *(workstation_type, pindex, error_status, mtype, msize, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
pindex	integer	I	Polymarker index
error_ status	integer	O	Error status
mtype	integer	O	Marker type
msize	real	O	Marker size scale factor
cindex	integer	O	Polymarker color index

INQUIRE PREDEFINED TEXT REPRESENTATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPTXR *(workstation_type, tindex, error_status, font, precision, exp_factor, spacing, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
tindex	integer	I	Text index
error_ status	integer	O	Error status
font	integer	O	Text font
precision	integer	O	Text precision (GSTRP, GCHARP, GSTRKP)
exp_factor	real	O	Text expansion factor
spacing	real	O	Text spacing
cindex	integer	O	Text color index

INQUIRE TEXT FACILITIES

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQTXF *(workstation_type, element, error_status, num_font, relement_f, relement_p, num_height, min_height, max_height, num_exp, min_exp, max_exp, nindexes)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
element	integer	I	Requested element of the list of available font and precision pairs
error_ status	integer	O	Error status
num_font	integer	O	Number of font and precision pairs
relement_f	integer	O	Font number of the requested element
relement_p	integer	O	Precision of the requested element (GSTRP, GCHARP, GSTRKP)
num_ height	integer	O	Number of available character heights
min_height	real	O	Minimum character height in device coordinates
max_ height	real	O	Maximum character height in device coordinates

Argument	Data Type	I/O	Description
num_exp	integer	O	Number of available character expansions
min_exp	real	O	Minimum character expansion
max_exp	real	O	Maximum character expansion
nindexes	integer	O	Number of predefined text indexes

INQUIRE WORKSTATION CATEGORY

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQWKCA *(workstation_type, error_status, category)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
category	integer	O	Workstation category (GOUTPT, GINPUT, GOUTIN, GWISS, GMO, GMI)

INQUIRE WORKSTATION CLASSIFICATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQWKCL *(workstation_type, error_status, class)*

Arguments

Argument	Data Type	I/O	Description
workstation_ type	integer	I	Workstation type
error_ status	integer	O	Error status
class	integer	O	Workstation classification (GVECTR, GRASTR, GOTHWK)

INQUIRE (current) ASPECT SOURCE FLAGS

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQASF *(error_status, asf_list)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
asf_list	integer	O	Array of 13 aspect source flags (GBUNDL, GINDIVI), as follows: 1. Linetype 2. Linewidth scale factor 3. Polyline color index 4. Marker type 5. Marker size scale factor 6. Polymarker color index 7. Text font and precision 8. Character expansion factor 9. Character spacing 10. Text color index 11. Fill area interior style 12. Fill area style index 13. Fill area color index

INQUIRE (current) CHARACTER BASE VECTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCHB *(error_status, x_vec, y_vec)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
x_vec	real	O	X value of the character base vector
y_vec	real	O	Y value of the character base vector

INQUIRE (current) CHARACTER EXPANSION FACTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCHXP *(error_status, xfactor)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
xfactor	real	O	Character expansion factor

INQUIRE (current) CHARACTER HEIGHT

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCHH *(error_status, cheight)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
cheight	real	O	Character height, in world coordinates

INQUIRE (current) CHARACTER SPACING

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCHSP *(error_status, cspacing)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
cspacing	real	O	Character spacing

INQUIRE (current) CHARACTER UP VECTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCHUP *(error_status, x_up_vec, y_up_vec)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
x_up_vec	real	O	X world coordinate value of the character up vector
y_up_vec	real	O	Y world coordinate value of the character up vector

INQUIRE (current) CHARACTER WIDTH

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCHW *(error_status, cwidth)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
cwidth	real	O	Character width

INQUIRE CLIPPING

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCLIP *(error_status, cflag, crec)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
cflag	integer	O	Clipping indicator (GNCLIP, GCLIP)
crec	real	O	Array of four elements containing the NDC values of the clipping rectangle

INQUIRE (current) FILL AREA COLOR INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQFACI *(error_status, cindex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
cindex	integer	O	Fill area color index

INQUIRE (current) FILL AREA INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQFAI *(error_status, findex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
findex	integer	O	Fill area index

INQUIRE (current) FILL AREA INTERIOR STYLE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQFAIS *(error_status, istyle)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
istyle	integer	O	Interior style (GHOLLO, GSOLID, GPATTR, GHATCH)

INQUIRE (current) FILL AREA STYLE INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQFASI *(error_status, sindex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
sindex	integer	O	Fill area style index

INQUIRE INPUT QUEUE OVERFLOW

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQIQOV *(error_status, workstation_id, in_class, device_number)*

Arguments

Argument	Data Type	I/O	Description
error_status	integer	O	Error status
workstation_id	integer	O	Workstation identifier
in_class	integer	O	Input class (GLOCAT, GSTROK, GVALUA, GCHOIC, GPICK, GSTRIN)
device_number	integer	O	Device number

INQUIRE (current) LINETYPE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQLN *(error_status, ltype)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
ltype	integer	O	Line type

INQUIRE (current) LINEWIDTH SCALE FACTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQLWSC *(error_status, lwidth)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
lwidth	real	O	Line width scale factor

INQUIRE MORE SIMULTANEOUS EVENTS

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQSIM *(error_status, flag)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
flag	integer	O	More simultaneously generated events flag (GNMORE, GMORE)

INQUIRE (current) MARKER SIZE SCALE FACTOR

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQMKSC *(error_status, msize)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
msize	real	O	Marker size scale factor

INQUIRE (current) MARKERTYPE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQMK *(error_status, mtype)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
mtype	integer	O	Polymarker type

INQUIRE CURRENT NORMALIZATION TRANSFORMATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQCNTN *(error_status, xform)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
xform	integer	O	Current normalization transformation number

INQUIRE (current) PATTERN REFERENCE POINT

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPARF *(error_status, px, py)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
px	real	O	X value of the pattern reference point, in world coordinates
py	real	O	Y value of the pattern reference point, in world coordinates

INQUIRE (current) PATTERN SIZE

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPA *(error_status, x_width, y_width, x_height, y_height)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
x_width	real	O	X value of the pattern width vector
y_width	real	O	Y value of the pattern width vector
x_height	real	O	X value of the pattern height vector
y_height	real	O	Y value of the pattern height vector

INQUIRE CURRENT PICK IDENTIFIER

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPKID *(error_status, pick_id)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
pick_id	integer	O	Pick identifier

INQUIRE (current) POLYLINE COLOR INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPLCI *(error_status, cindex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
cindex	integer	O	Polyline color index

INQUIRE (current) POLYLINE INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPLI *(error_status, plindex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
plindex	integer	O	Polyline index

INQUIRE (current) POLYMARKER COLOR INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPMCI *(error_status, cindex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
cindex	integer	O	Polymarker color index

INQUIRE (current) POLYMARKER INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQPMI *(error_status, pmindex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
pmindex	integer	O	Polymarker index

INQUIRE (current) TEXT ALIGNMENT

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQTXAL *(error_status, h_align, v_align)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
h_align	integer	O	Horizontal alignment (GAHNOR, GALEFT, GACENT, GARITE)
v_align	integer	O	Vertical alignment (GAVNOR, GATOP, GACAP, GAHALF, GABASE, GABOTT)

INQUIRE (current) TEXT COLOR INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQTXCI *(error_status, cindex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
cindex	integer	O	Text color index

INQUIRE (current) TEXT FONT AND PRECISION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQTXFP *(error_status, font, precision)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
font	integer	O	Text font
precision	integer	O	Text precision (GSTRP, GCHARP, GSTRKP)

INQUIRE (current) TEXT INDEX

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQTXI *(error_status, tindex)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
tindex	integer	O	Text index

INQUIRE (current) TEXT

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQTXP *(error_status, tpath)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
tpath	integer	O	Text path (GRIGHT, GLEFT, GUP, GDOWN)

INQUIRE NAME OF OPEN SEGMENT

Operating States: GSGOP

Syntax

GQOPSG *(error_status, segment_name)*

Arguments

Argument	Data Type	I/O	Description
error_ status	integer	O	Error status
segment_ name	integer	O	Segment name

INQUIRE OPERATING STATE VALUE

Operating States: GGKCL, GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQOPS *(op_state)*

Arguments

Argument	Data Type	I/O	Description
op_state	integer	O	GKS operating state (GGKCL, GGKOP, GWSOP, GWSAC, GSGOP)

INQUIRE NORMALIZATION TRANSFORMATION NUMBER

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQNT *(xform, error_status, window, vport)*

Arguments

Argument	Data Type	I/O	Description
xform	integer	I	Normalization transformation number
error_ status	integer	O	Error status
window	real	O	Array of four world coordinate values comprising the world window
vport	real	O	Array of four NDC values comprising the world viewport

INQUIRE LIST element OF NORMALIZATION TRANSFORMATION NUMBERS

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQENTN *(element, error_status, num_xforms, relement)*

Arguments

Argument	Data Type	I/O	Description
element	integer	I	Element of the set of normalization transformations requested
error_ status	integer	O	Error status
num_ xforms	integer	O	Number of normalization transformations
relement	integer	O	Normalization transformation number of the requested element

INQUIRE SET member OF ACTIVE WORKSTATIONS

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQACWK *(member, error_status, num_active, rmember)*

Arguments

Argument	Data Type	I/O	Description
member	integer	I	Member of the set of active workstations
integer	error_ status	O	Error status
num_active	integer	O	Number of active workstations
rmember	integer	O	Workstation identifier of the requested element of the list

INQUIRE SET member OF OPEN WORKSTATIONS

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQOPWK *(member, error_status, num_open, rmember)*

Arguments

Argument	Data Type	I/O	Description
member	integer	I	Member of the set of open workstations requested
error_ status	integer	O	Error status
num_open	integer	O	Number of open workstations
rmember	integer	O	Workstation identifier of the requested element of the list

INQUIRE SET member OF SEGMENT NAMES IN USE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQSGUS *(member, error_status, num_seg, rmember)*

Arguments

Argument	Data Type	I/O	Description
member	integer	I	Member of the set of segment names requested
error_ status	integer	O	Error status
num_seg	integer	O	Number of segment names in use
rmember	integer	O	Segment name of the requested element

INQUIRE CHOICE DEVICE STATE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQCHS *(workstation_id, device_number, dim_dr, error_status, operating_mode, echo_flag, in_status, in_choice, p_e_type, echo_area, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
device_number	integer	I	Device number
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
operating_ mode	integer	O	Operating input mode (GREQU, GSAMPL, GEVENT)
echo_flag	integer	O	Echo flag (GNECHO, GECHO)
in_status	integer	O	Initial input status (GOK, GNCHOI)
in_choice	integer	O	Initial choice value
p_e_type	integer	O	Prompt and echo type
echo_area	real	O	Array of four device coordinates specifying an echo area in the order XMIN, XMAX, YMIN, YMAX
len_dr	integer	O	Number of array elements written to the data record
dr	strdat	O	Choice data record

INQUIRE COLOR REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQCR *(workstation_id, cindex, type, error_status, red_i, green_i, blue_i)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
cindex	integer	I	Color index
type	integer	I	Type of output values (GSET, GREALI)
error_status	integer	O	Error status
red_i	real	O	Red intensity
green_i	real	O	Green intensity
blue_i	real	O	Blue intensity

INQUIRE FILL AREA REPRESENTATION

Operating States: GGKOP, GWSOP, GWSAC, GSGOP

Syntax

GQFAR *(workstation_id, findex, type, error_status, int_style, sindex, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
findex	integer	I	Fill area index
type	integer	I	Type of output values (GSET, GREALI)
error_status	integer	O	Error status
int_style	integer	O	Interior style (GHOLLO, GSOLID, GPATTR, GHATCH)
sindex	integer	O	Interior style index
cindex	integer	O	Fill area color index

INQUIRE LIST element OF COLOR INDICES

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQECI *(workstation_id, element, error_status, num_color, relement)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
element	integer	I	Element of the set of color indexes requested
error_status	integer	O	Error status
num_color	integer	O	Number of color indexes
relement	integer	O	Color index of the requested element of the list

INQUIRE LIST element OF FILL AREA INDICES

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQEFAI *(workstation_id, element, error_status, num_fill, relement)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
element	integer	I	Element of the set of fill area indexes requested
error_status	integer	O	Error status
num_fill	integer	O	Number of fill area indexes
relement	integer	O	Fill area index of the requested element

INQUIRE LIST element OF PATTERN INDICES

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQEPAI *(workstation_id, element, error_status, num_patt, relement)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
element	integer	I	Element of the set of pattern indexes requested
error_status	integer	O	Error status
num_patt	integer	O	Number of pattern indexes
relement	integer	O	Pattern index of the requested element

INQUIRE LIST element OF POLYLINE INDICES

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQEPLI *(workstation_id, element, error_status, num_line, relement)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
element	integer	I	Element of the set of polyline indexes requested
error_status	integer	O	Error status
num_line	integer	O	Number of polyline indexes
relement	integer	O	Polyline index of the requested element

INQUIRE LIST element OF POLYMARKER INDICES

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQEPMI *(workstation_id, element, error_status, num_mark, relement)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
element	integer	I	Element of the set of polymarker indexes requested
error_status	integer	O	Error status
num_mark	integer	O	Number of polymarker indexes
relement	integer	O	Polymarker index of the requested element

