# The graPHIGS Programming Interface: Subroutine Reference

SC23-6624-00

# The graPHIGS Programming Interface: Subroutine Reference

SC23-6624-00

**Note**
Before using this information and the product it supports, read the information in Appendix D, “Notices,” on page 771.

**First Edition (November 2007)**

This edition applies to the GDDM/graPHIGS Programming Interface, Version 2, Release 2.5, Program Number 5688-093, the AIXwindows Environment/6000 (1.2.5) AIXwindows/3D feature, Program Number 5601-257, and all subsequent releasees of this product until otherwise indicated in new editions.

# Contents

# About This Book

This manual provides information on the subroutines available in the graPHIGS API for creating graphical application programs. It is a reference manual containing the information you need to code your calls and to declare variables correctly, and it is intended to be used in conjunction with the other books in the graPHIGS API library. For example, the concepts and techniques employed in graphics programming are described in *The graPHIGS Programming Interface: Understanding Concepts*; specific information about device support capabilities, such as prompt/echo types for input devices, is provided in *The graPHIGS Programming Interface: Technical Reference*.

Each subroutine listed in this manual has information about error codes and functional relations to help you identify more readily the source of errors resulting from data, program flow. Each subroutine description explains the result to the subroutine call and a list of the errors associated with the subroutine. (See Appendix B. "Error Processing" for general information regarding errors and error processing. See *The graPHIGS Programming Interface: Understanding Concepts* and *The graPHIGS Programming Interface: Messages and Codes* for more detailed information regarding errors and error processing.)

## Who Should Use This Book

This book is intended for application programmers.

## Highlighting

The following highlighting conventions are used in this book:

| | |
|---|---|
| **Bold** | Identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system. Also identifies graphical objects such as buttons, labels, and icons that the user selects. |
| *Italics* | Identifies parameters whose actual names or values are to be supplied by the user. |
| `Monospace` | Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type. |

## ISO 9000

ISO 9000 registered quality systems were used in the development and manufacturing of this product.

## Related Publications

Following are related publications:

- *The graPHIGS Programming Interface: Understanding Concepts*
- *The graPHIGS Programming Interface: Technical Reference*
- *The graPHIGS Programming Interface: ISO PHIGS Subroutine Reference*
- *AIX Version 6.1 AIXwindows Programming Guide*
- *AIX® Version 6.1 Commands Reference* Volume 1: a through c
- *AIX® Version 6.1 Commands Reference* Volume 2: d through h
- *AIX® Version 6.1 Commands Reference* Volume 3: i through m
- *AIX® Version 6.1 Commands Reference* Volume 4: n through r
- *AIX® Version 6.1 Commands Reference* Volume 5: s through u
- *AIX® Version 6.1 Commands Reference* Volume 6: v through z
- *AIX Version 6.1 Technical Reference: Base Operating System and Extensions Volume 1*

- *AIX Version 6.1 Technical Reference: Base Operating System and Extensions Volume 2*

# Chapter 1. Introduction

This manual describes the syntax and operations of the subroutines available in the graPHIGS API interface.

## Calling Conventions for Subroutines

The graPHIGS API can be invoked from a number of programming languages. (To obtain language specific information, see *The graPHIGS Programming Interface: Writing Applications*.

Each invocation consists of the subroutine name with the appropriate parameters required by the subroutine. The graPHIGS API subroutine names are up to six characters long. Mnemonic conventions make the purpose of the subroutine calls evident in the names. Each subroutine name includes the 2-character prefix "GP" and the initial letter or letters from words in the subroutine name. All parameters must be specified in each subroutine and must be in the order shown in the subroutine description.

## Request Control Parameter Codes

The Request Control Parameter (RCP) codes give alternative access to the graPHIGS API subroutines. These codes comprise a system programmer's interface (SPI) which provides a common entry point (AFMASP) for the graPHIGS API. For more information on using the SPI, see *The graPHIGS Programming Interface: Writing Applications*.

## graPHIGS API Subroutines

For each graPHIGS API subroutine, you will find a description of the subroutine, its purpose, the processing performed, a list and description of the parameters, a list of associated error codes and messages, a list of related subroutines, and the associated RCP code (used only in programming for the System Programming Interface).

The graPHIGS API subroutine groups are as follows:

1. Control
2. Output Primitives
3. Attribute Structure Elements
4. Miscellaneous Structure Elements
5. Structure Operations
6. Archive
7. Workstation Table Operations
8. Display
9. Transformation
10. Input
11. Font
12. Image
13. Utility
14. Error Handling
15. Miscellaneous
16. Inquire
17. Compatibility
18. Distributed Application Processing (DAP)
19. Explicit Traversal

## Subroutine Descriptions

The page preceding each group of subroutines presents a brief overview of the purpose and result caused by invoking the subroutines referenced within the given section. This introductory material highlights important information that applies to all the subroutines in that section.

## Reference Manual Abbreviations

The following abbreviations are used frequently:

**ADIB** Application Defaults Interface Block.

**ARCL** Archive Closed.

**AROP** Archive Open.

**ASAP** As Soon As Possible.

**ASF** Attribute Source Flag.

**ASTI** At Some Time.

**BNIG** Before Next Interaction Globally.

**BNIL** Before Next Interaction Locally.

**CMY** Cyan-Magenta-Yellow color model.

**CSID** Character Set Identifier.

**EDF** External Defaults File.

**HSV** Hue-Saturation-Value color model.

**NDT** Nucleus Descriptor Table.

**NROP** Non-Retained Structure Open.

**NSL** Nucleus State List.

**PDT** graPHIGS API Description Table.

**PHCL** graPHIGS Closed.

**PHOP** graPHIGS Open.

**PSL** graPHIGS API State List.

**RGB** Red-Green-Blue color model.

**SSCL** Structure Store Close.

**SSL** Structure Store State List.

**SSOP** Structure Store Open.

**STCL** Structure Closed.

**STOP** Structure Open.

**UQUM**
Quick Update Method.

**USL** Utility State List.

**WAIT** When Application Requests It.

**WDO** Workstation-Dependent Output.

**WDT** Workstation Description Table.

**WSCL** Workstation Closed.

**WSID** Workstation Identifier.

**WSL** Workstation State List.

**WSOP**
Workstation Open.

**WSSL** Workstation Selected.

**WSTYPE**
Workstation Type.

The following abbreviations for coordinate spaces are used:

**DC** Device Coordinates

**MC** Modeling Coordinates

**NPC** Normalized Projection Coordinates

**VC** Viewing Coordinates

**WC** World Coordinates

**WU** Window Units

**Note:** For more information, see *The graPHIGS Programming Interface: Understanding Concepts*.

# Chapter 2. Control Subroutines

The control subroutines allow your application to have access to and control over the graphical resources available when using the graPHIGS API (*).

You can open and close the graPHIGS API and invoke the diagnostic trace for debugging.

When an application opens the graPHIGS API, a *nucleus*, which is the collection of resources available to your application, is connected to your application. When you connect the nucleus, you can create resources, such as workstations, structure stores, and font directories for use by your application. You can connect several application processes to the nucleus. Using the attach subroutines, several application processes share resources. Using message subroutines, application processes communicate with each other. In addition, your application can control the timing of when the graPHIGS API sends buffered transactions to the nucleus for processing.

Also invoke these subroutines to affect the timing of update operations and to explicitly control the update and redraw operations on a workstation.

These subroutines do not store or modify graphics data.

## GPATR - Attach Resource

**GPATR (*type*, *id*, *ncid*, *rid*, *pass*)**

### Purpose

Use **GPATR** to allow your application the use of resources created by other application processes that are connected to the specified nucleus. Use this subroutine when several application processes in different nodes (such as a host application and a Distributed Application Process [DAP]) share access and control of the same resource (such as a workstation or structure store).

The *id* parameter specifies an identifier to be assigned by your application to the attached resource. For example, if the *type* parameter specifies `1=WORKSTATION`, then the application uses the specified *id* as the workstation identifier (*wsid*) of the attached workstation. Similarly, for the other resource types, the *id* parameter is used as the structure store identifier (*ssid*), image board identifier (*ibid*), font directory identifier (*fdid*), or archive file identifier (*arid*) respectively.

The *rid* parameter is obtained by the application process that created the resource by issuing the Inquire Nucleus Resource Identifier (**GPQNCR**) subroutine. It is the resource identifier assigned by the nucleus to the resource when it was created. The application process must then provide the assigned identifier (*rid*) and any required password (*pass*) to your application process for use by this subroutine.

If the specified resource type is `1=WORKSTATION`, then the current workstation state is set to Workstation Open (WSOP).

If the specified resource type is `2=STRUCTURE_STORE`, then the current structure state is set to Structure Store Open (SSOP). After attaching to the specified structure store, your application must issue the Select Structure Store (**GPSSS**) subroutine before any editing of the structure store is allowed.

If the specified resource type is `5=ARCHIVE_FILE`, then the current archive state is set to Archive Open (AROP).

### Parameters

*type* **— specified by user, fullword integer**
Resource type (`1=WORKSTATION, 2=STRUCTURE_STORE, 3=IMAGE_BOARD, 4=FONT_DIRECTORY, 5=ARCHIVE_FILE`).

*id* **— specified by user, fullword integer**
Identifier to be assigned to the resource.

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*rid* **— specified by user, fullword integer**
Nucleus resource identifier.

*pass* **— specified by user, fullword integer**
Resource password.

### Error Codes

| | |
|---|---|
| **24** | SPECIFIED WORKSTATION IDENTIFIER ALREADY IS IN USE |
| **202** | SPECIFIED NUCLEUS DOES NOT EXIST |
| **211** | RESOURCE TYPE IS INVALID |
| **212** | SPECIFIED RESOURCE IDENTIFIER DOES NOT EXIST |
| **213** | SPECIFIED PASSWORD IS INCORRECT |
| **219** | SPECIFIED ARCHIVE FILE IDENTIFIER ALREADY IN USE |
| **221** | SPECIFIED STRUCTURE STORE IDENTIFIER ALREADY IS IN USE |
| **231** | SPECIFIED IMAGE BOARD IDENTIFIER ALREADY IS IN USE |
| **241** | SPECIFIED FONT DIRECTORY IDENTIFIER ALREADY IS IN USE |

### Related Subroutines

| | |
|---|---|
| **GPCRFD** | Create Font Directory |
| **GPCRIB** | Create Image Board |
| **GPCRSS** | Create Structure Store |
| **GPCRWS** | Create Workstation |
| **GPDTR** | Detach Resource |
| **GPOPAR** | Open Archive File |
| **GPQATR** | Inquire List of Attached Resources |
| **GPQNCR** | Inquire Nucleus Resource Identifier |
| **GPSSS** | Select Structure Store |

### RCP code

201341697 (X'0C003B01')

## GPCLAR - Close Archive File

**GPCLAR (*arid*)**

### Purpose

Use **GPCLAR** to close the specified open graPHIGS API archive file.