INQUIRE LIST element OF TEXT INDICES

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQETXI *(workstation_id, element, error_status, num_text, relement)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
element	integer	I	Element of the set of text indexes requested
error_status	integer	O	Error status
num_text	integer	O	Number of text indexes
relement	integer	O	Text index of the requested element

INQUIRE LOCATOR DEVICE STATE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQLCS *(workstation_id, device_number, type, dim_dr, error_status, operating_mode, echo_flag, in_xform, in_px, in_py, p_e_type, echo_area, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
device_number	integer	I	Device number
type	integer	I	Type of output values (GSET, GREALI)
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
operating_ mode	integer	O	Operating input mode (GREQU, GSAMPL, GEVENT)
echo_flag	integer	O	Echo flag (GNECHO, GECHO)
in_xform	integer	O	Initial normalization transformation used to translate the locator points
in_px	real	O	Initial X world coordinate value
in_py	real	O	Initial Y world coordinate value
p_e_type	integer	O	Prompt and echo type

Argument	Data Type	I/O	Description
echo_area	real	O	Array of four device coordinates specifying an echo area in the order of XMIN, XMAX, YMIN, YMAX
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	Locator data record

INQUIRE PATTERN REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

SYNTAX

GQPAR *(workstation_id, pindex, type, max_x_dim, max_y_dim, error_status, x_dim, y_dim, carray)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
pindex	integer	I	Pattern index
type	integer	I	Type of output values (GSET, GREALI)
max_x_dim	integer	I	Maximum columns of elements of the color array to traverse
max_y_dim	integer	I	Maximum rows of elements of the color array to traverse
error_status	integer	O	Error status
x_dim	integer	O	Columns of the color array to be traversed
y_dim	integer	O	Rows of the color array to be traversed
carray	integer	O	A two dimensional color array, with max_x_dim columns and max_y_dim rows, containing color indexes

INQUIRE PICK DEVICE STATE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQPKS *(workstation_id, device_number, type, dim_dr, error_status, operating_mode, echo_flag, in_status, in_seg, in_pick_id, p_e_type, echo_area, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
device_number	integer	I	Device number
type	integer	I	Type of output values (GSET, GREALI)
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
operating_ mode	integer	O	Operating input mode (GREQU, GSAMPL, GEVENT)
echo_flag	integer	O	Echo flag (GNECHO, GECHO)
in_status	integer	O	Initial input status (GOK, GNPICK)
in_seg	integer	O	Initially picked segment
in_pick_id	integer	O	Initially chosen pick identifier
p_e_type	integer	O	Prompt and echo type

Argument	Data Type	I/O	Description
echo_area	real	O	Array of four device coordinates specifying an echo area in the order of XMIN, XMAX, YMIN, YMAX
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	Pick data record

INQUIRE POLYLINE REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQPLR *(workstation_id, pindex, type, error_status, ltype, lwidth, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
pindex	integer	I	Polyline index
type	integer	I	Type of output values (GSET, GREALI)
error_status	integer	O	Error status
ltype	integer	O	Line type
lwidth	real	O	Line width scale factor
cindex	integer	O	Polyline color index

INQUIRE POLYMARKER REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQPMR *(workstation_id, pindex, type, error_status, mtype, msize, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
pindex	integer	I	Polymarker index
type	integer	I	Type of output values (GSET, GREALI)
error_status	integer	O	Error status
mtype	integer	O	Marker type
msize	real	O	Polymarker size scale factor
cindex	integer	O	Polymarker color index

INQUIRE SET member OF SEGMENT NAMES ON WORKSTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQSGWK *(workstation_id, member, error_status, num_names, rmember)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
member	integer	I	Member of the set of segment names
error_status	integer	O	Error status
num_names	integer	O	Number of segment names
rmember	integer	O	Segment name of requested set member

INQUIRE STRING DEVICE STATE (FORTRAN 77)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQSTS *(workstation_id, device_number, dim_dr, error_status, operating_mode, echo_flag, num_chars, in_string, p_e_type, echo_area, buf_size, i_cur_pos, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
device_number	integer	I	Device number
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
operating_ mode	integer	O	Operating input mode (GREQU, GSAMPL, GEVENT)
echo_flag	integer	O	Echo flag (GNECHO, GECHO)
num_chars	integer	O	Number of characters in the initial string
in_string	string	O	Initial string
p_e_type	integer	O	Prompt and echo type
echo_area	real	O	Array of four device coordinates specifying an echo area in the order of XMIN, XMAX, YMIN, YMAX

Workstation State List Inquiry Functions
INQUIRE STRING DEVICE STATE (FORTRAN 77)

Argument	Data Type	I/O	Description
buf_size	integer	O	Buffer size for the initial string
i_cur_pos	integer	O	Initial position of the cursor
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	String data record

INQUIRE STRING DEVICE STATE (FORTRAN 77 subset)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQSTS *(workstation_id, device_number, dim_dr, error_status, operating_mode, echo_flag, num_chars, in_string, p_e_type, echo_area, buf_size, i_cur_pos, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
device_number	integer	I	Device number
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
operating_ mode	integer	O	Operating input mode (GREQU, GSAMPL, GEVENT)
echo_flag	integer	O	Echo flag (GNECHO, GECHO)
num_chars	integer	O	Number of characters in the initial string
in_string	string	O	Initial string
p_e_type	integer	O	Prompt and echo type
echo_area	real	O	Array of four device coordinates specifying an echo area in the order XMIN, XMAX, YMIN, YMAX

Workstation State List Inquiry Functions
INQUIRE STRING DEVICE STATE (FORTRAN 77 subset)

Argument	Data Type	I/O	Description
buf_size	integer	O	Buffer size for the initial string
i_cur_pos	integer	O	Initial position of the cursor
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	String data record

INQUIRE STROKE DEVICE STATE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQSKS *(workstation_id, device_number, type, max_pts, dim_dr, error_status, operating_mode, echo_flag, xform, num_pts, px, py, p_e_type, echo_area, buf_size, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
device_number	integer	I	Device number
type	integer	I	Type of output values (GSET, GREALI)
max_pts	integer	I	Maximum number of points that the coordinate arrays can hold
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
operating_ mode	integer	O	Operating input mode (GREQU, GSAMPL, GEVENT)
echo_flag	integer	O	Echo flag (GNECHO, GECHO)
xform	integer	O	Normalization transformation
num_points	integer	O	Number of points in initial stroke
px	real	O	Array containing the initial X world coordinates points of the stroke
py	real	O	Array containing the initial Y world coordinates points of the stroke

Argument	Data Type	I/O	Description
p_e_type	integer	O	Prompt and echo type
echo_area	real	O	Array of four device coordinates specifying an echo area in the order XMIN, XMAX, YMIN, YMAX
buf_size	integer	O	Buffer size for the initial stroke
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	Stroke data record

INQUIRE TEXT EXTENT (FORTRAN 77)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQTXX *(workstation_id, px, py, cstring, error_status, con_pt_x, con_pt_y, ext_x, ext_y)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
px	real	I	Starting X world coordinate value of the text string
py	real	I	Starting Y world coordinate value of the text string
cstring	string	I	The character string
error_status	integer	O	Error status
con_pt_x	real	O	X world coordinate value of the concatenation point
con_pt_y	real	O	Y world coordinate value of the concatenation point
ext_x	real	O	Array of four elements containing the X world coordinate values of the extent rectangle
ext_y	real	O	Array of four elements containing the Y world coordinate values of the extent rectangle

INQUIRE TEXT EXTENT (FORTRAN 77 subset)

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQTXXS *(workstation_id, px, py, lstring, cstring, error_status, con_pt_x, con_pt_y, ext_x, ext_y)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
px	real	I	Starting X world coordinate value of the text string
py	real	I	Starting Y world coordinate value of the text string
lstring	integer	I	Length of string in characters
cstring	string	I	The character string
error_status	integer	O	Error status
con_pt_x	real	O	X world coordinate value of the concatenation point
con_pt_y	real	O	Y world coordinate value of the concatenation point
ext_x	real	O	Array of four elements containing the X world coordinate values of the extent rectangle
ext_y	real	O	Array of four elements containing the Y world coordinate values of the extent rectangle

INQUIRE TEXT REPRESENTATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQTXR *(workstation_id, tindex, type, error_status, font, precision, ex_fac, spacing, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
tindex	integer	I	Text index
type	integer	I	Type of output values (GSET, GREALI)
error_status	integer	O	Error status
font	integer	O	Text font
precision	integer	O	Text precision (GSTRP, GCHARP, GSTRKP)
ex_fac	real	O	The character expansion factor
spacing	real	O	The character spacing
cindex	integer	O	The text color index

INQUIRE VALUATOR DEVICE STATE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQVLS *(workstation_id, device_number, dim_dr, error_status, operating_mode, echo_flag, in_value, p_e_type, echo_area, low_val, high_val, len_dr, dr)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
device_number	integer	I	Device number
dim_dr	integer	I	Dimension of the data record variable
error_status	integer	O	Error status
operating_ mode	integer	O	Operating input mode (GREQU, GSAMPL, GEVENT)
echo_flag	integer	O	Echo flag (GNECHO, GECHO)
in_value	real	O	Initial value
p_e_type	integer	O	Prompt and echo type
echo_area	real	O	Array of four device coordinates specifying an echo area in the order XMIN, XMAX, YMIN, YMAX
low_val	real	O	Minimum value
high_val	real	O	Maximum value
len_dr	integer	O	Number of array elements returned in the data record array
dr	strdat	O	String data record

INQUIRE WORKSTATION CONNECTION AND TYPE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQWKC *(workstation_id, error_status, con_id, workstation_type)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
error_status	integer	O	Error status
con_id	integer	O	Connection identifier
workstation_ type	integer	O	Workstation type

INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQWKDU *(workstation_id, error_status, def_mode, reg_mode, surface, new_frame)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
error_status	integer	O	Error status
def_mode	integer	O	Deferral mode (GASAP, GBNIG, GBNIL, GASTI)
reg_mode	integer	O	Implicit regeneration mode (GSUPPD, GALLOW)
surface	integer	O	Display surface empty flag (GNEMPT, GEMPTY)
new_frame	integer	O	New frame necessary at update (GNO, GYES)

INQUIRE WORKSTATION STATE

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQWKS *(workstation_id, error_status, state)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
error_status	integer	O	Error status
state	integer	O	Workstation state (GINACT, GACTIV)

INQUIRE WORKSTATION TRANSFORMATION

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQWKT *(workstation_id, error_status, state, r_win, c_win, r_view, c_view)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
error_status	integer	O	Error status
state	integer	O	Transformation update state (GNPEND, GPEND)
r_win	real	O	Array of four NDC points composing the requested workstation window in the order XMIN, XMAX, YMIN, YMAX
c_win	real	O	Array of four NDC points composing the current workstation window in the order XMIN, XMAX, YMIN, YMAX
r_view	real	O	Array of four device coordinates composing the requested workstation viewport in the order XMIN, XMAX, YMIN, YMAX
c_view	real	O	Array of four device coordinates composing the current workstation viewport in the order XMIN, XMAX, YMIN, YMAX

INQUIRE SEGMENT ATTRIBUTES

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQSGA *(segment_name, error_status, matrix, visible, highlight, priority, detect)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name
error_ status	integer	O	Error status
matrix	real	O	A 2 x 3 array containing segment transformation matrix
visible	integer	O	Visibility (GINVIS, GVISI)
highlight	integer	O	Highlighting (GNORML, GHILIT)
priority	real	O	Priority
detect	integer	O	Detectability (GUNDET, GDETEC)

INQUIRE SET member OF ASSOCIATED WORKSTATIONS

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQASWK *(segment_name, member, error_status, num_ws, rmember)*

Arguments

Argument	Data Type	I/O	Description
segment_ name	integer	I	Segment name
member	integer	I	Requested member of the set of associated workstations
error_ status	integer	O	Error status
num_ws	integer	O	Number of associated workstations
rmember	integer	O	Workstation identifier of requested element

INQUIRE PIXEL

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQPX *(workstation_id, px, py, error_status, cindex)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
px	real	I	X world coordinate value
py	real	I	Y world coordinate value
error_status	integer	O	Error status
cindex	integer	O	Color index

INQUIRE PIXEL ARRAY

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQPXA *(workstation_id, corner_x, corner_y, dim_x, dim_y, scol, srow, pcols, prows, error_status, in_vals, carray)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
corner_x	real	I	X world coordinate of the upper left corner
corner_y	real	I	Y world coordinate of the upper left corner
dim_x	integer	I	Number of columns in the color array
dim_y	integer	I	Number of rows in the color array
scol	integer	I	Starting column of the color array
srow	integer	I	Starting row of the color array
pcols	integer	I	Number of columns in the pixel array
prows	integer	I	Number of rows in the pixel array
error_status	integer	O	Error status
in_vals	integer	O	Invalid value flag (GABSNT, GPRSNT)
carray	integer	O	Two dimensional array of dim_x and dim_y dimensions containing the color index values