If no other archive files are open, then **GPCLAR** sets the current archive state to Archive Closed (ARCL).

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

### Error Codes

| | |
|---|---|
| **7** | FUNCTION REQUIRES STATE AROP |
| **220** | SPECIFIED ARCHIVE FILE DOES NOT EXIST |

### Related Subroutines

| | |
|---|---|
| **GPOPAR** | Open Archive File |

### RCP code

201348356 (X'0C005504')

## GPCLPH - Close graPHIGS

**GPCLPH**

### Purpose

Use **GPCLPH** to terminate all graPHIGS API processing for this application process. This subroutine function detaches all attached resources or resources created by the application and disconnects all nuclei connected to the application. **GPCLPH** closes all graPHIGS API files and releases system resources, such as storage and locks. This subroutine also sets the graPHIGS system state to graPHIGS Closed (PHCL). Reopen the graPHIGS API by invoking the Open graPHIGS (**GPOPPH**) subroutine.

### Error Codes

None

### Related Subroutines

| | |
|---|---|
| **GPOPPH** | Open graPHIGS |

### RCP code

201327105 (X'0C000201')

## GPCLWS - Close Workstation

**GPCLWS (*wsid*)**

### Purpose

Use **GPCLWS** to close the specified workstation. The workstation updates automatically before closing.

This subroutine function releases the workstation state list and deletes the workstation's identifier from the set of opened workstations in the graPHIGS API state list. Additionally, **GPCLWS** flushes the input queue of all events from all input devices on the workstation, and releases the connection to the workstation. If no workstation remains open, the workstation state is set to Workstation Closed (WSCL).

**GPCLWS** clears the workstation. For workstations that keep a local copy of the structure store, the graPHIGS API frees the structure storage at this time.

This subroutine is functionally equivalent to the Detach Resource (**GPDTR**) subroutine with a resource type set to `1=WORKSTATION`.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **147** | EVENT QUEUE HAS OVERFLOWED |

### Related Subroutines

| | |
|---|---|
| **GPCRWS** | Create Workstation |
| **GPDTR** | Detach Resource |
| **GPOPWS** | Open Workstation |
| **GPQOPW** | Inquire Set of Open Workstations |

### RCP code

201327617 (X'0C000401')

## GPCNC - Connect Nucleus

**GPCNC (*ncid*, *conn*, *len*, *spec*)**

### Purpose

Use **GPCNC** to establish access to a nucleus from your application. The nucleus allows your application to access resources, such as a workstation and a structure store, for graphics processing.

This subroutine must be invoked in order to communicate with a nucleus. Your application may choose to invoke this subroutine or have the graPHIGS API invoke this subroutine for your application as part of *nucleus connection processing*.

When the graPHIGS API performs *nucleus connection processing*, it connects to a nucleus (**GPCNC**) with an identifier set to a value of one, creates a structure store (**GPCRSS**) with an identifier set to a value of one, and selects that structure store (**GPSSS**) for editing. This may be done as follows:

- If your application is using any workstation *except* the 6090 workstation and you want the graPHIGS API to invoke this subroutine for your application as part of *nucleus connection processing*, then you do *not* need to do anything additional. When your application executes an Open graPHIGS (**GPOPPH**) subroutine, the application connects to a nucleus with an identifier set to a value of one, using the `1=CALL` connection method.

- If your application is using the 6090 workstation and you want the graPHIGS API to invoke this subroutine for your application as part of *nucleus connection processing*, then you *must* use the Define Nucleus Connection Processing (DEFNUC) procopt default in either the Application Defaults Interface Block (ADIB) or the External Default File (EDF) to indicate this.
- If you want your application to invoke this subroutine explicitly, then you must use the Define Nucleus Connection Processing (DEFNUC) procopt default in either the Application Defaults Interface Block (ADIB) or the External Defaults File (EDF) to suppress the graPHIGS API from invoking the connect to nucleus as part of *nucleus connection processing*. (If your application is a distributed application process [DAP], then it is *not* necessary to suppress *nucleus connection processing*.

For information about nucleus connection, see *The graPHIGS Programming Interface: Understanding Concepts*.

For the available nucleus connection methods, connection specifications, and information concerning Application Defaults Interface Block (ADIB) and External Defaults File (EDF), see *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*conn* **— specified by user, fullword integer**
Connection method (1=CALL, 2=GAM, 3=SOCKETS) GAM stands for *Graphics Access Method* and is the connection method used with the IBM 6090 Graphics System Access.

The connection methods that are supported for each combination of nucleus and application environments are summarized in the following table:

**Figure 1. Nucleus/Application Environment**

| Nucleus Environment | Application Environment: VM | MVS | AIX® | 6090 |
|---|---|---|---|---|
| VM | CALL | N/A | N/A | N/A |
| MVS | N/A | CALL | N/A | N/A |
| AIX® | N/A | N/A | CALL or SOCKETS | N/A |
| 6090 | GAM | GAM | N/A | SOCKETS |

**Note:**
**N/A** Not a valid combination

*len* **— specified by user, fullword integer**
Length of the nucleus connection specification.

*spec* **— specified by user, variable length character string**
Connection specification. The required nucleus specification string for each connection method and application environment is summarized in the following table:

**Figure 2. Application Environment/Connection Method**

| Connection Method: 1=CALL | 2=GAM | 3=SOCKETS |
|---|---|---|

| Application Environment | | | | |
|---|---|---|---|---|
| | VM | Null (length = 0) | Filedef (length =>8) | N/A |
| | MVS | Null (length = 0) | DDNAME (length =>8) | N/A |
| | AIX® | Null (length = 0) | N/A | Nucspec* (length >= 2) |
| | 6090 | N/A | N/A | :0 (length = 2) |

**Note:**

**N/A** Not a valid combination

***** Nucspec is set to the following specification:

*hostname: nucleus connection number* where:

- *hostname* specifies the host name of the system where the remote graPHIGS API nucleus is running.
- *nucleus connection number* specifies the identifier of the nucleus on the named system and is a number in the range 0-255.

### Error Codes

| | |
|---|---|
| **201** | SPECIFIED NUCLEUS IDENTIFIER ALREADY IS IN USE |
| **203** | SPECIFIED CONNECTION METHOD IS NOT SUPPORTED |
| **204** | NUCLEUS CONNECTION FAILED |
| **208** | CONNECTION NOT CURRENTLY PERMITTED FROM THIS HOST |

### Related Subroutines

| | |
|---|---|
| **GPDNC** | Disconnect Nucleus |
| **GPQCMM** | Inquire List of Available Connection Methods |
| **GPQCNC** | Inquire List of Connected Nuclei |
| **GPQNS** | Inquire Nucleus Specification |
| **GPQSH** | Inquire Shell Identifier |

### RCP code

201337601 (X'0C002B01')

## GPCRFD - Create Font Directory

**GPCRFD (*fdid*, *ncid*, *fdtype*, *fddesc*)**

### Purpose

Use **GPCRFD** to create a font directory and attach it to your application. A font directory allows your application to send geometric text font files to a nucleus. The graPHIGS API uses these files to display

geometric text on a workstation. (To accomplish this, your application must also associate the font directory to the workstation and activate the font on the workstation.)

This subroutine is not required unless your application needs to replace the font files of the nucleus. By default, a workstation has access to the font files that reside on the nucleus. When overriding a nucleus font file, your application creates a font directory to store the font file that is to be loaded from your application's set of fonts. (For more information about fonts, see *The graPHIGS Programming Interface: Understanding Concepts*.)

### Parameters

*fdid* **— specified by user, fullword integer**
Font directory identifier.

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*fdtype* **— specified by user, fullword integer**
Font directory type (`1=NORMAL`).

*fddesc* **— specified by user, variable data**
Font directory descriptor. This parameter includes font directory type dependent data. For the font directory type set to `1=NORMAL`, no descriptive data is required; therefore, this parameter must be specified with a value of zero.

### Error Codes

| | |
|---|---|
| **202** | SPECIFIED NUCLEUS DOES NOT EXIST |
| **217** | RESOURCE CREATION REQUEST EXCEEDS NUCLEUS TABLE CAPACITY |
| **241** | SPECIFIED FONT DIRECTORY IDENTIFIER ALREADY IS IN USE |
| **243** | SPECIFIED FONT DIRECTORY TYPE IS NOT SUPPORTED |

### Related Subroutines

| | |
|---|---|
| **GPACFO** | Activate Font |
| **GPAFDW** | Associate Font Directory with Workstation |
| **GPDTR** | Detach Resource |
| **GPLDFO** | Load Font |
| **GPQNCR** | Inquire Nucleus Resource Identifier |

### RCP code

201341185 (X'0C003901')

## GPCRIB - Create Image Board

**GPCRIB (*ibid*, *ncid*, *depth*, *h*, *v*, *ibtype*, *ibdesc*)**

### Purpose

Use **GPCRIB** to create an image board and attach it to your application. An image board is a two-dimensional array of data values that your application can manipulate to create an image that your workstation can display. For information about image support, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*ibid* **— specified by user, fullword integer**
Image board identifier.

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*depth* **— specified by user, fullword integer**
Bit depth (`1, 2, 4, 8, 12`).

*h* **— specified by user, fullword integer**
Horizontal size (>=1).

*v* **— specified by user, fullword integer**
Vertical size (>=1).

*ibtype* **— specified by user, fullword integer**
Image board type (`1=NORMAL`).

*ibdesc* **— specified by user, variable data**
Image board descriptor. This parameter includes image board type dependent data. For the image board type set to `1=NORMAL`, no descriptive data is required; therefore, this parameter must be set to a value of zero.

### Error Codes

| | |
|---|---|
| **202** | SPECIFIED NUCLEUS DOES NOT EXIST |
| **217** | RESOURCE CREATION REQUEST EXCEEDS NUCLEUS TABLE CAPACITY |
| **231** | SPECIFIED IMAGE BOARD IDENTIFIER ALREADY IS IN USE |
| **233** | SPECIFIED IMAGE BOARD BIT DEPTH IS NOT SUPPORTED |
| **234** | SPECIFIED IMAGE BOARD SIZE IS INVALID |
| **235** | SPECIFIED IMAGE BOARD TYPE IS NOT SUPPORTED |
| **1109** | FUNCTION NOT SUPPORTED |

### Related Subroutines

| | |
|---|---|
| **GPCAI** | Cancel Image |
| **GPDFI** | Define Image |
| **GPDTR** | Detach Resource |
| **GPQIBC** | Inquire Image Board Characteristics |
| **GPQIBF** | Inquire Image Board Facilities |
| **GPQICH** | Inquire Image Characteristics |
| **GPQNCR** | Inquire Nucleus Resource Identifier |

### RCP code

201342977 (X'0C004001')

# GPCRSS - Create Structure Store

**GPCRSS (*ssid*, *ncid*, *sstype*, *ssdesc*)**

### Purpose

Use **GPCRSS** to create a structure store and attach it to your application.

If the current structure state is Structure Store Close (SSCL), it is set to Structure Store Open (SSOP).

A structure store is required if your application is to display graphical data on a workstation using output primitives and their attributes.

Before any editing of a structure is allowed, the application must select a structure store by issuing the Select Structure Store (**GPSSS**) subroutine. Before displaying any structure data on a workstation, the application must associate the structure store to the workstation by issuing the Associate Structure Store with Workstation (**GPASSW**) subroutine. For compatibility with previous releases, this subroutine, along with the Select Structure Store (**GPSSS**) subroutine, is called automatically during Open graPHIGS (**GPOPPH**) processing unless otherwise suppressed. If you suppress the default nucleus connection processing by using the Define Nucleus Connection Processing (DEFNUC) procopt default in either the Application Defaults Interface Block (ADIB) or the External Defaults File (EDF), then you must invoke this subroutine prior to opening a structure.

For information about structure stores, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*ssid* **— specified by user, fullword integer**
Structure store identifier.

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*sstype* **— specified by user, fullword integer**
Structure store type (`1=NORMAL`).

*ssdesc* **— specified by user, variable data**
Structure store descriptor. This parameter includes structure store type dependent data. For the structure store type `1=NORMAL`, no descriptive data is required and so this parameter must be specified as a zero.

### Error Codes

| | |
|---|---|
| **202** | SPECIFIED NUCLEUS DOES NOT EXIST |
| **217** | RESOURCE CREATION REQUEST EXCEEDS NUCLEUS TABLE CAPACITY |
| **221** | SPECIFIED STRUCTURE STORE IDENTIFIER ALREADY IS IN USE |
| **223** | SPECIFIED STRUCTURE STORE TYPE IS NOT SUPPORTED |

### Related Subroutines

| | |
|---|---|
| **GPASSW** | Associate Structure Store with Workstation |
| **GPDTR** | Detach Resource |

**GPOPST** Open Structure
**GPQSTV** Inquire Structure State Value
**GPSSS** Select Structure Store

**RCP code**

201340929 (X'0C003801')

## GPCRWS - Create Workstation

**GPCRWS (*wsid*, *ncid*, *length*, *connid*, *wstype*, *option*)**

**Purpose**

Use **GPCRWS** to open the specified workstation on the specified nucleus and optionally to provide default values to be used by the workstation during its initialization. The current workstation state is set to Workstation Open (WSOP).

The *option* parameter enables your application to modify the workstation's defaults by specifying a group of procopts. These specified procopts override the procopts available in the Application Defaults Interface Block (ADIB) or the External Defaults File (EDF). (For information about valid procopts and their values, see *The graPHIGS Programming Interface: Technical Reference*.

When no procopts are specified, a fullword integer with a value of zero must be specified for the parameter.

This subroutine does *not* automatically associate the currently selected structure store with a workstation. In order for association to occur, the application must explicitly issue the Associate Structure Store with Workstation (**GPASSW**) subroutine.

This subroutine is functionally equivalent to the Open Workstation (**GPOPWS**) subroutine with nucleus identifier of one with the exception that **GPCRWS** will *not* perform the implicit Associate Structure Store with Workstation (**GPASSW**) processing.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

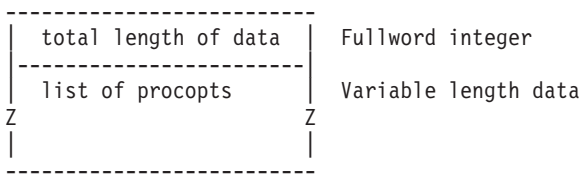
*length* **— specified by user, fullword integer**
Length of the connection identifier (>0).

*connid* **— specified by user, variable length character string**
Connection identifier indicates the physical device to be opened. (For information about the connection identifier, see *The graPHIGS Programming Interface: Technical Reference*.

*wstype* **— specified by user, 8-byte character string**
One of the graPHIGS API supported workstation types. (For information about the supported workstation types, see *The graPHIGS Programming Interface: Technical Reference*.

*option* **— specified by user, variable data**
Workstation creation option.

The *option* parameter has the following format:




For information about valid procopts and their values, see *The graPHIGS Programming Interface: Technical Reference*. When no option is specified, a fullword integer with a value zero must be specified for the parameter.

### Error Codes

| | |
|---|---|
| **21** | CONNECTION IDENTIFIER IS INVALID |
| **23** | SPECIFIED WORKSTATION TYPE DOES NOT EXIST |
| **24** | SPECIFIED WORKSTATION IDENTIFIER ALREADY IS IN USE |
| **26** | SPECIFIED WORKSTATION CANNOT BE OPENED |
| **61** | LENGTH IS INVALID |
| **202** | SPECIFIED NUCLEUS DOES NOT EXIST |
| **217** | RESOURCE CREATION REQUEST EXCEEDS NUCLEUS TABLE CAPACITY |
| **514** | INAPPROPRIATE DEVICE FOR WORKSTATION TYPE |
| **581** | PROCOPT SPECIFIES INVALID VIEW TABLE SIZE FOR WORKSTATION |
| **583** | PROCOPT SPECIFIES INVALID NUMBER OF INPUT DEVICES FOR WORKSTATION |
| **585** | PROCOPT SPECIFIES INVALID KEYBOARD FOR WORKSTATION |
| **586** | PROCOPT SPECIFIES INVALID DISPLAY MODEL NUMBER FOR WORKSTATION |
| **587** | PROCOPT SPECIFIES INVALID ECHO METHOD FOR WORKSTATION |
| **588** | PROCOPT SPECIFIES INVALID FRAME BUFFER VALUE FOR WORKSTATION |
| **596** | PROCOPT SPECIFIES INVALID NUMBER OF POLYLINE TABLE ENTRIES |
| **597** | PROCOPT SPECIFIES INVALID NUMBER OF POLYMARKER TABLE ENTRIES |
| **598** | PROCOPT SPECIFIES INVALID NUMBER OF TEXT TABLE ENTRIES |
| **599** | PROCOPT SPECIFIES INVALID NUMBER OF EDGE TABLE ENTRIES |
| **600** | PROCOPT SPECIFIES INVALID NUMBER OF DEPTH CUE TABLE ENTRIES |
| **601** | PROCOPT SPECIFIES INVALID NUMBER OF LIGHT SOURCE TABLE ENTRIES |
| **602** | PROCOPT SPECIFIES INVALID NUMBER OF INTERIOR TABLE ENTRIES |
| **648** | PROCOPT SPECIFIES AN INVALID DISPLAY WIDTH AND/OR HEIGHT |
| **649** | PROCOPT SPECIFIES AN INVALID IMAGE OUTPUT FORMAT |
| **650** | PROCOPT SPECIFIES AN INVALID HLHSR COORDINATE SYSTEM |

### Related Subroutines

**GPASSW** Associate Structure Store with Workstation
**GPATR** Attach Resource
**GPCLWS** Close Workstation
**GPDTR** Detach Resource
**GPOPWS** Open Workstation
**GPQNCR** Inquire Nucleus Resource Identifier
**GPQWSV** Inquire Workstation State Value
**GPQWTN** Inquire List of Available Workstation Types on Nucleus
**GPQWTO** Inquire Workstation Type and Options

### RCP code

201327362 (X'0C000302')

## GPDF - Set Deferral State

**GPDF (*wsid*, *defer*, *modif*)**

### Purpose

Use**GPDF** to set the deferral state and modification mode for the specified workstation.

These settings determine when pending updates are processed for display on a workstation and how the workstation is to perform the modifications. For information about the abbreviations and modes, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*defer* **— specified by user, fullword integer**
Deferral state (`1=ASAP, 2=BNIG, 3=BNIL, 4=ASTI, 5=WAIT`).

*modif* **— specified by user, fullword integer**
Modification mode (`1=NO_IMMEDIATE_VISUAL_EFFECT, 2=UPDATE_WITHOUT_REGENERATION, 3=QUICK_UPDATE`).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**303** DEFERRAL MODE IS INVALID

**304** MODIFICATION MODE IS INVALID

### Related Subroutines

**GPQDV**
Inquire Deferral and Update State Values

**GPQDDV**
Inquire Default Deferral State Values

**GPSHDF**
Set Shell Deferral State

### RCP code

201327875 (X'0C000503')

## GPDNC - Disconnect Nucleus

**GPDNC (*ncid*)**

### Purpose

Use **GPDNC** to disconnect a graPHIGS nucleus from your application. All of the resources of the nucleus become unavailable to your application.

The following actions are performed:

1. If the structure state is Structure Open (STOP) and the currently selected structure store is a resource of the specified nucleus, the structure is closed. As a result, the structure state becomes Structure Close (STCL).
2. If the structure state is Structure Close (STCL) and the currently selected structure store is a resource of the specified nucleus, the structure store identifier is removed from the currently selected structure store entry of the graPHIGS API state list. As a result, the structure state becomes Structure Store Open (SSOP).
3. All resources of the specified nucleus are detached from the shell. If there remains no structure store or no workstation attached to the shell, the structure state or workstation state becomes Structure Store Close (SSCL) or Workstation Close (WSCL), respectively.

### Parameters

*ncid* — **specified by user, fullword integer**
Nucleus identifier.

### Error Codes

| | |
|---|---|
| **202** | SPECIFIED NUCLEUS DOES NOT EXIST |

### Related Subroutines

| | |
|---|---|
| **GPCNC** | Connect Nucleus |
| **GPQCNC** | Inquire List of Connected Nuclei |

### RCP code

201337602(X'0C002B02')

## GPDTR - Detach Resource

**GPDTR (*type, id*)**

### Purpose

Use **GPDTR** to detach a resource from your application. The resource becomes unavailable to your application.

The *id* parameter specifies an identifier that your application assigned to the resource. The *type* parameter is used to identify the type of resource that is specified by the *id* parameter. For example, if the *type* parameter specifies 1=WORKSTATION, then the *id* parameter is used to specify a workstation identifier (*wsid*) to be detached.

**If the specified resource type is 1=WORKSTATION**, all events from the specified workstation are removed from the event queue. If the specified workstation is the last open workstation, the workstation state value is set to Workstation Closed (WSCL)

**If the specified resource type is 2=STRUCTURE_STORE**, all events from the specified structure store are removed from the event queue. If the current structure state is Structure Open (STOP) and the specified structure store is currently selected, then the current open structure is closed. As a result, the structure state is set to Structure Closed (STCL). If the current structure state is Structure Closed (STCL) and the specified structure store is the currently selected structure store, it is removed from the currently selected structure store entry of the graPHIGS API state list. As a result, the structure state is set to Structure Store Open (SSOP). If the specified structure store is the last structure store attached to the shell, then the structure state is set to Structure Store Closed (SSCL).

**If the specified resource type is 5=ARCHIVE_FILE** and if the specified archive file is the last open archive file, then the archive state value is set to Archive Closed (ARCL).

### Parameters

*type* **— specified by user, fullword integer**
Resource type (1=WORKSTATION, 2=STRUCTURE_STORE, 3=IMAGE_BOARD, 4=FONT_DIRECTORY, 5=ARCHIVE_FILE).