INQUIRE PIXEL ARRAY DIMENSIONS

Operating States: GWSOP, GWSAC, GSGOP

Syntax

GQPXAD *(workstation_id, px, py, dx, dy, error_status, pa_cols, pa_rows)*

Arguments

Argument	Data Type	I/O	Description
workstation_id	integer	I	Workstation identifier
px	real	I	X world coordinate value of the upper left corner
py	real	I	Y world coordinate value of the upper left corner
dx	real	I	X world coordinate value of the lower right corner
dy	real	I	Y world coordinate value of the lower right corner
error_status	integer	O	Error status
pa_cols	integer	O	Pixel array columns
pa_rows	integer	O	Pixel array rows

Appendix A

DEC GKS Function Names and FORTRAN Binding Function Names

Appendix A lists the DEC GKS function names, which use the GKS$ interface, and the corresponding FORTRAN binding name or names (if applicable). For a listing of the FORTRAN binding constant names, refer to Appendix B, DEC GKS Constants, in the *DEC GKS Reference Manual.*

Table A–1: DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
Control Functions:	
GKS$ACTIVATE_WS	GACWK
GKS$CLEAR_WS	GCLRWK
GKS$CLOSE_GKS	GCLKS
GKS$CLOSE_WS	GCLWK
GKS$DEACTIVATE_WS	GDAWK
GKS$ESCAPE	GESC
GKS$MESSAGE_GKS	GMSG (FORTRAN 77)
N/A	GMSGS(FORTRAN 77 subset)
GKS$OPEN_GKS	GOPKS
GKS$OPEN_WS	GOPWK
GKS$REDRAW_SEG_ON_WS	GRSGWK

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
GKS$SET_DEFER_STATE	GSDS
GKS$UPDATE_WS	GUWK
Output Functions:	
GKS$CELL_ARRAY	GCA
GKS$FILL_AREA	GFA
GKS$GDP	GGDP
GKS$POLYLINE	GPL
GKS$POLYMARKER	GPM
GKS$TEXT	GTX (FORTRAN 77)
N/A	GTXS (FORTRAN 77 subset)
Attribute Functions:	
GKS$SET_ASF	GSASF
GKS$SET_COLOR_REP	GSCR
GKS$SET_FILL_COLOR_INDEX	GSFACI
GKS$SET_FILL_INDEX	GSFAI
GKS$SET_FILL_INT_STYLE	GSFAIS
GKS$SET_FILL_REP	GSFAR
GKS$SET_FILL_STYLE_INDEX	GSFASI
GKS$SET_PAT_REF_PT	GSPARF
GKS$SET_PAT_REP	GSPAR
GKS$SET_PAT_SIZE	GSPA
GKS$SET_PICK_ID	GSPKID
GKS$SET_PLINE_COLOR_INDEX	GSPLCI
GKS$SET_PLINE_INDEX	GSPLI
GKS$SET_PLINE_LINETYPE	GSLN
GKS$SET_PLINE_LINEWIDTH	GSLWSC

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
GKS$SET_PLINE_REP	GSPLR
GKS$SET_PMARK_COLOR_INDEX	GSPMCI
GKS$SET_PMARK_INDEX	GSPMI
GKS$SET_PMARK_REP	GSPMR
GKS$SET_PMARK_SIZE	GSMKSC
GKS$SET_PMARK_TYPE	GSMK
GKS$SET_TEXT_ALIGN	GSTXAL
GKS$SET_TEXT_COLOR_INDEX	GSTXCI
GKS$SET_TEXT_EXPFAC	GSCHXP
GKS$SET_TEXT_FONTPREC	GSTXFP
GKS$SET_TEXT_HEIGHT	GSCHH
GKS$SET_TEXT_INDEX	GSTXI
GKS$SET_TEXT_PATH	GSTXP
GKS$SET_TEXT_REP	GSTXR
GKS$SET_TEXT_SPACING	GSCHSP
GKS$SET_TEXT_UPVEC	GSCHUP
Transformation Functions:	
GKS$SELECT_XFORM	GSELNT
GKS$SET_CLIPPING	GSCLIP
GKS$SET_WINDOW	GSWN
GKS$SET_VIEWPORT	GSVP
GKS$SET_VIEWPORT_PRIORITY	GSVPIP
GKS$SET_WS_WINDOW	GSWKWN
GKS$SET_WS_VIEWPORT	GSWKVP

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
Segment Functions:	
GKS$ACCUM_XFORM_MATRIX	GACTM
GKS$ASSOC_SEG_WITH_WS	GASGWK
GKS$CLOSE_SEG	GCLSG
GKS$COPY_SEG_TO_WS	GCSGWK
GKS$CREATE_SEG	GCRSG
GKS$DELETE_SEG	GDSG
GKS$DELETE_SEG_FROM_WS	GDSGWK
GKS$EVAL_XFORM_MATRIX	GEVTM
GKS$INSERT_SEG	GINSG
GKS$RENAME_SEG	GRENSG
GKS$SET_SEG_DETECTABILITY	GSDTEC
GKS$SET_SEG_HIGHLIGHTING	GSHLIT
GKS$SET_SEG_PRIORITY	GSSGP
GKS$SET_SEG_VISIBILITY	GSVIS
GKS$SET_SEG_XFORM	GSSGT
Input Functions:	
GKS$AWAIT_EVENT	GWAIT
GKS$FLUSH_DEVICE_EVENTS	GFLUSH
GKS$GET_CHOICE	GGTCH
GKS$GET_LOCATOR	GGTLC
GKS$GET_PICK	GGTPK
GKS$GET_STRING	GGTST (FORTRAN 77)
N/A	GGTST (FORTRAN 77 subset)
GKS$GET_STROKE	GGTSK
GKS$GET_VALUATOR	GGTVL
GKS$INIT_CHOICE	GINCH

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
GKS$INIT_LOCATOR	GINLC
GKS$INIT_PICK	GINPK
GKS$INIT_STRING	GINST (FORTRAN 77)
N/A	GINST (FORTRAN 77 subset)
GKS$INIT_STROKE	GINSK
GKS$INIT_VALUATOR	GINVL
GKS$REQUEST_CHOICE	GRQCH
GKS$REQUEST_LOCATOR	GRQLC
GKS$REQUEST_PICK	GRQPK
GKS$REQUEST_STRING	GRQST (FORTRAN 77)
N/A	GRQST (FORTRAN 77 subset)
GKS$REQUEST_STROKE	GRQSK
GKS$REQUEST_VALUATOR	GRQVL
GKS$SAMPLE_CHOICE	GSMCH
GKS$SAMPLE_LOCATOR	GSMLC
GKS$SAMPLE_PICK	GSMPK
GKS$SAMPLE_STRING	GSMST (FORTRAN 77)
N/A	GGTST (FORTRAN 77 subset)
GKS$SAMPLE_STROKE	GSMSK
GKS$SAMPLE_VALUATOR	GSMVL
GKS$SET_CHOICE_MODE	GSCHM
GKS$SET_LOCATOR_MODE	GSLCM
GKS$SET_PICK_ID	GSPKID
GKS$SET_PICK_MODE	GSPKM
GKS$SET_STRING_MODE	GSSTM
GKS$SET_STROKE_MODE	GSSKM
GKS$SET_VALUATOR_MODE	GSVLM

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
Inquiry Functions	
GKS Description Table:	
GKS$INQ_LEVEL	GQLVKS
GKS$INQ_MAX_XFORM	GQMNTN
GKS$INQ_WS_MAX_NUM	GQWKM
GKS$INQ_WSTYPE_LIST	GQEWK
GKS State List:	
GKS$INQ_ACTIVE_WS	GQACWK
GKS$INQ_CLIP	GQCLIP
GKS$INQ_CURRENT_XFORMNO	GQCNTN
GKS$INQ_INDIV_ATTB	GQLN
N/A	GQLWSC
N/A	GQPLCI
N/A	GQMK
N/A	GQMKSC
N/A	GQPMCI
N/A	GQTXFP
N/A	GQCHXP
N/A	GQCHSP
N/A	GQTXCI
N/A	GQFAIS
N/A	GQFASI
N/A	GQFACI
N/A	GQASF
GKS$INQ_INPUT_QUEUE_OVERFLOW	GQIQOV
GKS$INQ_MORE_SIMUL_EVENTS	GQSIM
GKS$INQ_NAME_OPEN_SEG	GQOPSG

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
GKS$INQ_OPERATING_STATE	GQOPS
GKS$INQ_OPEN_WS	GQOPWK
GKS$INQ_PICK_ID	GQPKID
GKS$INQ_PRIM_ATTB	
N/A	GQPMI
N/A	GQTXI
N/A	GQCHH
N/A	GQCHUP
N/A	GQCHW
N/A	GQCHB
N/A	GQTXP
N/A	GQTXAL
N/A	GQFAI
N/A	GQPA
N/A	GQPARF
GKS$INQ_SEG_NAMES	GQSGUS
GKS$INQ_XFORM	GQNT
GKS$INQ_XFORM_LIST	GQENTN
Workstation State List:	
GKS$INQ_CHOICE_STATE	GQCHS
GKS$INQ_COLOR_INDEXES	GQECI
GKS$INQ_COLOR_REP	GQCR
GKS$INQ_FILL_INDEXES	GQEFAI
GKS$INQ_FILL_REP	GQFAR
GKS$INQ_LOCATOR_STATE	GQLCS
GKS$INQ_PAT_INDEXES	GQEPAI
GKS$INQ_PAT_REP	GQPAR

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
GKS$INQ_PICK_STATE	GQPKS
GKS$INQ_PLINE_INDEXES	GQEPLI
GKS$INQ_PLINE_REP	GQPLR
GKS$INQ_PMARK_INDEXES	GQEPMI
GKS$INQ_PMARK_REP	GQPMR
GKS$INQ_SEG_NAMES_ON_WS	GQSGWK
GKS$INQ_STRING_STATE	GQSTS (FORTRAN 77)
GKS$INQ_STRING_STATE	GQSTS (FORTRAN 77 subset)
GKS$INQ_STROKE_STATE	GQSKS
GKS$INQ_TEXT_EXTENT	GQTXX (FORTRAN 77)
N/A	GQTXXS (FORTRAN 77 subset)
GKS$INQ_TEXT_INDEXES	GQETXI
GKS$INQ_TEXT_REP	GQTXR
GKS$INQ_VALUATOR_STATE	GQVLS
GKS$INQ_WS_DEFER_AND_UPDATE	GQWKDU
GKS$INQ_WS_STATE	GQWKS
GKS$INQ_WS_TYPE	GQWKC
GKS$INQ_WS_XFORM	GQWKT
Workstation Description Table:	
GKS$INQ_AVAIL_GDP	GQEGDP
GKS$INQ_COLOR_FAC	GQCF
GKS$INQ_DEF_CHOICE_DATA	GQDCH
GKS$INQ_DEF_DEFER_STATE	GQDDS
GKS$INQ_DEF_PICK_DATA	GQDPK
GKS$INQ_DEF_STRING_DATA	GQDST
GKS$INQ_DEF_STROKE_DATA	GQDSK
GKS$INQ_DEF_VALUATOR_DATA	GQDVL

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
GKS$INQ_DYN_MOD_SEG_ATTB	GQDSGA
GKS$INQ_DYN_MOD_WS_ATTB	GQDWKA
GKS$INQ_FILL_FAC	GQFAF
GKS$INQ_GDP	GQGDP
GKS$INQ_INPUT_DEV	GQLI
GKS$INQ_MAX_DS_SIZE	GQDSP
GKS$INQ_MAX_WS_STATE_TABLE	GQLWK
GKS$INQ_PAT_FAC	GQPAF
GKS$INQ_PLINE_FAC	GQPLF
GKS$INQ_PMARK_FAC	GQPMF
GKS$INQ_PREDEF_COLOR_REP	GQPCR
GKS$INQ_PREDEF_FILL_REP	GQPFAR
GKS$INQ_PREDEF_PAT_REP	GQPPAR
GKS$INQ_PREDEF_PLINE_REP	GQPPLR
GKS$INQ_PREDEF_PMARK_REP	GQPPMR
GKS$INQ_PREDEF_TEXT_REP	GQPTXR
GKS$INQ_SEG_PRIORITY	GQSGP
GKS$INQ_TEXT_FAC	GQTXF
GKS$INQ_WS_CATEGORY	GQWKCA
GKS$INQ_WS_CLASSIFICATION	GQWKCL
Segment:	
GKS$INQ_SEG_ASSOC_WS	GQASWK
GKS$INQ_SEG_ATTB	GQSGA

(continued on next page)

Table A–1 (Cont.): DEC GKS Function Names and Corresponding FORTRAN Binding Names

DEC GKS Function	FORTRAN Binding Function(s)
Pixel:	
GKS$INQ_PIXEL	GQPX
GKS$INQ_PIXEL_ARRAY	GQPXA
GKS$INQ_PIXEL_ARRAY_DIM	GQPXAD
Metafile Functions:	
GKS$GET_ITEM	GGTITM
GKS$INTERPRET_ITEM	GIITM
GKS$READ_ITEM	GRDITM
GKS$WRITE_ITEM	GWITM
Error Functions:	
GKS$EMERGENCY_CLOSE	GECLKS
GKS$ERROR_HANDLER	GERHND
GKS$LOG_ERROR	GERLOG
Data Record Functions:	
NA	GPREC
NA	GUREC

Appendix B

DEC GKS FORTRAN Constants

This appendix lists the DEC GKS FORTRAN binding constants, and a short description of each. Using constants in your DEC GKS programs adds to the legibility of your program.