*id* **— specified by user, fullword integer**
Identifier of the resource to be detached.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **147** | EVENT QUEUE HAS OVERFLOWED |
| **211** | RESOURCE TYPE IS INVALID |
| **220** | SPECIFIED ARCHIVE FILE DOES NOT EXIST |
| **222** | SPECIFIED STRUCTURE STORE DOES NOT EXIST |
| **232** | SPECIFIED IMAGE BOARD DOES NOT EXIST |
| **242** | SPECIFIED FONT DIRECTORY DOES NOT EXIST |

### Related Subroutines

| | |
|---|---|
| **GPATR** | Attach Resource |
| **GPCLAR** | Close Archive File |
| **GPQATR** | Inquire List of Attached Resources |

### RCP code

201341698 (X'0C003B02')

## GPMSG - Message

**GPMSG (*wsid*, *length*, *text*)**

### Purpose

Use **GPMSG** to display a message on the specified workstation.

The message text appears in the lower left corner of the workstation viewport and is clipped to this viewport.

The message is unaffected by deferral state settings and remains displayed until removed by another message. It can be cleared by calling **GPMSG** with a length of zero.

The appearance (size and color) of the message text is workstation dependent. The workstation's primary character set is used to convert the text if necessary.

For workstation-specific information about message text, see information about character sets in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*length* **— specified by user, fullword integer**
Length of text to be displayed (>=0).

*text* **— specified by user, variable length character string**
Message to be displayed.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**197** MESSAGE STRING LENGTH < ZERO

### Related Subroutines

None

### RCP code

201327876 (X'0C000504')

## GPMSPW - Set Message Password

**GPMSPW (*ncid*, *pass*)**

### Purpose

Use **GPMSPW** to set a password for application message receiving. The graPHIGS API uses this password to authorize the Send Private Message (**GPSPMS**) subroutine or Send Broadcast Message (**GPSBMS**) subroutine issued by other application processes.

For more information about the use of broadcast and private messages, see *The graPHIGS Programming Interface: Understanding Concepts* and *The graPHIGS Programming Interface: Writing Applications*.

### Parameters

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*pass* — **specified by user, fullword integer**
Password.

**0=None**
Application will not receive messages.

**-1=All** Application will receive any broadcast or private messages.

**Unique Integer**
Application will receive only private messages.

### Error Codes

**202** SPECIFIED NUCLEUS DOES NOT EXIST

### Related Subroutines

**GPSBMS**
Send Broadcast Message

**GPSPMS**
Send Private Message

### RCP code

201341443 (X'0C003A03')

## GPOPAR - Open Archive File

**GPOPAR (*arid*, *ncid*, *flag*, *length*, *ardesc*)**

### Purpose

Use **GPOPAR** to open and initialize a graPHIGS API archive file.

This subroutine function sets the current archive state to Archive Open (AROP).

If the specified archive file exists as read only and the application attempts to open the file as `1=OPEN_READ/WRITE`, then the graPHIGS API opens the file and issues the warning message 1113. If it cannot open the file, then the graPHIGS API issues an error.

If the specified archive file does not exist and the archive flag is set to a value of `1=OPEN_READ/WRITE`, then the archive file resource creates a new read/write file. If the specified archive file does not exist and the archive flag is set to a value of `2=OPEN_READ_ONLY`, then the graPHIGS API issues an error.

The graPHIGS API External Defaults File (EDF) allows the application to denote, indirectly, the actual value of the file descriptor. For more information on the contents and formats of the EDF, see *The graPHIGS Programming Interface: Technical Reference* under "Defaults and Nicknames."

### Parameters

*arid* — **specified by user, fullword integer**
Archive file identifier.

*ncid* — **specified by user, fullword integer**
Nucleus identifier.

*flag* **— specified by user, fullword integer**
Archive flag (`1=OPEN_READ/WRITE, 2=OPEN_READ_ONLY.`)

*length* **— specified by user, fullword integer**
Length of the file descriptor (>=1).

*ardesc* **— specified by user, variable length character string**
Archive file descriptor. This parameter looks like a Unix file descriptor which consists of a **[path]/filename[extension]**. **Path** is the route of directories through the file system. **Path** is optional and ignored for MVS and VM. An example of a full file descriptor:

```
/phigs/file1.archive
```

where:

- **path** = **/phigs** which says go from the root directory to the directory **phigs**
- **filename** = **file1**
- **extension** = **.archive**

The following rules apply to the descriptor, depending on which system the nucleus is running in:

- **AIX®**

  If you do not specify the path, then the graPHIGS API uses the default directory at the time of the execution of the subroutine.
- **MVS**
  - `filename` - You must specify a filename. This is the DD-name of the BSAM data set of the archive file.
  - `extension` - Any extension is ignored.
- **VM/CMS**

  The file descriptor can have one of two forms:
  - `filename [filetype [filemode]]` or
  - `filename[.filetype[.filemode]]`

  If the filetype is missing, then the graPHIGS API uses a filetype of `ARCHIVE`.

  If the filemode is missing, then the graPHIGS API uses a filemode of `A1`

**Error Codes**

**61** LENGTH IS INVALID

**142** VALUE OF ARCHIVE FLAG IS INVALID

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**217** RESOURCE CREATION REQUEST EXCEEDS NUCLEUS TABLE CAPACITY

**218** ARCHIVE FILES ARE NOT SUPPORTED ON SPECIFIED NUCLEUS

**219** SPECIFIED ARCHIVE FILE IDENTIFIER ALREADY IN USE

**1105** INVALID FILE NAME, *a2*

**1107** FILE *a2* NOT FOUND

**1110** CONCURRENT USAGE OF FILE *a2* NOT ALLOWED

**1113** FILE IS READ ONLY

**1114** FILE CANNOT BE CREATED. DISK IS READ ONLY

**1117** INCORRECT RECORD LENGTH OR FORMAT ON *a2*

**1205** FILE IS NOT A VALID graPHIGS ARCHIVE FILE

**1206** VERSION OF graPHIGS ARCHIVE FILE NOT RECOGNIZED

**Related Subroutines**

**GPCLAR**
Close Archive File

**RCP code**

201348353 (X'0C005501')

## GPOPPH - Open graPHIGS

**GPOPPH (*errfil*, *adib*)**

**Purpose**

Use **GPOPPH** to open and initialize the graPHIGS API. This subroutine makes all the graPHIGS API subroutines available. Call **GPOPPH** before invoking any other graPHIGS API subroutine.

This subroutine creates one graPHIGS API shell and initializes the graPHIGS API state list (PSL) **GPOPPH** sets the system state value to graPHIGS Open (PHOP), sets the workstation state value to Workstation Closed (WSCL), and sets the structure state value to Structure Closed (STCL).

The Application Defaults Interface Block (ADIB) allows the application to modify the graPHIGS API system and workstation defaults. For the contents and formats of the Application Defaults Interface Block (ADIB) and the External Defaults File (EDF), see *The graPHIGS Programming Interface: Technical Reference* under "Defaults and Nicknames."

In this version of the graPHIGS API, some additional subroutines must be issued in order to communicate with a nucleus and to edit structures. These subroutines consist of:

- Connect to Nucleus (**GPCNC**)
- Create Structure Store (**GPCRSS**)
- Select Structure Store (**GPSSS**)

Your application may chose to issue these additional subroutines or have the graPHIGS API issue these subroutines for your application (which is called *nucleus connection processing*). Nucleus connection processing may be done as follows:

- If your application is using any workstation except the 6090 workstation and you want the graPHIGS API to invoke nucleus connection processing for your application, then you do *not* need to do anything additional. When you execute your application, it connects to a nucleus with the identifier set to a value of one, and the graPHIGS API creates and selects a structure store with the identifier set to a value of one for your application.
- If your application is going to use the 6090 workstation and you want the graPHIGS API to invoke nucleus connection processing, then you *must* use the Default Nucleus Connection Processing (DEFNUC) procopt default in either the Application Defaults Interface Block (ADIB) or the External Defaults File (EDF) to indicate this.
- If your application is going to issue these additional subroutines, then you *must* use the Default Nucleus Connection (DEFNUC) procopt in the Application Defaults Interface Block (ADIB) or the External Defaults File (EDF) to suppress the graPHIGS API from invoking nucleus connection processing. (It is *not* necessary to suppress nucleus connection processing if your application is a distributed application process [DAP]).

To determine if the **GPOPPH** subroutine call was successful, your application can use the Inquire System State Value (**GPQSYV**) subroutine.

For formats and contents of the Application Defaults Interface Block (ADIB) and External Defaults File (EDF), see *The graPHIGS Programming Interface: Technical Reference* under "Defaults and Nicknames."

**Parameters**

*errfil* — **specified by user, 8-byte character string**
Name of the error file.

*adib* — **specified by user, variable data**
Application Defaults Interface Block (ADIB). When no defaults are specified, an ADIB that consists of a fullword integer with a value of zero must be specified.

**Error Codes**

**1** FUNCTION REQUIRES STATE PHCL

**208** CONNECTION NOT CURRENTLY PERMITTED FROM THIS HOST

**Related Subroutines**

**GPCLPH**
Close graPHIGS

**GPQSYV**
Inquire System State Value

**RCP code**

201326849 (X'0C000101')

## GPOPWS - Open Workstation

**GPOPWS (*wsid*, *connid*, *wstype*)**

**Purpose**

Use **GPOPWS** to open the specified workstation on a nucleus with the identifier set to a value of one.

If a structure store is currently selected (structure state is Structure Close (STCL) or Structure Open (STOP)), the graPHIGS API calls the Associated Structure Store with Workstation (**GPASSW**) subroutine to associate the currently selected structure store with the specified workstation. This implicit association, along with the implicit processing during Open graPHIGS (**GPOPPH**), allows the graPHIGS API to create the correct processing environment for applications unchanged from Version 1.

**GPOPWS** uses a nucleus identifier set to a value of one to create the specified workstation. If the nucleus identifier of value one does not exist, then the graPHIGS API issues error 202. If you wish to use a nucleus identifier other than value one, then you must use the Create Workstation (**GPCRWS**) subroutine.

If a connection identifier longer than eight characters is required, then you must either issue the Create Workstation (**GPCRWS**) subroutine or you must define a nickname in the Application Defaults Interface Block (ADIB) or the External Defaults File (EDF).

This subroutine is functionally equivalent to the Create Workstation (**GPCRWS**) subroutine with a nucleus identifier set to a value of one,with the exception that **GPOPWS** performs the implicit Associate Structure Store with Workstation (**GPASSW**) processing.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*connid* — **specified by user, 8-byte character string**
Connection identifier indicates the physical device to be opened.

*wstype* — **specified by user, 8-byte character string**
One of the graPHIGS API supported workstation types.

**Error Codes**

**21** CONNECTION IDENTIFIER IS INVALID

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**24** SPECIFIED WORKSTATION IDENTIFIER ALREADY IS IN USE

**26** SPECIFIED WORKSTATION CANNOT BE OPENED

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**217** RESOURCE CREATION REQUEST EXCEEDS NUCLEUS TABLE CAPACITY

**514** INAPPROPRIATE DEVICE FOR WORKSTATION TYPE

**581** PROCOPT SPECIFIES INVALID VIEW TABLE SIZE FOR WORKSTATION

**583** PROCOPT SPECIFIES INVALID NUMBER OF INPUT DEVICES FOR WORKSTATION

**585** PROCOPT SPECIFIES INVALID KEYBOARD FOR WORKSTATION

**586** PROCOPT SPECIFIES INVALID DISPLAY MODEL NUMBER FOR WORKSTATION

**587** PROCOPT SPECIFIES INVALID ECHO METHOD FOR WORKSTATION

**588** PROCOPT SPECIFIES INVALID FRAME BUFFER VALUE FOR WORKSTATION

**596** PROCOPT SPECIFIES INVALID NUMBER OF POLYLINE TABLE ENTRIES

**597** PROCOPT SPECIFIES INVALID NUMBER OF POLYMARKER TABLE ENTRIES

**598** PROCOPT SPECIFIES INVALID NUMBER OF TEXT TABLE ENTRIES

**599** PROCOPT SPECIFIES INVALID NUMBER OF EDGE TABLE ENTRIES

**600** PROCOPT SPECIFIES INVALID NUMBER OF DEPTH CUE TABLE ENTRIES

**601** PROCOPT SPECIFIES INVALID NUMBER OF LIGHT SOURCE TABLE ENTRIES

**602** PROCOPT SPECIFIES INVALID NUMBER OF INTERIOR TABLE ENTRIES

**648** PROCOPT SPECIFIES AN INVALID DISPLAY WIDTH AND/OR HEIGHT

**649** PROCOPT SPECIFIES AN INVALID IMAGE OUTPUT FORMAT

**650** PROCOPT SPECIFIES AN INVALID HLHSR COORDINATE SYSTEM

**Related Subroutines**

**GPATR**
Attach Resource

**GPCLWS**
Close Workstation

**GPCRWS**
Create Workstation

**GPDTR**
Detach Resource

**GPQRCT**
Inquire Realized Connection Type

**GPQWCT**
Inquire Workstation Connection and Type

**GPQWTN**
Inquire List of Available Workstation Types on Nucleus

**RCP code**

201327361 (X'0C000301')

## GPPW - Set Password

**GPPW (*type*, *id*, *pass*)**

**Purpose**

Use **GPPW** to assign a password to a resource. The graPHIGS API uses this password to validate the Attach Resource (**GPATR**) request from other application processes.

The *id* parameter specifies an identifier that your application assigned to the resource. The graPHIGS API uses the *type* parameter to identify the type of resource that the *id* parameter specifies. For example, if the *type* parameter specifies `1=WORKSTATION`, then the graPHIGS API uses the *id* parameter to specify a workstation identifier (*wsid*) whose password is set.

Your application must provide this password (as well as the nucleus's resource identifier for the resource) to other applications that you authorize to access the resource. Use the Send Private Message (**GPSPMS**) subroutine to send this information. The Attach Resource (**GPATR**) subroutine then uses this information to authorize access to the resource.

For more information about the use of resource passwords and application messages, see *The graPHIGS Programming Interface: Understanding Concepts*.

**Parameters**

*type* **— specified by user, fullword integer**
Resource type (`1=WORKSTATION, 2=STRUCTURE_STORE, 3=IMAGE_BOARD, 4=FONT_DIRECTORY, 5=ARCHIVE_FILE`).

*id* **— specified by user, fullword integer**
An identifier of the resource.

*pass* **— specified by user, fullword integer**
Password.

**0=None**
Only the application that created the resource can access the resource.

**-1=All** Any application can access the resource.

**Unique Integer**
Only applications that supply the password can access the resource.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**211** RESOURCE TYPE IS INVALID

**214** PASSWORD CANNOT BE CHANGED FROM THIS APPLICATION

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**222** SPECIFIED STRUCTURE STORE DOES NOT EXIST

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

**242** SPECIFIED FONT DIRECTORY DOES NOT EXIST

### Related Subroutines

**GPATR**
Attach Resource

**GPSPMS**
Send Private Message

### RCP code

201341699 (X'0C003B03')

## GPRAST - Redraw All Structures

**GPRAST (*wsid*, *flag*)**

### Purpose

Use **GPRAST** to redraw all structures on the specified workstation.

When your application invokes this subroutine, the graPHIGS API executes the actions in the sequence outlined below:

1. The graPHIGS API executes all deferred actions for the specified workstation without an intermediate clearing of the display surface.
2. If the control flag is set to `1=CONDITIONALLY`, then the graPHIGS API clears the display surface only if the Display Surface Empty entry in the WSL is `1=NOT_EMPTY` If the control flag is set to `2=ALWAYS`, then the graPHIGS API clears the display surface regardless of the setting of the Display Surface Empty entry. At the conclusion of this step, the graPHIGS API sets the entry to `2=IS_EMPTY`.
3. If the workstation window, viewport, view matrix, view mapping, or view characteristics have changed for any view, then the graPHIGS API assigns the current entries in the Workstation State List (WSL) to the corresponding values from the requested entries. The Transformation Update State is set to `1=NOT_PENDING` and the transformation view matrix, view mapping, and view characteristics change flags are set to `NOTCHANGED`.
4. Finally, the graPHIGS API retraverses all structures associated with views for this workstation. If the set of structures associated with this workstation's views are not empty, retraversal usually sets the Display Surface Empty entry in the WSL to `1=NOT_EMPTY`.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*flag* **— specified by user, fullword integer**
Control flag (`1=CONDITIONALLY`, `2=ALWAYS`).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**301** CONTROL FLAG IS INVALID

**Related Subroutines**

**GPSHDF**
Set Shell Deferral State

**RCP code**

201327873 (X'0C000501')

## GPSBMS - Send Broadcast Message

**GPSBMS (*ncid*, *major*, *minor*, *len*, *msg*)**

**Purpose**

Use **GPSBMS** to send a broadcast message to all application processes that can accept messages and are connected to the specified nucleus. To be able to accept messages, you previously must have set the application's message password to `-1=ALL`. For information about sending and receiving messages between application processes, see *The graPHIGS Programming Interface: Understanding Concepts*.

**Parameters**

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*major* **— specified by user, fullword integer**
Major code.

*minor* **— specified by user, fullword integer**
Minor code.

*len* **— specified by user, fullword integer**
Length of the message in bytes (>=0).

*msg* **— specified by user, variable data**
Message.

**Error Codes**

**197** MESSAGE STRING LENGTH < ZERO

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**Related Subroutines**

**GPAWEV**
Await Event

**GPCVD**
Convert Data

**GPMSPW**
Set Message Password

**GPQSH**
Inquire Shell Identifier

**GPSPMS**
Send Private Message

**RCP code**

201341442 (X'0C003A02')

## GPSDAL - Sound Alarm

**GPSDAL (*wsid*)**

**Purpose**

Use **GPSDAL** to sound the alarm (bell, buzzer, etc.) at the specified workstation. If an alarm is not present, or the workstation does not support this subroutine, then the subroutine is ignored.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**Related Subroutines**

**GPQES**
Inquire List of Available Escape Subroutines

**RCP code**

201327877 (X'0C000505')

## GPSHDF - Set Shell Deferral State

**GPSHDF (*deferral*, *update*)**

**Purpose**

Use **GPSHDF** to set the shell deferral mode and the update notification mode.

A shell deferral mode set to `2=DEFERRED` controls whether the following workstation control subroutines cause the graPHIGS API to send buffered requests to the nucleus which contains the workstation:

- Set Deferral State (**GPDF**)
- Redraw All Structures (**GPRAST**)
- Update Workstation (**GPUPWS**)
- Update Workstation Asynchronous (**GPUPWA**)

A shell deferral mode set to `3=DEFERRED_PLUS_MSGS` controls whether the following subroutines cause the graPHIGS API to send buffered requests to the nucleus:

- Set Deferral State (**GPDF**)
- Redraw All Structures (**GPRAST**)

- Update Workstation (**GPUPWS**)
- Update Workstation Asynchronous (**GPUPWA**)
- Send Private Message (**GPSBMS**)
- Send Broadcast Message(**GPSPMS**)

When the shell deferral mode is set to 1=FLUSH (default), each of the above subroutines results in sending buffered API requests to the nucleus. When the shell deferral mode is set to 2=DEFERRED or 3=DEFERRED_PLUS_MSGS, the graPHIGS API suppresses this automatic initiation for the specified subroutines. The application can use the Synchronize (**GPSYNC**) subroutine explicitly to send a buffered request to the nucleus. However, there are other graPHIGS API subroutines that the application has no control over that implicitly cause the API buffered requests to be sent to the nucleus. For information about buffered requests being sent to the nucleus, see *The graPHIGS Programming Interface: Understanding Concepts*.

The update notification mode controls whether or not the graPHIGS API notifies the application about the completion of an update process initiated by the Redraw All Structures (**GPRAST**) or the Update Workstation (**GPUPWS**) subroutine.

When the update notification mode is set to 1=NO (default), the graPHIGS API does not notify the application when the graPHIGS API completes the update process. The nucleus ensures that the update process is completed without any intermediate interruption but the application cannot know when it has completed. When the update notification mode is 2=YES, the application is asynchronously notified by an Update Completion Event that the graPHIGS API completed the update process. For information about Update Completion Events, see *The graPHIGS Programming Interface: Technical Reference*.

> **Note:** The Update Completion Event is generated even for the Update Workstation subroutine with regeneration flag 1=POSTPONE. In this case, the event is generated immediately after the Update Workstation request is processed by the nucleus and it may contain a display status of 2=DEFERRED.

**Parameters**

*deferral* **— specified by user, fullword integer**
Shell deferral mode (1=FLUSH, 2=DEFERRED, 3 DEFERRED_PLUS_MSGS).

*update* **— specified by user, fullword integer**
Update notification mode (1=NO, 2=YES).

**Error Codes**

**205** SHELL DEFERRAL MODE IS INVALID

**209** UPDATE NOTIFICATION MODE IS INVALID

**Related Subroutines**

**GPAWEV**
Await Event

**GPEVHN**
Define Event Handling Subroutine

**GPSYNC**
Synchronize

**GPQSHD**
Inquire Shell Deferral State

**RCP code**

201337603 (X'0C002B03')

# GPSPMS - Send Private Message

**GPSPMS (*ncid*, *shid*, *pass*, *major*, *minor*, *len*, *msg*)**

### Purpose

Use **GPSPMS** to send an application message to another application process connected to the specified nucleus. To be able to send private messages, you previously must have set the message of the target application process to `-1=ALL` or set a unique password that matches the password supplied in the *pass* parameter. The message is limited to the size of the outbound buffer of the sending application and the size of the inbound buffer and event queue of the target application.