To use constants in your program, you must include a definitions file in your code. Table B–1 lists the DEC GKS constant names.

Table B–1: DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
Input Mode Types:		
GREQU	0	Request mode
GSAMPL	1	Sample mode
GEVENT	2	Event mode
Echo States:		
GNECHO	0	Echo disabled
GECHO	1	Echo enabled
Input Status Types:		
GNONE	0	No input obtained
GOK	1	Input obtained

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GNCHOI	2	Input is NOCHOICE
GNPICK	2	Input is NOPICK
GKS Status Types:		
GGKCL	0	GKS closed
GGKOP	1	GKS open
GWSOP	2	At least one workstation open
GWSAC	3	At least one workstation active
GSGOP	4	At least one segment open
GKS Level Types:		
GLMA	-3	level ma
GLMB	-2	level mb
GLMC	-1	level mc
GL0A	0	level 0a
GL0B	1	level 0b
GL0C	2	level 0c
GL1A	3	level 1a
GL1B	4	level 1b
GL1C	5	level 1c
GL2A	6	level 2a
GL2B	7	level 2b
GL2C	8	level 2c
Text Path Types:		
GRIGHT	0	Right
GLEFT	1	Left
GUP	2	Up
GDOWN	3	Down

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
Text Horizontal Alignment Types:		
GAHNOR	0	Normal
GALEFT	1	Left
GACENT	2	Center
GARITE	3	Right
Text Vertical Alignment Types:		
GAVNOR	0	Normal
GATOP	1	Top
GACAP	2	Cap
GAHALF	3	Half
GABASE	4	Base
GABOTT	5	Bottom
Standard Line Types:		
GLSOLI	1	Solid
GLDASH	2	Dashed
GLDOT	3	Dotted
GLDASD	4	Dash-Dotted
Standard Marker Types:		
GPOINT	1	Dot (.)
GPLUS	2	Plus (+)
GAST	3	Asterisk (*)
GOMARK	4	Circle (o)
GXMARK	5	Diagonal Cross (X)

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
Text Precision Types:		
GSTRP	0	String
GCHARP	1	Char
GSTRKP	2	Stroke
Fill Area Types:		
GHOLLO	0	Hollow
GSOLID	1	Solid
GPATTR	2	Pattern
GHATCH	3	Hatched
Attribute Source States:		
GBUNDL	0	Bundled
GINDIV	1	Individual
Clear Screen States:		
GCONDI	0	Clear conditionally
GALWAY	1	Clear always
Error Handling States:		
GCONDI	0	Clear conditionally
GALWAY	1	Clear always
Clipping States:		
GNCLIP	0	Clipping off
GCLIP	1	Clipping on
Workstation Types:		
GWSDEF	0	Default workstation type
GMOUTP	2	GKS output metafile
GMINPT	3	GKS input metafile

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GWSWIS	5	GKS workstation independent segment storage
GCGMO	7	CGM output metafile
GVTO	10	DIGITAL VT125 (output only)
GV125C	11	DIGITAL VT125 with color option
GV125B	12	Black and white DIGITAL VT125
GV240C	13	DIGITAL VT240 with color option
GV240B	14	Black and white DIGITAL VT240
GV330	16	DIGITAL VT330 (black and white)
GV340	17	DIGITAL VT340 (color)
GLCP01	15	DIGITAL LCG01 printer
GLCG01	15	DIGITAL LCG01 printer
GLA34	31	DIGITAL LA34 with graphics option
GLA100	31	DIGITAL LA100
GLA50	32	DIGITAL LA50 with 2:1 aspect ratio
GLA75	35	DIGITAL LA75
GLA12	33	DIGITAL LA12 with graphics option
GLA210	34	DIGITAL LA210
GLN03P	38	DIGITAL LN03 PLUS
GVSI	42	DIGITAL VAXstation I
GVSII	41	DIGITAL VAXstation II
GVSGPX	41	DIGITAL VAXstation II/GPX
GV2000	41	DIGITAL VAXstation 2000
GV3200	41	DIGITAL VAXstation 3200
GV3500	41	DIGITAL VAXstation 3500
GLVPA	51	DIGITAL LVP16 color plotter (8 1/2 by 11)
GLVPB	52	DIGITAL LVP16 color plotter (11 by 17)
GHP747	51	Hewlett Packard HP7475

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GHP755	53	HP7550 pen plotter
GHP758	54	HP7580 pen plotter
GLGMPS	55	Lasergraphics film recorder
GH7585	56	HP7585 pen plotter
GPTSC	61	PostScript graphics handler
GTEKO	70	TEKTRONIX—4014 (output only)
GT4014	72	TEKTRONIX—4014
GT41O	80	TEKTRONIX—4107 (output only)
GT4107	82	TEKTRONIX—4107
GLJ250	91	DIGITAL LJ250 90 DPI
GLJ25X	92	DIGITAL LJ250 180 DPI
GDECWO	210	DECwindows output
GDECW	211	DECwindows
GDECWD	212	DECwindows drawable
GDECWN	219	DECwindows no toolkit
GCONID	0	Default connection identifier
Workstation States:		
GINACT	0	Inactive
GACTIV	1	Active
Deferral State Types:		
GASAP	0	As soon as possible
GBNIG	1	Before the next global interaction
GBNIL	2	Before the next local interaction
GASTI	3	At some time

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
Implicit Regeneration States:		
GSUPPD	0	Implicit regeneration suppressed
GALLOW	1	Implicit regeneration allowed
Regeneration Flag States:		
GPOSTP	0	Implicit regeneration postponed
GPERFO	1	Implicit regeneration performed
New Frame Action States:		
GNO	0	No new frame action on update
GYES	1	New frame action on update
Action Pending States:		
GNPEND	0	Not pending
GPEND	1	Pending
Representation States:		
GSET	0	Type of returned value is set
GREALI	1	Type of returned value is realized
Workstation Category Types:		
GOUTPT	0	Output
GINPUT	1	Input
GOUTIN	2	Out/In
GWISS	3	Workstation independent segment storage
GMO	4	Metafile output
GMI	5	Metafile input

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
Workstation Class Types:		
GVECTR	0	Vector
GRASTR	1	Raster
GOTHWK	2	Other device
Device Coordinate States:		
GMETRE	0	Meters
GOTHU	1	Other units
Workstation Color States:		
GMONOC	0	Monochrome
GCOLOR	1	Color
Pixel Validity States:		
GABSNT	0	Invalid values absent
GPRSNT	1	Invalid values present
Display Surface States:		
GNEMPT	0	Display surface not empty
GEMPTY	1	Display surface empty
World or NDC Coordinate State:		
GWC	0	WC coordinates
GNDC	1	NDC coordinates
Dynamic Modification States:		
GIRG	0	Implicit regeneration necessary
GIMM	1	Immediate

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
Input Priority States:		
GHIGHR	0	Relative input priority higher
GLOWER	1	Relative input priority lower
Attribute Control Function Types:		
GCURNT	0	Input data record current values
GSPEC	1	Input data record specified values
GPLINE	0	Data record polyline control flag
GFILLA	1	Data record fill area control flag
GDP Bundle Types:		
GPLATT	0	GDP polyline bundle
GPMATT	1	GDP polymarker bundle
GTXATT	2	GDP text bundle
GFAATT	3	GDP fill area bundle
Segment Attribute States:		
GINVIS	0	Set to invisible
GVISI	1	Set to visible
GUNDET	0	Set to undetectable
GDETEC	1	Set to detectable
GNORML	0	Primitives are not highlighted
GHILIT	1	Primitives are highlighted
Input Class States:		
GNCLAS	0	No input class
GLOCAT	1	Locator input class

(continued on next page)

Table B–1 (Cont.): DEC GKS FORTRAN Binding Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GSTROK	2	Stroke input class
GVALUA	3	Valuator input class
GCHOIC	4	Choice input class
GPICK	5	Pick input class
GSTRIN	6	String input class
GVIEW	7	Viewport input class
Simultaneously Generated Events:		
GKS$K_NOMORE_EVENTS	0	No more simultaneously generated events
GKS$K_MORE_EVENTS	1	More simultaneously generated events
Line Cap Types:		
GLCBUT	2	Line cap type butted
GLCRND	3	Line cap type rounded
GLCSQR	4	Line cap type square
Line Join Types:		
GLJMTR	2	Line join type mitre
GLJRND	3	Line join type round
GLJBVL	4	Line join type bevel
Writing Modes:		
GWMCMT	2	Complement writing mode
GWMERS	3	Erase writing mode
GWMOVY	4	Overlay writing mode

Table B–2 lists the DEC GKS specific constants.

Table B–2: DEC GKS Specific Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
Line Types:		
GLDS2D	–1	Line type dash-2-dots
GLDS3D	–2	Line type dash-3-dots
GLLGDS	–3	Line type long-dash
GLLSDS	–4	Line type long-short-dash
GLSPDS	–5	Line type spaced-dash
GLSPDT	–6	Line type spaced-dot
GLDBDT	–7	Line type double dots
GLTPDT	–8	Line type triple dots
Marker Types:		
GMSCIR	–1	Marker type solid circle
GMTRU	–2	Marker type hollow up triangle
GMSTRU	–3	Marker type solid up triangle
GMTRD	–4	Marker type hollow down triangle
GMSTRD	–5	Marker type solid down triangle
GMSQ	–6	Marker type hollow square
GMSSQ	–7	Marker type solid square
GMBT	–8	Marker type hollow bow tie
GMSBT	–9	Marker type solid bow tie
GMHG	–10	Marker type hollow hour glass
GMSHG	–11	Marker type solid hour glass
GMDIA	–12	Marker type hollow diamond
GMSDIA	–13	Marker type solid diamond

(continued on next page)

Table B–2 (Cont.): DEC GKS Specific Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GDPs:		
GGDISP	–100	Disjoint polyline
GGCCP	–101	Center and point on circle
GGC3P	–102	3 points on circle
GGCCR	–103	Center and radius of circle
GGC2PR	–104	2 points and radius of circle
GGAC2P	–106	Center and two points of the arc
GGA3P	–107	3 points of arc
GGACVR	–108	Center and 2 vector radius of arc
GGA2PR	–109	Two points and radius of the arc
GGACPA	–110	Center point and angle for arc
GGECA	–111	Center and axes of ellipse
GGEFP	–113	Focii and point of ellipse
GGEACA	–114	Center, 2 vectors of elliptic arc
GGEAFP	–116	Focci, 2 points on elliptic arc
GGR2P	–125	Rectangle by 2 points
GGRT	–231	Restricted text
GGFAS	–332	Fill area set
GGFCCP	–333	Fill circle using center, point
GGFC3P	–334	Fill circle using 3 points
GGFCCR	–335	Fill circle using center and radius
GGFCPR	–336	Fill circle using 2 points and radius
GGFACP	–338	Fill arc using center and two points of the arc
GGFA3P	–339	Fill arc using 3 points
GGFACV	–340	Fill arc using 2 vectors and radius
GGFAPR	–341	Fill arc using two points and radius of the arc
GGFACA	–342	Fill arc using center, point, angle

(continued on next page)

Table B–2 (Cont.): DEC GKS Specific Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GGFECA	–343	Fill ellipse using center, axes
GGFEFP	–345	Fill ellipse using focii, point
GGFEACA	–346	Fill elliptic arc using center, axes, 2 vectors
GGFEAF	–348	Fill elliptic arc using focii, 2 points
GGFR2P	–349	Fill rectangle using 2 points
GGIA	–400	Packed cell array GDP
Escapes:		
GESP	–100	Set speed
GEP	–101	Print
GEB	–103	Beep
GEPOPW	–106	Pop workstation
GEPSHW	–107	Push workstation
GESEHM	–108	Set Error Handling Mode
GESVE	–109	Set viewport event
GEAWC	–110	Associate a conid with a workstation
GESWM	–150	Set writing mode
GESLC	–151	Set line cap
GESLJ	–152	Set line join
GESEC	–153	Set edge control flag in GKS state list
GESET	–154	Set edge type in GKS state list
GESEW	–155	Set edge width scale factor in GKS state list
GESECI	–156	Set edge color index in GKS state list
GESEI	–157	Set edge index in GKS state list
GESEA	–158	Set aspect source flag entries in GKS state list
GEBTB	–160	Begin transformation block
GEETB	–161	End transformation block
GESSHM	–162	Set segment highlighting method

(continued on next page)

Table B–2 (Cont.): DEC GKS Specific Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GESHM	–163	Set highlighting method
GESER	–200	Set edge representation
GESFN	–201	Set font name
GESWT	–202	Set window title
GEIC	–250	Inquire cursor
GEIWM	–251	Inquire writing mode
GEILC	–252	Inquire line cap
GEILJ	–253	Inquire line join
GEIEA	–254	Inquire current edge attributes
GEIS	–300	Inquire speed
GEISE	–303	Inquire segment extent
GEILEI	–302	Inquire list of edge indexes for workstation
GEIWID	–304	Inquire window identifiers
GEISHM	–305	Inquire segment highlighting
GEIHM	–306	Inquire highlighting method
GEILE	–350	Inquire list of escapes
GEIDS	–351	Inquire default display speed
GEILCJ	–352	Inquire cap join facility
GEIEF	–354	Inquire edge facilities
GEIPER	–355	Inquire predefined edge representation for workstation type and edge index
GEIMEB	–356	Inquire maximum number of edge bundle entries
GEILH	–358	Inquire list highlighting
GEMNW	–400	Map WC to NDC
GEMDN	–401	Map NDC to DC
GEMWN	–402	Map NDC to WC
GEMND	–403	Map DC to NDC
GEIGEX	–404	Inquire GDP extent