For information about sending and receiving messages between application processes, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*shid* **— specified by user, fullword integer**
Target application identifier.

*pass* **— specified by user, fullword integer**
Password. This must match the password of the target application process.

*major* **— specified by user, fullword integer**
Major code.

*minor* **— specified by user, fullword integer**
Minor code.

*len* **— specified by user, fullword integer**
Length of the message in bytes (>=0).

*msg* **— specified by user, variable data**
Message.

### Error Codes

**197** MESSAGE STRING LENGTH < ZERO

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**207** SPECIFIED APPLICATION IDENTIFIER DOES NOT EXIST

**213** SPECIFIED PASSWORD IS INCORRECT

### Related Subroutines

**GPAWEV**
Await Event

**GPCVD**
Convert Data

**GPMSPW**
Set Message Password

**GPQNCR**
Inquire Nucleus Resource Identifier

**GPQSH**
Inquire Shell Identifier

**GPSBMS**
Send Broadcast Message

**RCP code**

201341441 (X'0C003A01')

## GPSYNC - Synchronize

**GPSYNC (*ncid*, *synch*)**

### Purpose

Use **GPSYNC** to send buffered requests to the specified nucleus.

When the synchronization mode (*synch*) parameter is set to `1=NOWAIT`, the graPHIGS API sends the buffer to the nucleus and returns control to the application before the nucleus processes the buffer.

When the synchronization mode (*synch*) parameter is set to `2=SYNC_WAIT`, the graPHIGS API does not return control to the application until the nucleus processes all requests and the graPHIGS API has reported any errors.

While debugging your application, you may wish to send each buffered request after every graPHIGS API subroutine call so that errors detected by the nucleus can be processed immediately. This synchronization can be accomplished through a second-level error exit defined using the Specify Error Exit and Error Threshold (**GPEXIT**) subroutine. With the error severity threshold set to a value of zero, your application defined error exit will receive control after every graPHIGS API subroutine call (except the **GPEXIT** subroutine) Therefore, a call to **GPSYNC** from a second-level error exit sends all buffered requests before the graPHIGS API returns control to your application.

This technique will degrade the performance of your application and is only recommended for debugging purposes. For information about second-level error handlers and error severities, see *The graPHIGS Programming Interface: Understanding Concepts*.

Preventing second-level error exits from causing infinite loops is discussed under the Specify an Error Exit and Error Threshold (**GPEXIT**) subroutine.

### Parameters

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*synch* **— specified by user, fullword integer**
Synchronization mode (`1=NOWAIT`, `2=SYNC_WAIT`).

### Error Codes

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**206** SYNCHRONIZATION MODE IS INVALID

### Related Subroutines

None

**RCP code**

201337604 (X'0C002B04')

## GPTRCE - Internal Trace Control

**GPTRCE (*control*)**

**Purpose**

Use **GPTRCE** to control the tracing facilities of the graPHIGS API.

The tracing information may be helpful in the diagnosis of problems and may be requested by IBM for the diagnosis of reported problems.

By default, trace is set to a value of zero (trace is not active).

**Parameters**

*control* **— specified by user, fullword integer**
A fullword integer whose contents determine the type of tracing performed. A value of zero deactivates all tracing. For information about this parameter, see *The graPHIGS Programming Interface: Writing Applications*.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201337863 (X'0C002C07')

## GPUPWA - Update Workstation Asynchronous

**GPUPWA (*wsid*)**

**Purpose**

Use **GPUPWA** to immediately update the specified workstation. The system does not wait for the completion of any retraversal which may be under way before the occurrence of the next transaction with this device.

If the application requires the viewing of each update in separate frames, your application must send the Update Workstation (**GPUPWS**) subroutine call. **GPUPWA** allows the combination of consecutive frames into one retraversal when possible.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**Related Subroutines**

**GPSHDF**
Set Shell Deferral State

**RCP code**

201327879 (X'0C000507')

## GPUPWS - Update Workstation

**GPUPWS (*wsid*, *regen*)**

**Purpose**

Use **GPUPWS** to update the specified workstation. If any actions have been deferred or simulated as a result of explicit traversal, the graPHIGS API performs a retraversal to update the display.

This subroutine function executes all deferred actions for the specified workstation without intermediate clearing of the display surface. If the regeneration flag is set to `2=PERFORM` and the New Frame Action Necessary at Update entry in the Workstation State List (WSL) is set to `2=YES`, then the graPHIGS API executes all the actions in the sequence outlined below:

1. If the Display Surface Empty entry in the WSL is `1=NOT_EMPTY`, then the graPHIGS API clears the display surface. At the conclusion to this step (Step 1), the graPHIGS API sets the entry to `2=IS_EMPTY`.
2. If the workstation window, viewport, view matrix, view mapping, or view characteristics have changed for any view, then the graPHIGS API assigns the current entries in the Workstation State List (WSL) to the corresponding values from the requested entries. The Transformation Update State entry is set to `1=NOT_PENDING` and the transformation view matrix, view mapping and view characteristics change flags are set to `NOTCHANGED`.
3. The graPHIGS API re-displays all structures associated with views on this workstation. Usually, this action sets the Display Surface Empty entry in the WSL to `1=NOTEMPTY`. The Transformation Update State entry is set to `1=NOT_PENDING`.
4. The state, New Frame Action Necessary at Update, is set to `1=NO` in the WSL. If the value of the state, New Frame Action Necessary at Update is set to `2=YES` and the regeneration flag is set to `2=PERFORM`, then this subroutine is functionally equivalent to the Redraw All Structures (**GPRAST**) subroutine.
5. If the current state of visual representation is set to `3=SIMULATED` and the modification mode is UQUM (Quick Update Method), then the state of visual representation remains unchanged. In all other cases, the state of visual representation is set to `1=CORRECT`.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*regen* — **specified by user, fullword integer**
Regeneration flag (`1=POSTPONE, 2=PERFORM`).

When the regeneration flag is set to 1=POSTPONE, the device sends the pending update information without forcing a retraversal if possible. This is workstation dependent.

When the regeneration flag is set to 2=PERFORM, the device sends the pending update information and immediately initiates a retraversal.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**302** REGENERATION FLAG IS INVALID

**Related Subroutines**

**GPSHDF**
Set Shell Deferral State

**GPQDV**
Inquire Deferral and Update State Values

**RCP code**

201327874 (X'0C000502')

# Chapter 3. Output Primitives

These subroutines address the specification and creation of output primitives, which are structure elements. Many have both two- and three-dimensional forms and are displayed when the structure elements defining them are encountered during structure traversal. To use primitive subroutines, the structure state must be Structure Open (STOP).

Some output primitive subroutines accept lists of structure element data. These lists often are accompanied by a *width* parameter which allows the application program to easily input an array of structure element data that may contain additional application specific information. When these arrays are accepted as parameters, only the structure element data is stored in the structure.

For all two-dimensional output primitive subroutines, the *z* coordinate is assumed to equal zero by default.

If a specified workstation does not support a requested output primitive, then the graPHIGS API updates only the element number in the graPHIGS API traversal state list.

> **Note:** When an application inserts an element into the open structure following the element pointer, the pointer updates to that element.

## GPAN2 - Annotation Text 2

**GPAN2 (*point*, *length*, *text*)**

### Purpose

Use **GPAN2** to insert an Annotation Text 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Annotation Text 2 structure element depending on the current edit mode.

This structure element specifies a string of text in two-dimensional Modelling Coordinate (MC) space to be drawn at the specified location in a plane parallel to the view plane.

At the time this structure element is created, the current Text Character Set value in the graPHIGS API State List is bound to this character string.

The size and orientation of annotation text is not affected by rotation and scaling. Only its position changes.

### Parameters

*point* **— specified by user, 2 short floating-point numbers (MC)**
*x* and *y* coordinates of the annotation text position.

*length* **— specified by user, fullword integer**
Length of the annotation text string in bytes (>=0).

*text* **— specified by user, variable length character string**
Text string to be displayed.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **108** | NUMBER OF CHARACTERS IN TEXT STRING < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPAAL** | Set Annotation Alignment |
| **GPAH** | Set Annotation Height |
| **GPAHSC** | Set Annotation Height Scale Factor |
| **GPAPT** | Set Annotation Path |
| **GPAUP** | Set Annotation Up Vector |
| **GPCHPM** | Set Character Positioning Mode |
| **GPCHSP** | Set Character Spacing |
| **GPTXCD** | Set Text Color Direct |
| **GPTXCI** | Set Text Color Index |
| **GPTXFO** | Set Text Font |
| **GPTXI** | Set Text Index |
| **GPTXPR** | Set Text Precision |
| **GPQXAF** | Inquire Extended Annotation Font Characteristics |

### RCP code

201328139 (X'0C00060B')

## GPAN3 - Annotation Text 3

**GPAN3 (*point*, *length*, *text*)**

### Purpose

Use **GPAN3** to insert an Annotation Text 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Annotation Text 3 structure element depending on the current edit mode.

This structure element specifies a string of text in three-dimensional Modelling Coordinate (MC) space to be drawn at the specified location in a plane parallel to the view plane.

At the time this structure element is created, the current Text Character Set value in the graPHIGS API State List is bound to this character string.

The size and orientation of annotation text is not affected by rotation and scaling. Only its position changes.

### Parameters

*point* **— specified by user, 3 short floating-points numbers (MC)**
*x*, *y*, and *z* coordinates of the annotation text position.

*length* **— specified by user, fullword integer**
Length of the annotation text string in bytes (>=0).

*text* **— specified by user, variable length character string**
Text string to be displayed.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |

### Related Subroutines

**GPAAL** Set Annotation Alignment
**GPAH** Set Annotation Height
**GPAHSC** Set Annotation Height Scale Factor
**GPAPT** Set Annotation Path
**GPAUP** Set Annotation Up Vector
**GPCHPM** Set Character Positioning Mode
**GPCHSP** Set Character Spacing
**GPTXCD** Set Text Color Direct
**GPTXCI** Set Text Color Index
**GPTXFO** Set Text Font
**GPTXI** Set Text Index
**GPTXPR** Set Text Precision
**GPQXAF** Inquire Extended Annotation Font Characteristics

### RCP code

201328140 (X'0C00060C')

## GPANR2 - Annotation Text Relative 2

**GPANR2 (*refpt*, *textvec*, *length*, *text*)**

### Purpose

Use **GPANR2** to insert a two-dimensional Annotation Text Relative 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Annotation Text Relative 3 structure element depending on the current edit mode.

When you create this structure element, the current Text Character Set value in the graPHIGS API State List is bound to this character string.

During structure traversal, this element annotates the specified reference point according to the annotation style in the traversal state list. The annotation string is positioned according to *textvec* which defines the origin of a local text coordinate system relative to *refpt* after transformation to NPC. The text plane is always parallel to the *x*, *y* plane (*z*=transformed location) in NPC. If the resulting text position is outside the usable NPC space [0,1]x[0,1]x[0,1], then the graPHIGS API may clip part or all of the string.

The application positions and renders the text string in the local coordinate system according to the annotation text attributes in the traversal state list. If the graPHIGS API clips the specified reference point (*refpt*) to NPC during structure traversal, then no representation for this primitive is displayed. If the graPHIGS API does not clip the reference point (*refpt*), then the graPHIGS API clips the displayed representation according to the rules for the corresponding primitive type (e.g., text, polyline, etc.).

If the style (*style*) parameter on the Set Annotation Style (**GPAS**) subroutine is set to `2=LEAD_LINE`, then after transformation, the graPHIGS API draws a single line segment from the reference point (*refpt*) to the origin of the local text coordinate system using the polyline attributes in the traversal state list.

### Parameters

*refpt* **— specified by user, 2 short floating-point numbers (MC)**
*x* and *y* coordinates of the reference location that is to be annotated.

*textvec* **— specified by user, 2 short floating-point numbers (NPC)**
*x* and *y* components of a vector which defines the location to position the origin of the local text coordinate system relative to *refpt* after transformation to NPC.

*length* **— specified by user, fullword integer**
Length of text string in bytes (>=0).

*text* **— specified by user, variable length character string**
Text string to be displayed.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **108** | NUMBER OF CHARACTERS IN TEXT STRING < ZERO |

### Related Subroutines

**GPAS** Set Annotation Style

### RCP code

201328152 (X'0C000618')

## GPANR3 - Annotation Text Relative 3

**GPANR3 (*refpt*, *textvec*, *length*, *text*)**

### Purpose

Use **GPANR3** to insert an Annotation Text Relative 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Annotation Text Relative 3 structure element depending on the current edit mode.

When you create this structure element, the current Text Character Set value in the graPHIGS API State List is bound to this character string.

During structure traversal, this element annotates the specified reference point according to the annotation style in the traversal state list. The annotation string is positioned according to *textvec* which defines the origin of a local text coordinate system relative to *refpt* after transformation to NPC. The text plane is always parallel to the *x*, *y* plane in NPC. If the resulting text position is outside the usable NPC space [0,1]x[0,1]x[0,1], then the graPHIGS API may clip part or all of the string.

The application positions and renders the text string in the local coordinate system according to the annotation text attributes in the traversal state list. If the graPHIGS API clips the specified reference point (*refpt*) to NPC during structure traversal, then no representation for this primitive is displayed. If the graPHIGS API does not clip the reference point (*refpt*), then the graPHIGS API clips the displayed representation according to the rules for the corresponding primitive type (e.g., text, polyline, etc.).

If the style (*style*) parameter on the Set Annotation Style (**GPAS**) subroutine is set to `2=LEAD_LINE`, then after transformation, the graPHIGS API draws a single line segment from the reference point (*refpt*) to the origin of the local text coordinate system using the polyline attributes in the traversal state list.

### Parameters

*refpt* — **specified by user, 3 short floating-point numbers (MC)**
*x*, *y*, and *z* coordinates of the reference location that is to be annotated.

*textvec* — **specified by user, 3 short floating-point numbers (NPC)**
*x*, *y*, and *z* components of a vector which defines the location to position the origin of the local text coordinate system relative to *refpt* after transformation to NPC.

*length* — **specified by user, fullword integer**
Length of text string in bytes (>=0).

*text* — **specified by user, variable length character string**
Text string to be displayed.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **108** | NUMBER OF CHARACTERS IN TEXT STRING < ZERO |

### Related Subroutines

**GPAS** Set Annotation Style

### RCP code

201328153 (X'0C000619')

## GPCFA2 - Composite Fill Area 2

**GPCFA2 (*ncontour*, *ncurve*, *curveinfo*, *knot*, *tess*, *vwidth*, *vdata*)**

### Purpose

Use **GPCFA2** to insert a Composite Fill Area 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Composite Fill Area 2 structure element depending on the current edit mode.

During structure traversal, the planar area geometry defined by the specified contours is drawn using the polygon attributes in the traversal state list. Each contour consists of one or more types of curves. The normal on Composite Fill Area 2 is (0,0,1).

When the current edge flag is set to `2=ON` or `3=GEOMETRY_ONLY`, each curve with its boundary flag `ON` is rendered as an edge of the area geometry. You must connect curves within a contour in a head to tail fashion. To ensure that the contour is closed, the graPHIGS API connects each pair of curves by a straight line as is the first point and the last point. The boundary flag for the connecting line segment is determined by the boundary flag of the preceding curve.

Polygon attributes are applied to this primitive.

**GPCFA2** is identified as GDP 1027.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*ncontour* **— specified by user, fullword integer**
Number of contours to be generated (>=0).

*ncurve* **— specified by user, array of fullword integers**
Number of curves in each contour. Each entry must be greater than or equal to one. The length of this array is defined by the value of the *ncontour* parameter.

*curveinfo* **— specified by user, array of 6 fullwords of data**
Array containing information about each curve. Each entry of this parameter must have the following fields in this order. However, for some curve types, one or more fields may be ignored.

**Type of curve — fullword integer**

This field must contain one of the following values:

**1 -** Line Segments
**2 -** Elliptical Arc
**3 -** Non-Uniform B-Spline Curve

**Options — fullword integer**

This parameter specifies various options for the curve. Each option is specified as a bit in this word. The following bits are currently defined:

| Bit | Meaning |
|---|---|
| **0-28** | Reserved. Must be set to zero. |
| **29** | Tessellation quality flag.<br>This option is valid for the Non-Uniform B-Spline Curve only. If set, a tessellation quality value for each span of this curve is specified in the *tess* parameter. |
| **30** | Weight flag.<br>This option is valid for the Non-Uniform B-Spline Curve only. If set, the curve is rational and the weight is specified for each control point in the *vdata* parameter. |
| **31** | Boundary flag.<br>If set, the curve is treated as an edge of the composite fill area. |

**Order — fullword integer**

Order of the curve. For each curve type, this parameter has the following meaning:

| | |
|---|---|
| **1 -** | Ignored. |
| **2 -** | Ignored. |
| **3 -** | Corresponds to the *order* parameter of the Non-Uniform B-Spline Curve 2. |

**Number of vectors — fullword integer**

Number of entries of the *vdata* parameter used to define the curve. This parameter has the following meaning for each curve type:

| | |
|---|---|
| **1 -** | Corresponds to the *npoint* parameter of Polyline 2. The specified number's entries of the *vdata* parameter are used as its *pointlist* parameter. |
| **2 -** | Must be three. Three entries of the *vdata* parameter are used as the center, first and second reference vectors of the Elliptical Arc 2. |

**3 -** Corresponds to the *npoint* parameter of the Non-Uniform B-Spline Curve 2. The specified number's entries of the *vdata* parameter are used as its *ctlpts* parameter.

**Start — short floating-point number**

The parameter value representing the start point of the curve. For each curve type, this parameter has the following meaning:

**1 -** Ignored.
**2 -** Corresponds to the *startv* parameter of the Elliptical Arc 2.
**3 -** Corresponds to the *tmin* parameter of the Non-Uniform B-Spline Curve 2.

**End — short floating-point number**

The parameter value representing the end point of the curve. For each curve type, this parameter has the following meaning:

**1 -** Ignored.
**2 -** Corresponds to the *endv* parameter of the Elliptical Arc 2.
**3 -** Corresponds to the *tmax* parameter of the Non-Uniform B-Spline Curve 2.

*knot* **— specified by user, array of short floating-point numbers**
Array of knot values. This array must contain one list for each Non-Uniform B-Spline curve within the *curveinfo* parameter. For other curves, this array is not referenced. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo*. The length of each list is equal to the *order* + `number of vectors` for the curve.

*tess* **— specified by user, array of short floating-point numbers**
Array of tessellation quality values. This array must contain one list for each Non-Uniform B-Spline curve with a tessellation quality flag set to a value of one (specified). For other curves, this array is not referenced. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo*. The length of each list is equal to the `number of vectors` - *order* + 1 of the curve.

*vwidth* **— specified by user, fullword integer**
Specifies the number of fullwords between each entry of the array in *vdata*. If there is any rational curve in the *curveinfo* parameter, this parameter must be at least three. Otherwise, it must be larger than or equal to a value of two.

*vdata* **— specified by user, array of short floating-point numbers**
This array must contain one list for each curve. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo*. The length of each list is equal to the `number of vectors` specified in the *curveinfo* parameter.

For each entry, the following fields are defined and the fields must be specified in this order without any gap:

***x* and *y* components —** two short floating-point numbers
**weight —** short floating-point number

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)
**100** NUMBER OF POINTS < ZERO

| | |
|---|---|
| **107** | REFERENCE VECTORS ARE COLINEAR |
| **341** | ORDER OF BASIS FUNCTION < TWO |
| **342** | ORDER IS GREATER THAN NUMBER OF CONTROL POINTS |
| **343** | KNOT VECTOR IS INVALID |
| **345** | WEIGHT IN CONTROL POINT IS <= ZERO |
| **347** | PARAMETER LIMITS ARE OUTSIDE VALID PARAMETER RANGE |
| **348** | MINIMUM PARAMETER LIMIT > MAXIMUM |
| **353** | NUMBER OF CONTOURS < ZERO |
| **354** | NUMBER OF CURVES PER CONTOUR < ONE |
| **355** | CURVE TYPE IS INVALID |
| **361** | CURVE OPTIONS FIELD IS INVALID |
| **362** | TESSELLATION CONTROL VALUE IS INVALID |
| **557** | WIDTH PARAMETER < MINIMUM ALLOWED |

### Related Subroutines

| | |
|---|---|
| **GPBICD** | Set Back Interior Color Direct |
| **GPBICI** | Set Back Interior Color Index |
| **GPBSCD** | Set Back Specular Color Direct |
| **GPBSCI** | Set Back Specular Color Index |
| **GPBSPR** | Set Back Surface Properties |
| **GPECD** | Set Edge Color Direct |
| **GPECI** | Set Edge Color Index |
| **GPEI** | Set Edge Index |
| **GPELT** | Set Edge Linetype |
| **GPESC** | Set Edge Scale Factor |
| **GPFDMO** | Set Face Distinguish Mode |
| **GPICD** | Set Interior Color Direct |
| **GPICI** | Set Interior Color Index |
| **GPII** | Set Interior Index |
| **GPIS** | Set Interior Style |
| **GPISI** | Set Interior Style Index |
| **GPLMO** | Set Lighting Calculation Mode |
| **GPLSS** | Set Light Source State |
| **GPPGC** | Set Polygon Culling |
| **GPSAC** | Set Surface Approximation Criteria |
| **GPSCD** | Set Specular Color Direct |
| **GPSCI** | Set Specular Color Index |
| **GPSPR** | Set Surface Properties |
| **GPTCAC** | Set Trimming Curve Approximation Criteria |

### RCP code

201345026 (X'0C004802')

## GPCHL2 - Character Line 2

**GPCHL2 (*startp*, *endp*, *nomhgt*, *char*)**

### Purpose

Use **GPCHL2** to insert a Character Line structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Character Line 2 structure element depending on the current edit mode.