(continued on next page)

Table B–2 (Cont.): DEC GKS Specific Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
Arc Types:		
GATOPN	1	Arc type open
GATPIE	2	Arc type pie
GATCHRD	3	Arc type chord
CGM Encoding Bit Masks:		
GMCHAR	NA	Character
GMCLTX	NA	Clear text
ReGIS Bit Masks:		
GMV125	NA	VT125 color, output only
GMV12B	NA	VT125 black/white, output only
GMV240	NA	VT240 color, output only
GMV24B	NA	VT240 black/white, output only
GMV330	NA	VT330 black/while, output only
GMV340	NA	VT340 color, output only
GMNPTR	65536	VT330 or VT340 that does not have a mouse
GVTCMR	16777216	VT330 or VT340 that saves the colormap
Paper Size Bit Masks:		
GSIZA	NA	8.5 x 11 inches
GSIZB	NA	11 x 17 inches
GSIZC	NA	17 x 22 inches
GSIZD	NA	22 x 34 inches
GSIZE	NA	34 x 44 inches
GSIZA0	NA	84.1 x 118.9 centimeters
GSIZA1	NA	59.4 x 84.1 centimeters
GSIZA2	NA	42 x 59.4 centimeters
GSIZA3	NA	29.7 x 42 centimeters

(continued on next page)

Table B–2 (Cont.): DEC GKS Specific Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GSIZA4	NA	21 x 29.7 centimeters
GSIZA5	NA	14.8 x 21 centimeters
GSIZLG	NA	8.5 x 14 inches (legal size)
Paper Orientation Bit Masks:		
GPRTRT	NA	Portrait orientation
GLDSCP	NA	Landscape orientation
Color Mapping Bit Masks:		
GCMPHY	16777216	Use the physical color indexes
GCMVIR	0	Use the virtual color indexes
Dots Per Inch (DPI):		
GDI72	16777216	72 dots per inch
GDI90	0	90 dots per inch
GDI144	2097152	144 dots per inch
GDI180	50331648	180 dots per inch
Input on device handler:		
GKS$K_CURSOR_ MOVEMENT	1	Input key is a cursor movement
GKS$K_ CHARACTER	2	Input key is a character
GKS$K_POINT_ TRIGGER	3	Input key is a point trigger
GKS$K_ TERMINATION_ TRIGGER	4	Input key is a termination trigger
GKS$K_DELETE_ KEY	5	Input key is delete

(continued on next page)

Table B–2 (Cont.): DEC GKS Specific Constant Names

DEC GKS FORTRAN Binding Constant	Value	Description
GKS$K_BREAK	6	Input key is break
GKS$K_CHOICE_NUMBER	7	Input key is a choice number
GKS$K_CYCLE	8	Input key is cycle
GKS$K_NOCYCLE	9	Input key is no cycle
GKS$K_STROKE_MEASURE	10	Input key is a stroke measure
GKS$K_TOGGLE_INSERT	11	Input key is insert toggle
GKS$K_RESTORE_INITIAL	12	Input key is to restore initial string
GKS$K_BUFFER_BEGINNING	13	Input key is move to beginning
GKS$K_BUFFER_END	14	Input key is move to end
GKS$K_VALUATOR_VALUE	15	Input key is a valuator measure
GKS$K_SIGNAL	16	Input signal occured
GKS$K_LOG_ERROR	17	Input error occured
Highlighting methods:		
GHMDEF	0	Default highlighting
GHMCOM	1	
GHMCOL	2	Color highlighting
GHMLIN	3	Line highlighting
GHMFIL	4	Fill area highlighting
GHMDUA	5	Dual highlighting

Appendix C

DEC GKS Error Messages

This appendix lists each of the DEC GKS error messages, the DEC GKS error numbers, and the VMS completion status codes.

The VMS completion status codes correspond to the longword condition value returned by each DEC GKS function. You can compare the completion status codes directly to the function return values. In this way, you do not have to directly access the individual bits of the returned longword condition value to determine the cause of the error.

If you choose, you can perform the normal VMS processing of the returned longword condition value by using LIB$SIGNAL, $GETMSG, or $PUTMSG. For detailed information concerning this type of processing, refer to the *VAX/VMS Run-Time Library Routines Reference Manual.*

Some of the DEC GKS specific error messages substitute program information in the message text. In this appendix, the portion of the text to be substituted is shown as ****.

The following sections describe the DEC GKS error messages by category.

C.1 Implementation-Specific Errors

All of the DEC GKS specific errors are negative in number; their condition status codes read DECGKS$_ERROR_NEG_number. These errors are either errors or fatal errors as described.

–2 Requested color map could not be created as specified in routine ****

DECGKS$_ERROR_NEG_2:

Error: Specified color map is too large.

User Action: Check to make sure that you specified the correct color map size and type (either physical or virtual). Keep in mind the limitations of your VAXstation when reserving color indexes.

–3 Invalid data in workstation description file in routine ****

DECGKS$_ERROR_NEG_3:

Error: Workstation description file contains invalid data.

User Action: Make sure that the format of your description file is valid for your particular workstation.

–4 Invalid bit mask in workstation type in routine ****

DECGKS$_ERROR_NEG_4:

Error: The high word of the workstation type value is invalid.

User Action: Check to make sure that you specified a bit mask workstation type value that is valid for your workstation, and that you are running your program on the expected type of workstation.

–5 Bad string addresses found writing choice data record in routine ****

DECGKS$_ERROR_NEG_5:

Error: Illegal array of string pointers passed to the choice data record in
routine ****

User Action: Make sure that you properly initialized the arrays containing string addresses and string lengths. Also, make sure that you have declared a buffer to hold choice strings, and that your string address array contains addresses of the elements in your choice string array. For more information, refer to the program example for GQDCH in Chapter 12, Inquiry Functions, in the *DEC GKS Reference Manual.*

–6 Echo area is too narrow for data in routine ****

DECGKS$_ERROR_NEG_6:

Error: The specified input echo area minimum and maximum X values are too close in proximity.

User Action: Make sure that you did not swap X and Y values, and that your specified X values are of a greater distance from each other.

–7 Maximum number of representable choices exceeded in routine ****

DECGKS$_ERROR_NEG_7:

Error: The number of requested choices is too large for the workstation type.

User Action: You can use GQDCH to obtain the maximum choices available for your workstations, and then break your menu into two smaller menus.

–8 Echo area is too short for data in routine ****

DECGKS$_ERROR_NEG_8:

Error: The specified input echo area minimum and maximum Y values are too close in proximity.

User Action: Make sure that you did not swap X and Y values, and that your specified Y values are of a greater distance from each other.

–9 Binary format and integer number representation not supported in this implementation of GKS in routine ****

DECGKS$_ERROR_NEG_9:

Error: You opened a metafile of an incompatible type.

User Action: Check the metafile type.

–10 Invalid value specified for ASF in routine ****

DECGKS$_ERROR_NEG_10:

Error: You specified an incorrect value within the aspect source flag array.

User Action: Check the array to make sure that it has 13 elements and that its elements only contain the value GBUNDL (0) or GINDIV (1).

–11 Invalid value specified for fill area interior style in routine ****

DECGKS$_ERROR_NEG_11:

Error: You did not specify a proper integer value for an interior style argument.

User Action: Make sure that you passed one of the values GHOLLO (0), GSOLID (1), GPATTR (2), or GHATCH (3).

–12 Invalid value specified for horizontal component of text alignment in routine ****

DECGKS$_ERROR_NEG_12:

Error: You did not specify a proper integer value for a horizontal text alignment argument.

User Action: Make sure that you passed one of the values GAHNOR (0), GALEFT (1), GACENT (2), or GARITE (3).

–13 Invalid value specified for vertical component of text alignment in routine ****

DECGKS$_ERROR_NEG_13:

Error: You did not specify a proper integer value for a vertical text alignment argument.

User Action: Make sure that you passed one of the values GAVNOR (0), GATOP (1), GACAP (2), GAHALF (3), GABASE (4), or GABOTT (5).

–14 Invalid value specified for text precision in routine ****

DECGKS$_ERROR_NEG_14:

Error: You did not specify a proper integer value for a text precision argument.

User Action: Make sure that you passed one of the values GSTRP (0), GCHARP (1), or GSTRKP (2).

–15 Invalid value specified for text path in routine ****

DECGKS$_ERROR_NEG_15:

Error: You did not specify a proper integer value for a text path argument.

User Action: Make sure that you passed one of the values GRIGHT (0), GLEFT (1), GUP (2), or GDOWN (3).

–16 Echo switch is invalid in routine ****

DECGKS$_ERROR_NEG_16:

Error: You did not specify a proper integer value for an echo switch in one of the arguments to the SET MODE input functions.

User Action: Make sure that you passed GNECHO (0) or GECHO (1). Also, if you used an inquiry function to obtain the echo switch, check to see that the arguments to the inquiry function are specified in the correct order.

–17 Inquired device values not set or realized in routine ****

DECGKS$_ERROR_NEG_17:

Error: You neglected to specify GSET or GREALI when calling an inquiry function.

User Action: Check the value type argument to make sure that it is either GSET or GREALI.

–18 The following error occurred when GKS was interpreting an item ****

DECGKS$_ERROR_NEG_18:

Error: An error occurred while interpreting a metafile item.

User Action: DEC GKS follows this error message with another message that signals the appropriate action.

–19 Invalid error status parameter specified in routine ****

DECGKS$_ERROR_NEG_19:

Error: You passed an illegal error code to GERLOG.

User Action: Make sure that the error code passed to GERLOG is one of the codes described in this appendix.

–20 GKS not in proper state: GKS in the ERROR state in routine ****

DECGKS$_ERROR_NEG_20:

Error: You attempted to execute a DEC GKS function other than an error-handling or inquiry function.

User Action: Remove all calls to DEC GKS functions, other than inquiry and error-handling function calls, from your error-handling code.

–21 Function is not supported in this level of GKS in routine ****

DECGKS$_ERROR_NEG_21:

User Action: Remove the call to the unsupported function.

–22 Invalid segment transformation in routine ****

DECGKS$_ERROR_NEG_22:

Error: You specified an invalid transformation matrix.

User Action: Check your calls to GEVTM and to GACTM to make sure that you passed valid transformation components. Also, make sure that you specified a transformation matrix to GSSGT or to GINSG.

–23 Invalid value specified for clipping flag in routine ****

DECGKS$_ERROR_NEG_23:

User Action: Make sure that you passed either the value GNCLIP (0) or GCLIP (1).

–24 Invalid value specified for viewport priority flag in routine ****

DECGKS$_ERROR_NEG_24:

User Action: Make sure that you passed either the value GHIGHER (0) or GLOWER (1).

–25 Invalid value specified for update workstation flag in routine ****

DECGKS$_ERROR_NEG_25:

User Action: Make sure that you passed either the value GPOSTP (0) or GPERFO (1).

–26 Invalid value specified for deferral mode in routine ****

DECGKS$_ERROR_NEG_26:

User Action: Make sure that you passed one of the values GASAP (0), GBNIG (1), GBNIL (2), or GASTI (3).

–27 Invalid value specified for regeneration mode in routine ****

DECGKS$_ERROR_NEG_27:

User Action: Make sure that you passed either the value GSUPPD (0) or GALLOW (1).

–28 Invalid value specified for expansion factor in routine ****

DECGKS$_ERROR_NEG_28:

User Action: Check to make sure that you specified a real number value greater than the value 0.0. The value 1.0 causes no expansion.

–29 Invalid data record size for specified prompt and echo type in routine ****

DECGKS$_ERROR_NEG_29:

User Action: Check to make sure that you specified a data record of the correct size as determined by your chosen prompt and echo type.

–30 Cannot load workstation handler: error during image activation in routine ****

DECGKS$_ERROR_NEG_30:

Error: DEC GKS could not activate your workstation handler's shareable image.

User Action: Make sure that your workstation handler is a valid, shareable image.

–31 Cannot load graphics handler: invalid DFT in routine ****

DECGKS$_ERROR_NEG_31:

Error: Your device function tables are incompatible.

User Action: You need to build your device function table again using the appropriate macro. For more information, refer to *Building a DEC GKS Device Handler System* or *Building a DEC GKS Workstation Handler System*.

–32 Font file for stroke precision text not found or unusable in routine ****

DECGKS$_ERROR_NEG_32:

Error: DEC GKS could not locate the specified stroke font.

User Action: Refer to the appropriate device-specific appendix in this manual to determine if the specified font is supported on your device. If you are not using a DEC GKS supported graphics handler, make sure that your handler defines the proper logical names, and that the logicals reference a valid file.

–33 Array descriptor is not acceptable in routine ****

DECGKS$_ERROR_NEG_33:

Error: An item in the array descriptor is either invalid or inconsistent.

User Action: Make sure that you have passed the array by descriptor and that you fill the descriptor with valid values. If you have, and you use an inquiry function to initialize the array variable, make sure that all of the arguments are specified to the inquiry function in the correct order. Also, if the array is passed to the CELL ARRAY function, make sure that you have declared a two-dimensional array.

–34 String length less than or equal to 0 in routine ****

DECGKS$_ERROR_NEG_34:

Error: You specified an invalid character string.

User Action: Check the declaration, definition, or assignment statements involving the character variable.

–35 Kernel has detected an unexpected error from a device handler in routine ****

DECGKS$_ERROR_NEG_35:

Error: The device handler encountered an error.

User Action: DEC GKS follows this error message with another message that signals the appropriate action.

–36 Cannot load device handler: error during image activation in routine ****

DECGKS$_ERROR_NEG_36:

Error: DEC GKS could not activate your device handler's shareable image

User Action: Make sure that your device handler is a valid, shareable image. This error message is specific to handlers that affect a device (VAXstations) as opposed to a graphics language (PostScript).

–37 Error in device handler during event flag allocation in routine ****

DECGKS$_ERROR_NEG_37:

Error: A graphics handler was unable to acquire all of its needed event flags.

User Action: The application must release event flags for use by the graphics handler.

–38 Error in device handler, cannot allocate device in routine ****

DECGKS$_ERROR_NEG_38:

Error: You used your graphics handler with an invalid physical device.

User Action: Make sure that you use the proper physical device or that you specify the correct workstation type value to GOPWK.

–39 Descriptor is not acceptable in routine ****

DECGKS$_ERROR_NEG_39:

User Action: Make sure that you have passed the variable by descriptor. If you have, and you use an inquiry function to initialize the variable, make sure that all of the arguments are specified to the inquiry function in the correct order.

–40 Illegal device pointer in routine ****

DECGKS$_ERROR_NEG_40:

User Action: Check your handler code for null pointers or otherwise invalid pointers.

–41 Driver handler WDT is invalid in routine ****

DECGKS$_ERROR_NEG_41:

Error: You illegally defined a workstation description table entry.

User Action: Check your workstation description table definitions for your graphics handler.

–42 Logical name for the list of workstation types, GKS$LIST_TYPES, could not be translated in routine ****

DECGKS$_ERROR_NEG_42:

Error: You improperly defined the logical name.

User Action: Make sure that the translation of GKS$LIST_TYPES is as expected.

–43 VAX Workstation Software is not present, workstation type is invalid in routine ****

DECGKS$_ERROR_NEG_43:

Explanation: Check to make sure either that you specify the correct workstation type when opening a non-VAXstation workstation, or that you passed a correct workstation type value to one of the workstation description table or state list inquiry functions. If you are working on a MicroVAX, make sure that you install the VAXstation Windowing Software.

The following errors are fatal errors. Should one occur, submit a Software Performance Report (SPR) indicating the error number, corresponding message, and any relevant particulars. For more information concerning SPRs, refer to the *DEC GKS Installation Guide*.

–90 Internal GKS error: Bad memory address freed in routine ****

DECGKS$_ERROR_NEG_90:

Fatal: DEC GKS memory data structures were corrupted.

User Action: Submit an SPR.

–91 Internal GKS error: Invalid function pointer parameter in error handler in routine ****

DECGKS$_ERROR_NEG_91:

Fatal: A DEC GKS internal data structure was corrupted.

User Action: Submit an SPR.

–92 Internal GKS error: Insufficient virtual memory in routine ****

DECGKS$_ERROR_NEG_92:

Fatal: DEC GKS was unable to allocate enough virtual memory.

User Action: Check to make sure that the problem is not caused by storing too much in segment storage or by defining a very large cell array. If you cannot reduce storage by checking segments and cell arrays, submit an SPR.

–93 Internal GKS error: Prompt and echo type not supported in routine ****

DECGKS$_ERROR_NEG_93:

Fatal:

User Action: Submit an SPR.

–94 Internal GKS error: Corrupted segment memory in routine ****

DECGKS$_ERROR_NEG_94:

Fatal:

User Action: Submit an SPR.

–95 Internal GKS error: Negative size passed to allocate memory in routine ****

DECGKS$_ERROR_NEG_95:

Fatal: An invalid size was passed to the DEC GKS memory allocation routines.

User Action: If you generate this error using a user-written graphics handler, make sure that the value of the local storage area is a valid value.

–96 Internal GKS error: Illegal number of points to device handler for rectangular polygon in routine ****

DECGKS$_ERROR_NEG_96:

Fatal:

User Action: Submit an SPR.

–97 Internal GKS error: Insufficient buffer size for translated logical name in routine ****

DECGKS$_ERROR_NEG_97:

Fatal:

User Action: Submit an SPR.

–98 Internal GKS error: Too many translations of logical name in routine ****

DECGKS$_ERROR_NEG_98:

Fatal: You may have recursively defined a logical name.

User Action: Check the currently defined logical names to see if all are properly defined. If you cannot locate an error, submit an SPR.

–99 Internal GKS error: Unable to reduce number of points in fill area to requested limit in routine ****

DECGKS$_ERROR_NEG_99:

Fatal:

User Action: Submit an SPR.

–100 Internal GKS error: Device handler received unexpected input access in routine ****

DECGKS$_ERROR_NEG_100:

Fatal:

User Action: Submit an SPR.

–150 Edge index is less than zero in routine ****

DECGKS$_ERROR_NEG_150:

User Action:

–151 Edge width calse factor is less than zero ****

DECGKS$_ERROR_NEG_151:

User Action:

–152 Text font/precision cannot be named in routine ****

DECGKS$_ERROR_NEG_152:

User Action:

–153 Text font name is invalid in routine ****

DECGKS$_ERROR_NEG_153:

User Action:

–154 A representation for the specified edge index has not been predefined on this workstation in routine ****

DECGKS$_ERROR_NEG_154:

User Action:

–155 Display speed is less than zero in routine ****

DECGKS$_ERROR_NEG_155:

User Action: Pass a positive real value to GESP.

–156 Loudness is outside range [0,1] in routine ****

DECGKS$_ERROR_NEG_156:

User Action: Pass a valid value to GEB.

–157 Duration is less than zero in routine ****

DECGKS$_ERROR_NEG_157:

User Action: Make sure that your duration value is greater than or equal to zero.

–158 GDP primitive is not defined by the supplied data in routine ****

DECGKS$_ERROR_NEG_158:

Error: DEC GKS is unable to form the desired primitive.

User Action: Refer to the error message listing in the description of the GDP that generated the error (Appendix I, DEC GKS GDPs and Escapes, in the *DEC GKS Reference Manual*). This listing gives specific information concerning the primitive you attempted to draw.

–159 Arc type is invalid in routine ****

DECGKS$_ERROR_NEG_159:

User Action: Refer to the error message listing in the description of the GDP that generated the error (Appendix I, DEC GKS GDPs and Escapes, in the *DEC GKS Reference Manual*). This listing gives specific information concerning the primitive you attempted to draw.

–160 Insufficient space in escape output data record arrays in routine ****

DECGKS$_ERROR_NEG_160:

Error: You passed addresses of arrays that were too small to contain the data to be written to them.

User Action: Pass addresses of larger array buffers in the last four components of the escape data record.

–161 Specified bounding box is too small in routine ****

DECGKS$_ERROR_NEG_161:

Error: You specified text attributes that were too large to fill the text in the bounding box (the extent rectangle).

User Action: Use a larger bounding box, or reduce the text height or the character expansion factor.

–300 Invalid value specified for highlighting in routine ****

DECGKS$_ERROR_NEG_300:

User Action: Make sure that you specify either GNORML (0) or GHILIT (1).

–301 Invalid value specified for visibility in routine ****

DECGKS$_ERROR_NEG_301:

User Action: Make sure that you specify either GINVIS (0) or GVISI (1).

–302 Invalid value specified for detectability in routine ****

DECGKS$_ERROR_NEG_302:

User Action:

User Action: Make sure that you specify either GUNDET (0) or GDETEC (1).

–303 Input device cannot be activated due to conflict with another input device that is currently active in routine ****

DECGKS$_ERROR_NEG_303:

User Action:

–304 Cannot set input device echo on due to conflict with other input devices active in the same echo area in routine ****

DECGKS$_ERROR_NEG_304:

User Action:

–306 The definition of GKS$HPGL_THRESHOLD is invalid (contains nonnumeric values) in routine ****

DECGKS$_ERROR_NEG_306:

User Action: Check the definition of GKS$HPGL_THRESHOLD and redefine to range 0 to 1023.

C.2 Operating State Errors

This section lists the errors that result when you call a function that is not permitted in the current operating state. For a list of the functions that you can or cannot call in a given DEC GKS operating state, refer to Chapter 4, Control Functions, in the *DEC GKS Reference Manual.*

1 GKS not in proper state: GKS shall be in the state GKCL in routine ****

GKS$_ERROR_1:

Error: You called a function unsupported in the current operating state.

User Action: Call the appropriate DEC GKS control function to change the current state. (You must call GCLKS before the current DEC GKS state changes to GGKCL.)

2 GKS not in proper state: GKS shall be in the state GKOP in routine ****

GKS$_ERROR_2:

Error: You called a function unsupported in the current operating state.

User Action: Call the appropriate DEC GKS control function to change the current state. (You must call either the function GOPKS or GCLWK before the DEC GKS state changes to GGKOP.)

3 GKS not in proper state: GKS shall be in the state WSAC in routine ****

GKS$_ERROR_3:

Error: You called a function unsupported in the current state.

User Action: Call the appropriate DEC GKS control function to change the current state. (You must call either the function GACWK or GCLSG before the DEC GKS state changes to GWSAC.)

4 GKS not in proper state: GKS shall be in the state SGOP in routine ****

GKS$_ERROR_4:

Error: You called a function unsupported in the current state.

User Action: Call the appropriate DEC GKS control function to change the current state. (You must call the function GCRSG before the DEC GKS state changes to GSGOP.)

5 GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP in routine ****

GKS$_ERROR_5:

Error: You called a function unsupported in the current state.

User Action: Call the appropriate DEC GKS control function to change the current state. (You must call the function GACWK before the DEC GKS state changes to GWSAC.)

6 GKS not in proper state: GKS shall be in the state WSOP or in the state WSAC in routine ****

GKS$_ERROR_6:

Error: You called a function unsupported in the current state.

User Action: Call the appropriate DEC GKS control function to change the current state. (You must call the function GOPWK before the DEC GKS state changes to GWSOP.)

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP in routine ****

GKS$_ERROR_7:

Error: You called a function unsupported in the current state.

User Action: Call the appropriate DEC GKS control function to change the current state. (You must call the function GOPWK before the DEC GKS state changes to GWSOP.)

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP in routine ****

GKS$_ERROR_8:

Error: You called a function unsupported in the current state.

User Action: Call the appropriate DEC GKS control function to change the current state. (You must call the function GOPWK before the DEC GKS state changes to GWSOP.)

C.3 Workstation Errors

This section lists the errors that result when you call a DEC GKS function with invalid or undefined arguments pertaining to workstations.

20 Specified workstation identifier is invalid in routine ****

GKS$_ERROR_20:

User Action: Make sure that you have opened a workstation associated with that identifier, that you are not trying to generate output to an inactive workstation, that the arguments are presented in the right order, and if you are using a variable to specify the workstation identifier, that the variable is declared to be an integer.

21 Specified connection identifier is invalid in routine ****

GKS$_ERROR_21:

User Action: Make sure that the specified connection exists and is allocated to your process (by typing SHOW DEVICES at the DCL command line), that the workstation type supports the specified connection identifier (especially in the case of output-only workstations that write information to files, such as GVTO), and that the arguments are specified in the correct order.

22 Specified workstation type is invalid in routine ****

GKS$_ERROR_22:

User Action: Check to make sure that you passed either a DEC GKS constant (GWSDEF, GV240C), or the corresponding integer values.

23 Specified workstation type does not exist in routine ****

GKS$_ERROR_23:

Error: The implementation of GKS does not support a device handler associated with the identifier you passed.

User Action: Pass an identifier associated with a supported device. If you are using the constant GWSDEF, you should use GQWKC to check to see if DEC GKS supports the currently defined workstation type.

24 Specified workstation is open in routine ****

GKS$_ERROR_24:

Error: You tried to reopen a workstation.

User Action: Either remove the function call to GOPWK, or replace the incorrect workstation-type argument.

25 Specified workstation is not open in routine ****

GKS$_ERROR_25:

Error: You tried to input or generate output on a closed workstation.

User Action: Call GOPWK and pass the appropriate workstation identifier.

26 Specified workstation cannot be opened in routine ****

GKS$_ERROR_26:

User Action: Make sure that you specify valid workstation types, bit masks, or logical name definitions (GKS$CONID and GKS$WSTYPE), and make sure that the information corresponds to a supported, functional physical device.

27 Workstation Independent Segment Storage is not open in routine ****

GKS$_ERROR_27:

Error: You tried to copy, associate, or insert a segment from WISS to another workstation.

User Action: Make sure that you have opened WISS in a call to GOPWK, passing GWSWIS as an argument.

28 Workstation Independent Segment Storage is already open in routine ****

GKS$_ERROR_28:

User Action: Either remove the function call to GOPWK, or replace the incorrect workstation-type argument.

29 Specified workstation is active in routine ****

GKS$_ERROR_29:

Error: You tried to activate a workstation twice.

User Action: Either remove the function call to GACWK, or replace the incorrect workstation-type argument.

30 Specified workstation is not active in routine ****

GKS$_ERROR_30:

Error: You tried to generate output on an inactive workstation.

User Action: Call GACWK passing the appropriate workstation.

31 Specified workstation is of category MO in routine ****

GKS$_ERROR_31:

Error: You attempted to perform an operation that is not permissible on MO workstations.

User Action: Either remove the function call, change the state of the MO workstation, or check to see if you passed the correct arguments to GOPWK.

32 Specified workstation is not of category MO in routine ****

GKS$_ERROR_32:

User Action: Open and activate an MO workstation.

33 Specified workstation is of category MI in routine ****

GKS$_ERROR_33:

Error: You attempted to perform an operation that is not permissible on MI workstations.

User Action: Either remove the function call, change the state of the MI workstation, or check to see if you passed the correct arguments to GOPWK.

34 Specified workstation is not of category MI in routine ****

GKS$_ERROR_34:

Error: You tried to interpret a file that was not associated with an MI workstation.

User Action: Open a workstation of category MI.

35 Specified workstation is of category INPUT in routine ****

GKS$_ERROR_35:

Error: You attempted to perform an operation that is not permissible on workstations of category INPUT, such as generating output.

User Action: Either remove the function call, change the state of the INPUT workstation, or check to see if you passed the correct arguments to GOPWK.

36 Specified workstation is Workstation Independent Segment Storage in routine ****

GKS$_ERROR_36:

Error: You attempted to perform an operation that is not permissible on workstations of category WISS, such as requesting input.

User Action: Either remove the function workstation identifier or check to see if you passed the correct arguments to GOPWK.

37 Specified workstation is not of category OUTIN in routine ****

GKS$_ERROR_37:

Error: You attempted to perform an operation that is only permissible on workstations of category OUTIN.

User Action: Either remove the function call, open and activate an OUTIN workstation, or check to see if you passed the correct arguments to GOPWK.

38 Specified workstation is neither of category INPUT nor of category OUTIN in routine ****

GKS$_ERROR_38:

Error: You attempted to perform an operation that is only permissible on workstations of category INPUT and OUTIN, such as requesting input.

User Action: Either remove the function call, change the state of the INPUT workstation, or check to see if you passed the correct arguments to GOPWK.

39 Specified workstation is neither of category OUTPUT nor of category OUTIN in routine ****

GKS$_ERROR_39:

Error: You attempted to perform an operation that is only permissible on workstations of category OUTPUT or OUTIN, such as generating output.

User Action: Either remove the function call, open and activate a workstation of the correct category, or check to see if you passed the correct arguments to GOPWK.

40 Specified workstation has no pixel store readback capability in routine ****

GKS$_ERROR_40:

Error: You called one of the pixel inquiry functions for a device incapable of returning such information.

User Action: Either remove the function call, or make sure that you passed the correct workstation identifier.

41 Specified workstation type is not able to generate the specified generalized drawing primitive in routine ****

GKS$_ERROR_41:

User Action: Either remove the function call to GGDP, or make sure that you passed the correct GDP identifier.

42 Maximum number of simultaneously open workstations would be exceeded in routine ****

GKS$_ERROR_42:

User Action: You must remove the function call to GOPWK. You can use GQWKM to determine the maximum number of open workstations that DEC GKS supports.

43 Maximum number of simultaneously active workstations would be exceeded in routine ****

GKS$_ERROR_43:

User Action: You must remove the function call to GACWK. You can use GQWKM to determine the maximum number of active workstations that DEC GKS supports.

C.4 Transformation Errors

This section lists the errors that result when you call a DEC GKS transformation function with invalid or undefined arguments.

50 Transformation number is invalid in routine ****

GKS$_ERROR_50:

User Action: Either make sure that the arguments are specified in the correct order, that the transformation number is not negative, or that the transformation number is an integer.

51 Rectangle definition is invalid in routine ****

GKS$_ERROR_51:

Error: Either the normalization window or viewport is invalid.

User Action: Either make sure that you have not reversed the order of the X and Y argument values, that your coordinate values form a valid rectangle, and that your coordinate values are real numbers.

52 Viewport is not within the Normalized Device Coordinate unit square in routine ****

GKS$_ERROR_52:

Error: DEC GKS allows unclipped primitives to exceed the NDC unit square ([0,1] x [0,1]), but does not allow you to define a normalization viewport whose boundaries exceed this square.

User Action: Redefine the function normalization viewport.

53 Workstation window is not within the Normalized Device Coordinate unit square in routine ****

GKS$_ERROR_53:

User Action: Redefine the function normalization viewport to be within the NDC square ([0,1] x [0,1]).

54 Workstation viewport is not within the display space in routine ****

GKS$_ERROR_54:

User Action: Either make sure that you have not reversed the order of the X and Y argument values, that your coordinate values form a valid rectangle, and that your coordinate values are real numbers. You can use the function GQDSP to determine the maximum X and Y values of the device coordinate plane.

C.5 Output Attribute Errors

This section lists the errors that result when you call the DEC GKS output attribute functions with invalid or undefined arguments.

60 Polyline index is invalid in routine ****

GKS$_ERROR_60:

User Action: Make sure that the arguments are specified in the correct order and that the index is an integer.

61 A representation for the specified polyline index has not been defined on this workstation in routine ****

GKS$_ERROR_61:

User Action: Use GSPLR to define a representation for the index, or use another, predefined index value.

62 A representation for the specified polyline index has not been predefined on this workstation in routine ****

GKS$_ERROR_62:

User Action: Use GSPLR to define a representation for the index, or use another, predefined index value.

63 Specified linetype is equal to zero in routine ****

GKS$_ERROR_63:

User Action: Make sure that the order and the number of the arguments is correct. If you used an inquiry function to obtain a default line type, check the order of the arguments passed to the inquiry function.

64 Specified linetype is not supported on this workstation in routine ****

GKS$_ERROR_64:

Error: You specified a line type value that is workstation dependent but is not supported by the specified workstation.

User Action: Change the line type specification. You can use the function GQPLF to obtain a list of supported line types for a given workstation.

65 Linewidth scale factor is less than zero in routine ****

GKS$_ERROR_65:

User Action: Either change the scale factor, or check the order and the number of the specified arguments.

66 Polymarker index is invalid in routine ****

GKS$_ERROR_66:

User Action: Make sure that the arguments are specified in the correct order and that the index is an integer.

67 A representation for the specified polymarker index has not been defined on this workstation in routine ****

GKS$_ERROR_67:

User Action: Use GSPMR to define a representation for a given index, or use another, defined index value.

68 A representation for the specified polymarker index has not been predefined on this workstation in routine ****

GKS$_ERROR_68:

User Action: Use GSPMR to define a representation for a given index, or use another, predefined index value.

69 Specified marker type is equal to zero in routine ****

GKS$_ERROR_69:

User Action: Make sure that the order of the arguments is correct. If you used an inquiry function to obtain a default marker type, check the order of the arguments passed to the inquiry function.

70 Specified marker type is not supported on this workstation in routine ****

GKS$_ERROR_70:

Error: You specified a marker type value that is workstation dependent but is not supported by the specified workstation.

User Action: Change the marker type specification. You can use the function GQPMF to obtain a list of supported line types for a given workstation.

71 Marker size scale factor is less than zero in routine ****

GKS$_ERROR_71:

User Action: Either change the scale factor, or check the order and the number of the specified arguments.

72 Text index is invalid in routine ****

GKS$_ERROR_72:

User Action: Make sure that the arguments are specified in the correct order and that the index is an integer.

73 A representation for the specified text index has not been defined on this workstation in routine ****

GKS$_ERROR_73:

User Action: Use GSTXR to define a representation for the index value, or use another, predefined index value.

74 A representation for the specified text index has not been predefined on this workstation in routine ****

GKS$_ERROR_74:

User Action: Use GQTXR to define a representation for the index value, or use another, predefined index value.

75 Text font is equal to zero in routine ****

GKS$_ERROR_75:

User Action: Either change the font number, or check the order and the number of the specified arguments. If you used an inquiry function to obtain a default value, check the order and the number of the arguments passed to the inquiry function.

76 Requested text font is not supported for the specified precision on this workstation in routine ****

GKS$_ERROR_76:

User Action: Lower the precision or change the font number.

77 Character expansion factor is less than or equal to zero in routine ****

GKS$_ERROR_77:

User Action: Either change the expansion factor value or check the order and the number of the arguments. If you used an inquiry function to obtain a default value, check the order and the number of the arguments passed to the inquiry function.

78 Character height is less than or equal to zero in routine ****

GKS$_ERROR_78:

User Action: Either change the height value, or check the order and the number of the arguments. If you used an inquiry function to obtain a default value, check the order and the number of the arguments passed to the inquiry function.

79 Length of character up vector is zero in routine ****

GKS$_ERROR_79:

User Action: Change the character up vector, or check the order and the number of the arguments. If you used an inquiry function to obtain a default value, check the order and the number of the arguments passed to the inquiry function.

80 Fill area index is invalid in routine ****

GKS$_ERROR_80:

User Action: Make sure that the arguments are specified in the correct order and that the index is an integer.

81 A representation for the specified fill area index has not been defined on this workstation in routine ****

GKS$_ERROR_81:

User Action: Use GSFAR to define a representation for the given index value, or pass another, defined index value.

82 A representation for the specified fill area index has not been predefined on this workstation in routine ****

GKS$_ERROR_82:

User Action: Use GSFAR to define a representation for the given index value, or pass another, predefined index value.

83 Specified fill area interior style is not supported on this workstation in routine ****

GKS$_ERROR_83:

Error: You specified a fill area interior style value that is workstation-dependent but is not supported by the specified workstation.

User Action: Change the interior style specification. You can use

the function GQFAF to obtain a list of supported interior styles for a given workstation.

84 Style (pattern or hatch) index is equal to zero in routine ****

GKS$_ERROR_84:

User Action: Either change the style index, or check the order and the number of the specified arguments. If you used an inquiry function to obtain a style index, check the order and the number of the arguments passed to the inquiry function.

85 Specified pattern index is invalid in routine ****

GKS$_ERROR_85:

User Action: Make sure that the arguments are specified in the correct order and that the index is an integer.

86 Specified hatch style is not supported on this workstation in routine ****

GKS$_ERROR_86:

User Action: Either replace the hatch style index, or check the order and the number of the arguments. The inquiry function GQFAF returns the list of available hatch style indexes.

87 Pattern size value is not positive in routine ****

GKS$_ERROR_87:

User Action: Either alter the size of the pattern, or check the order and the number of the arguments. If you used an inquiry function to obtain the size of the pattern, check the order and the number of the arguments passed to the inquiry function.

88 A representation for the specified pattern index has not been defined on this workstation in routine ****

GKS$_ERROR_88:

User Action: Use GSPAR to define a representation for the pattern index, or pass another, defined index to the function.

89 A representation for the specified pattern index has not been predefined on this workstation in routine ****

GKS$_ERROR_89:

User Action: Use GSPAR to define a representation for the pattern index, or pass another, predefined index to the function.

90 Interior style PATTERN is not supported on this workstation in routine ****

GKS$_ERROR_90:

User Action: Specify another interior style to GSFAIS.

91 Dimensions of color array are invalid in routine ****

GKS$_ERROR_91:

Error: One or more of the arguments passed to GCA are invalid.

User Action: Make sure that the color array is a two-dimensional array. Also, make sure that you have not specified more rows and columns in the cell array that exist from the offset point to the end of the array. Also, make sure that the cell array contains integers representing colors supported on that workstation.

92 Color index is less than zero in routine ****

GKS$_ERROR_92:

User Action: Either remove the index, or check the order and the number of the arguments. If you used an inquiry function to obtain the index value, check the order and the number of the arguments passed to the inquiry function.

93 Color index is invalid in routine ****

GKS$_ERROR_93:

User Action: Make sure that the arguments are specified in the correct order and that the index is an integer.

94 A representation for the specified color index has not been defined on this workstation in routine ****

GKS$_ERROR_94:

User Action: Use GSCR to define a color representation for the index value, or pass another, defined index value.

95 A representation for the specified color index has not been predefined on this workstation in routine ****

GKS$_ERROR_95:

User Action: Use GSCR to define a color representation for the index value, or pass another, defined index value.

96 Color index is outside range [0,1] in routine ****

GKS$_ERROR_96:

User Action: Specify either the value 0 or 1 for the color index value.

97 Pick identifier is invalid in routine ****

GKS$_ERROR_97:

User Action: Either remove the call to GSPKIP or make sure that the pick identifier is an integer. If you obtained the pick identifier from an inquiry function, check the order and the number of the arguments passed to the inquiry function.

C.6 Output Function Errors

This section lists the errors that result when you call a DEC GKS output function with invalid or undefined arguments.

100 Number of points is invalid in routine ****

GKS$_ERROR_100:

Error: The number of points specified does not match the number of coordinate points passed.

User Action: Either alter the specified number of points, or alter the number of coordinate values contained in the arrays passed as arguments.

101 Invalid code in string in routine ****

GKS$_ERROR_101:

Error: Your text string contained characters that cannot be printed.

User Action: Remove the characters.

102 Generalized drawing primitive identifier is invalid in routine ****

GKS$_ERROR_102:

User Action: Specify another identifier or check to see if the identifier is an integer value.

103 Content of Generalized drawing primitive data record is invalid in routine ****

GKS$_ERROR_103:

User Action: Make sure that you passed a correct size as the data record size.

104 At least one active workstation is not able to generate the specified generalized drawing primitive in routine ****

GKS$_ERROR_104:

User Action: Deactivate the workstations that do not generate the GDPs, or redefine the GDP data record so that all of the workstations can generate the primitive.

105 At least one active workstation is not able to generate the specified generalized drawing primitive under the current transformation and clipping rectangle in routine ****

GKS$_ERROR_105:

User Action: Either redefine the current normalization transformation (creating a different clipping rectangle), or supply different world coordinate points so that the GDP falls within the current clipping rectangle.

C.7 Segment Function Errors

This section lists the errors that result when you call a DEC GKS segment function with invalid or undefined arguments.

120 Specified segment name is invalid in routine ****

GKS$_ERROR_120:

User Action: Either check the number and the order of the arguments or make sure that the segment name is an integer value. If you obtained the segment name from an inquiry function, check the order and the number of the arguments passed to the inquiry function.

121 Specified segment name is already in use in routine ****

GKS$_ERROR_121:

User Action: Either remove the call to GCRSG or check to make sure that you specified the correct segment name.

122 Specified segment does not exist in routine ****

GKS$_ERROR_122:

User Action: Either check the order and the number of the arguments or make sure that you specified an integer value as a segment name. If you used an inquiry function to obtain the segment name, check the order and the number of the arguments passed to the inquiry function.

123 Specified segment does not exist on specified workstation in routine ****

GKS$_ERROR_123:

User Action: Either remove the function call, or if the segment exists in WISS, associate the segment with the appropriate workstation.

124 Specified segment does not exist on Workstation Independent Segment Storage in routine ****

GKS$_ERROR_124:

User Action: You attempted to copy, associate, or insert a segment that is not stored in WISS. Either remove the function call or check to see that you specified the correct segment name.

125 Specified segment is open in routine ****

GKS$_ERROR_125:

User Action: Either remove the call to GCRSG or specify another segment name.

126 Segment priority is outside the range [0,1] in routine ****

GKS$_ERROR_126:

User Action: Change the specified segment priority. If you used an inquiry function to obtain the segment priority value, check the order and the number of the arguments passed to the inquiry function.

C.8 Input Function Errors

This section lists the errors that result when you call a DEC GKS input function with invalid or undefined arguments.

140 Specified input device is not present on workstation in routine ****

GKS$_ERROR_140:

User Action: Make sure that you specified the function that applies to the correct logical input device and the correct workstation identifier.

141 Input device is not in REQUEST mode in routine ****

GKS$_ERROR_141:

User Action: Use one of the SET MODE input functions to set request mode before using this logical input device.

142 Input device is not in SAMPLE mode in routine ****

GKS$_ERROR_142:

User Action: Use one of the SET MODE input functions to set to sample mode before using this logical input device.

143 EVENT and SAMPLE input mode are not available at this level of GKS in routine ****

GKS$_ERROR_143:

User Action: DEC GKS does not generate this error.

144 Specified prompt and echo type is not supported on this workstation in routine ****

GKS$_ERROR_144:

User Action: Make sure that the order of the arguments is correct or change the prompt and echo value. If you obtained the prompt and echo type from an inquiry function, check the order and the number of the arguments passed to the inquiry function.

145 Echo area is outside display space in routine ****

GKS$_ERROR_145:

User Action: Make sure that the specified coordinate points are real values that specify a valid rectangle on the display surface. If you used an inquiry function to obtain the echo area, check the order and the number of the arguments passed to the inquiry function.

146 Contents of input data record are invalid in routine ****

GKS$_ERROR_146:

User Action: Make sure that you specified the correct size of the data record, that the elements of the data record are of the correct data type, and that you have chosen the correct corresponding prompt and echo type. If you used an inquiry function to obtain the data record, check the order and number of the arguments passed to the inquiry function. Also, make sure that you have not specified input values that are not accepted by the particular device; you can check the device's capabilities by calling one of the DEFAULT DATA inquiry functions.

147 Input queue has overflowed in routine ****

GKS$_ERROR_147:

User Action: Check the input queue with greater frequency or flush the input queue.

148 Input queue has not overflowed since GKS was opened or the last invocation of INQUIRE INPUT QUEUE OVERFLOW in routine ****

GKS$_ERROR_148:

Error: You called GQIQOV when the queue was not full, and had not been filled since the beginning of your application.

User Action: Allow the user to continue to generate events, if your application still requires input.

149 Input queue has overflowed, but associated workstation has been closed in routine ****

GKS$_ERROR_149:

Error: You called GQIQOV when the queue was full, but since the workstation is closed, information about the overflow is not available.

User Action: You can set the devices to request mode (removing their prompts from the workstation surface), and then you can either process reports from the queue until empty or you can flush the queue of all reports.

150 No input value of the correct class is in the current event report in routine ****

GKS$_ERROR_150:

User Action: Make sure that you check the input class argument passed to GWAIT before you try to call the appropriate GET function.

151 Timeout is invalid in routine ****

GKS$_ERROR_151:

User Action: Make sure that the timer argument in GWAIT is a real value between 0.0 and 356,400, specified in the format described in the GWAIT function description in Chapter 8, Input Functions, in the *DEC GKS Reference Manual.*

152 Initial value is invalid in routine ****

GKS$_ERROR_152:

User Action: Either check to make sure that you specified the correct value, or check the capabilities of the device to see if you requested a value unsupported by the device. If you obtained the value from an inquiry function, check the order and number of arguments specified to the inquiry function.

153 Number of points in the initial stroke is greater than the buffer size in routine ****

GKS$_ERROR_153:

User Action: Either increase the size of the buffer or reduce the number of points in the initial stroke.

154 Length of initial string is greater than the buffer size in routine ****

GKS$_ERROR_154:

User Action: Either increase the size of the buffer or decrease the size of the initial string.

C.9 Metafile Function Errors

This section lists the errors that result when you call a DEC GKS metafile function with invalid or undefined arguments.

160 Item type is not allowed for user items in routine ****

GKS$_ERROR_160:

Error: You used an item type less than 101 to write to a GKSM.

User Action: Use an item type greater than or equal to 101.

161 Item length is invalid in routine ****

GKS$_ERROR_161:

Error: The length of the data item was shorter than necessary for its type.

User Action: Make sure that DEC GKS does not truncate your record when reading the item from a GKSM.

162 No item is left in GKS Metafile input in routine ****

GKS$_ERROR_162:

Error: You tried to read past the end of the GKSM.

User Action: Do not attempt to read items past the item of type 0.

163 Metafile item is invalid in routine ****

GKS$_ERROR_163:

Error: Your item data was incorrect.

User Action: Make sure that DEC GKS did not truncate the item while reading from a GKSM and that you specified correct sizes and types. Make sure that you are not trying to interpret a user-defined record type. User-defined records have item numbers greater than 100.

164 Item type is not a valid GKS item in routine ****

GKS$_ERROR_164:

Error: You tried to interpret an item of type less than 0 or greater than 100.

User Action: Make sure that DEC GKS did not truncate the item while reading from a GKSM and that you specified correct sizes and types.

165 Content of item data record is invalid for the specified item type in routine ****

GKS$_ERROR_165:

Error: There was unexpected or incorrect information in the data record.

User Action: Make sure that you pass the correct storage area.

166 Maximum item data record length is invalid in routine ****

GKS$_ERROR_166:

User Action: Make sure that the data length is not negative.

167 User item cannot be interpreted in routine ****

GKS$_ERROR_167:

User Action: Do not pass user items to DEC GKS for interpretation.

168 Specified function is not supported in this level of GKS in routine ****

GKS$_ERROR_168:

User Action: DEC GKS does not generate this error.

C.10 Escape Function Errors

This section lists the errors that result when you call a DEC GKS escape function with invalid or undefined arguments.

180 Specified escape function is not supported in routine ****

GKS$_ERROR_180:

User Action: Check the escape function identifier to make sure that it is a valid integer representing an escape function, and make sure that you specified the correct workstation identifier.

181 Specified escape function identifier is invalid in routine ****

GKS$_ERROR_181:

User Action: Make sure that the escape function identifier is a valid integer value.

182 Contents of escape data record are invalid in routine ****

GKS$_ERROR_182:

User Action: Make sure that you specified the correct size of the data record. Also, make sure that the elements of the data record are declared to be the correct data type.

C.11 Miscellaneous Errors

This section lists the DEC GKS miscellaneous errors.

200 Specified error file in invalid in routine ****

GKS$_ERROR_200:

User Action: Make sure that your specified error handler exists and that it includes the three required parameters in its definition.

C.12 System Errors

This section lists implementation-dependent errors.

300 Storage overflow has occured in GKS ****

GKS$_ERROR_300:

User Action: Either remove the index, or check the order and the number of the arguments. If you used an inquiry function to obtain the index value, check the order and the number of the arguments passed to the inquiry function.

301 Storage overflow has occured in segment storage ****

GKS$_ERROR_301:

User Action: Either remove the index, or check the order and the number of the arguments. If you used an inquiry function to obtain the index value, check the order and the number of the arguments passed to the inquiry function.

302 Input/Output error has occurred while reading in routine ****

GKS$_ERROR_302:

Error: You specified an illegal metafile for a metafile input workstation.

User Action: Make sure that you work with a valid GKSM metafile, and that you correctly specify the connection identifier.

303 Input/Output error has occurred while writing in routine ****

GKS$_ERROR_303:

Error: You specified an illegal metafile for a metafile output workstation.

User Action: Make sure that you work with a valid GKSM metafile, and that you correctly specify the connection identifier.

304 Input/Output error has occurred while sending data to a workstation ****

GKS$_ERROR_304:

User Action:

305 Input/Output error has occurred while receiving data from a workstation ****

GKS$_ERROR_305:

User Action:

306 Input/Output error has occurred during program library management ****

GKS$_ERROR_307:

User Action:

307 Input/Output error has occurred while reading workstation description table ****

GKS$_ERROR_307:

User Action:

308 Arithmetic error has occurred in routine ****

GKS$_ERROR_308:

Error: You either divided by zero or caused data overflow.

User Action: Check the arguments passed in the function call.

C.13 FORTRAN Binding Errors

This section lists those error messages that are specific to the FORTRAN binding functions.

2000 Enumeration type out of range—the INTEGER passed as a GKS enumerated type is not within the range of valid values in routine ****

GKS$_ERROR_2000:

User Action: Make sure that you properly define the enumerated values.

2001 Output parameter size insufficient—a FORTRAN array or string being passed as an output parameter is too small to contain the returned information in routine ****

GKS$_ERROR_2001:

User Action: Redefine the size of the output string, or check the order or number of arguments passed to the function.

2002 List or set element not available—for a non-empty list or set, a value less than zero or greater than the size of the list or set was passed as the requested list or set element in an inquiry routine in routine ****

GKS$_ERROR_2002:

User Action: Either check the declaration of the value passed, or check the order and number of all arguments passed.

2003 Invalid data record—the data record passed to a GKS routine cannot be decoded, or there was a problem encountered when GKS was creating a data record, making the result invalid in routine ****

GKS$_ERROR_2003:

User Action: Make sure that you used the function GPREC to pack and create the data record. Also, check the order and number of arguments passed to the function. If you used an inquiry function to obtain the data record value, check the order and number of arguments passed to the inquiry function.

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

Reader's Comments

DEC GKS FORTRAN Binding Reference Manual
AA–MJ29A–TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
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)	☐	☐	☐	☐

I would like to see more/less _____

_____

What I like best about this manual is _____

_____

What I like least about this manual is _____

_____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

_____

_____

_____

I am using **Version** _____ of the software this manual describes.

Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

_____ Phone _____

--- Do Not Tear - Fold Here and Tape ---

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications—Spit Brook
ZK01-3/J35
110 SPIT BROOK ROAD
NASHUA, NH 03062-9987

--- Do Not Tear - Fold Here ---

Cut Along Dotted Line

Reader's Comments

DEC GKS FORTRAN Binding
Reference Manual
AA–MJ29A–TE

Please use this postage-paid form to comment on this manual. If you require a written reply to a software problem and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Thank you for your assistance.

I rate this manual's:	Excellent	Good	Fair	Poor
Accuracy (software works as manual says)	☐	☐	☐	☐
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)	☐	☐	☐	☐

I would like to see more/less _____

_____

What I like best about this manual is _____

_____

What I like least about this manual is _____

_____

I found the following errors in this manual:

Page	Description
_____	_____
_____	_____
_____	_____

Additional comments or suggestions to improve this manual:

_____

_____

_____

I am using **Version** _____ of the software this manual describes.

Name/Title _____ Dept. _____

Company _____ Date _____

Mailing Address _____

_____ Phone _____

-- Do Not Tear - Fold Here and Tape ----------

digital™

No Postage
Necessary
if Mailed
in the
United States

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
Corporate User Publications—Spit Brook
ZK01-3/J35
110 SPIT BROOK ROAD
NASHUA, NH 03062-9987

-- Do Not Tear - Fold Here ----------

Cut Along Dotted Line