When you create **GPCHL2**, the graPHIGS API binds the current character set to this structure element.

During structure traversal, the graPHIGS API draws an integral number of characters between the starting point (*startp* and the ending point (*endp*). The character set and character code (*char*) within the structure element in conjunction with the current font in the traversal state list, determine the characters drawn at each point. If the character set and font combination is not currently active, then the graPHIGS API applies the normal defaulting rules. The height of the characters is the product of the nominal character height (*nomhgt*) and the current Character Line Scale Factor in the traversal state list. The height is subject to all transformations. The actual width of the characters may be adjusted slightly so that there is an integral number between the two endpoints.

The application draws the character in the character line in the *x*, *y* plane in Modelling Coordinate (MC) space. For the exact method on how text attributes are applied to this primitive, see *The graPHIGS Programming Interface: Understanding Concepts*.

If the currently active character set and font combination is a double bye character set, then the graPHIGS API ignores this structure element.

GPCHL2 is identified as GDP 1039.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*startp* **— specified by user, 2 short floating-point numbers (MC)**
Starting point.

*endp* **— specified by user, 2 short floating-point numbers (MC)**
End point.

*nomhgt* **— specified by user, short floating-point number (MC)**
Nominal character height (>0).

*char* **— specified by user, 1-byte character string**
Character code.

**Error Codes**

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **78** | CHARACTER HEIGHT VALUE <= ZERO |

**Related Subroutines**

| | |
|---|---|
| **GPCHLS** | Set Character Line Scale Factor |
| **GPCHPM** | Set Character Positioning Mode |
| **GPCHXP** | Set Character Expansion Factor |
| **GPTXCD** | Set Text Color Direct |
| **GPTXCI** | Set Text Color Index |
| **GPTXFO** | Set Text Font |
| **GPTXI** | Set Text Index |

### RCP code

201344514 (X'0C004602')

## GPCR2 - Circle 2

**GPCR2 (*center*, *radius*)**

### Purpose

Use **GPCR2** to insert a Circle 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Circle 2 structure element depending on the current edit mode.

**GPCR2** draws a line and uses the polyline attributes.

**GPCR2** is identified as GDP 1005.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*center* **— specified by user, 2 short floating-point numbers (MC)**
Center of circle.

*radius* **— specified by user, short floating-point number (MC)**
Radius of circle (>=0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **582** | RADIUS SPECIFIED < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPLT** | Set Linetype |
| **GPLWSC** | Set Linewidth Scale Factor |
| **GPPLCD** | Set Polyline Color Direct |
| **GPPLCI** | Set Polyline Color Index |
| **GPPLET** | Set Polyline End Type |
| **GPPLI** | Set Polyline Index |

### RCP code

201328146 (X'0C000612')

## GPCRA2 - Circular Arc 2

**GPCRA2 (*center*, *radius*, *startang*, *endang*)**

### Purpose

Use **GPCRA2** to insert a Circular Arc 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Circular Arc 2 structure element depending on the current edit mode.

**GPCRA2** draws the arc from the starting angle (*startang*) to the end angle (*endang*) through increasing angles. The x-axis, in Modeling Coordinates (MC), serves as the origin for the measurement of angles. All angles are specified in radians. Polyline attributes are applied to this primitive.

**GPCRA2** is identified as GDP 1006.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*center* **— specified by user, 2 short floating-point numbers (MC)**
Center of arc.

*radius* **— specified by user, short floating-point number (MC)**
Radius of arc (>=0).

*startang* **— specified by user, short floating-point number**
Start angle for creation of arc (specified in radians).

*endang* **— specified by user, short floating-point number**
End angle for creation of arc (specified in radians).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **582** | RADIUS SPECIFIED < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPLT** | Set Linetype |
| **GPLWSC** | Set Linewidth Scale Factor |
| **GPPLCD** | Set Polyline Color Direct |
| **GPPLCI** | Set Polyline Color Index |
| **GPPLET** | Set Polyline End Type |
| **GPPLI** | Set Polyline Index |

### RCP code

201328147 (X'0C000613')

## GPDPL2 - Disjoint Polyline 2

**GPDPL2 (*npoint*, *width*, *pointlist*, *mdarray*)**

### Purpose

Use **GPDPL2** to create a two-dimensional, disjoint polyline element and insert it into the open structure following the element pointer or to replace the element pointed at by the element pointer with a two-dimensional, disjoint polyline element depending on the current edit mode.

This structure element defines multiple two-dimensional polylines. Each polyline consists of a number of two-dimensional points (*x*, *y*) in Modelling Coordinate (MC) space that the graPHIGS API connects by straight lines starting with the first point in the list and ending with the last point in the list. The *z* coordinate is assumed to be zero.

If the application specifies less than two points in the list, then the graPHIGS API does not generate any output for that list. If the application specifies two contiguous points as the same point, then the graPHIGS API generates a point of one pixel in size.

The move/draw indicators(*mdarray*) specify the action required to go from one corresponding point to the next; that is, whether to draw or to move to the next point.

Polyline attributes are applied to this primitive.

**GPDPL2** is identified as GDP 1004.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**)subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*npoint* **— specified by user, fullword integer**
Number of points (>=0).

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent*x* values (>=2).

*pointlist* **— specified by user, array of short floating-point numbers (MC)**
Array of points specified in row order. The input array contains a list of points in which subsequent*x* values are separated by *width* fullwords.

*mdarray* **— specified by user, array of fullword integers**
Move/draw indicators (`1=MOVE, 2=DRAW`). The last entry of this array is disregarded.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **100** | NUMBER OF POINTS < ZERO |
| **555** | MOVE/DRAW INDICATOR IS INVALID |
| **557** | WIDTH PARAMETER < MINIMUM ALLOWED |

### Related Subroutines

| | |
|---|---|
| **GPLT** | Set Linetype |
| **GPLWSC** | Set Linewidth Scale Factor |
| **GPPLCD** | Set Polyline Color Direct |
| **GPPLCI** | Set Polyline Color Index |
| **GPPLET** | Set Polyline End Type |
| **GPPLI** | Set Polyline Index |

### RCP code

# GPDPL3 - Disjoint Polyline 3

**GPDPL3 (*npoint*, *width*, *pointlist*, *mdarray*)**

### Purpose

Use **GPDPL3** to create a three-dimensional, disjoint polyline element and insert it into the open structure following the element pointer or to replace the element pointed at by the element pointer with a three-dimensional, disjoint polyline element depending on the current edit mode.

This structure element defines multiple three-dimensional polylines. Each polyline consists of three-dimensional points (*x*, *y*, *z*) in Modelling Coordinate (MC) space that the graPHIGS API connects by straight lines starting with the first point in the list and ending with the last point in the list.

If the application specifies less than two points, then the graPHIGS API does not generate any output for that list. If the application specifies two contiguous points as the same point, then the graPHIGS API generates a point of one pixel in size.

The move/draw indicators (*mdarray*) specify the action required to go from one corresponding point to the next; that is, whether to draw or move to the next point.

Polyline attributes are applied to this primitive.

**GPDPL3** is identified as GDP 1003.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*npoint* **— specified by user, fullword integer**
Number of points (>=0).

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values (>=3).

*pointlist* **— specified by user, array of short floating-point numbers (MC)**
Array of points specified in row order. The input array contains a list of points in which subsequent *x* values are separated by *width* fullwords.

*mdarray* **— specified by user, array of fullword integers**
Move/draw indicators (`1=MOVE, 2=DRAW`). The last entry of this array is disregarded.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **100** | NUMBER OF POINTS < ZERO |
| **555** | MOVE/DRAW INDICATOR IS INVALID |
| **557** | WIDTH PARAMETER < MINIMUM ALLOWED |

### Related Subroutines

| | |
|---|---|
| **GPPLD3** | Polyline Set 3 With Data |
| **GPLT** | Set Linetype |
| **GPLWSC** | Set Linewidth Scale Factor |
| **GPPLCD** | Set Polyline Color Direct |
| **GPPLCI** | Set Polyline Color Index |
| **GPPLET** | Set Polyline End Type |
| **GPPLI** | Set Polyline Index |

### RCP code

201328144 (X'0C000610')

## GPEL2 - Ellipse 2

**GPEL2 (*center*, *refv1*, *refv2*)**

### Purpose

Use **GPEL2** to insert an Ellipse 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Ellipse 2 structure element depending on the current edit mode.

The major and minor axis (*refv1* and *refv2* respectively) parameters define a parallelogram that circumscribes the ellipse and is anchored at the center of the ellipse (*center*). Two sides are parallel to the major axis. The other two sides are parallel to the minor axis. The major and minor axes extend from the center of the ellipse to midpoints of sides of the parallelogram. The ellipse is tangent to the sides of the parallelogram at their midpoints.

Polyline attributes are applied to this primitive.

**GPEL2** is identified as GDP 1007.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*center* **— specified by user, 2 short floating-point numbers (MC)**
Center of ellipse.

*refv1* **— specified by user, 2 short floating-point numbers (MC)**
Major axis of ellipse. This direction vector is anchored at the center of the ellipse.

*refv2* **— specified by user, 2 short floating-point numbers (MC)**
Minor axis of ellipse. This direction vector is anchored at the center of the ellipse.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **107** | REFERENCE VECTORS ARE COLINEAR |

### Related Subroutines

| | |
|---|---|
| **GPLT** | Set Linetype |
| **GPLWSC** | Set Linewidth Scale Factor |
| **GPPLCD** | Set Polyline Color Direct |
| **GPPLCI** | Set Polyline Color Index |
| **GPPLET** | Set Polyline End Type |
| **GPPLI** | Set Polyline Index |

### RCP code

201328148 (X'0C000614')

## GPEL3 - Ellipse 3

**GPEL3 (*center*, *refv1*, *refv2*)**

### Purpose

Use **GPEL3** to insert an Ellipse 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Ellipse 3 structure element depending on the current edit mode.

This is the fully specified, three-dimensional form of the ellipse primitive.

The major and minor axis (*refv1* and *refv2* respectively) parameters define a parallelogram that circumscribes the ellipse and is anchored at the center of the ellipse (*center*). Two sides are parallel to the major axis. The other two sides are parallel to the minor axis. The major and minor axes extend from the center of the ellipse to midpoints of sides of the parallelogram. The ellipse is tangent to the sides of the parallelogram at their midpoints.

Polyline attributes are applied to this primitive.

**GPEL3** is identified as GDP 1008.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*center* **— specified by user, 3 short floating-point numbers (MC)**
Center of ellipse.

*refv1* **— specified by user, 3 short floating-point numbers (MC)**
Major axis of ellipse. This direction vector is anchored at the center of the ellipse.

*refv2* **— specified by user, 3 short floating-point numbers (MC)**
Minor axis of ellipse. This direction vector is anchored at the center of the ellipse.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **107** | REFERENCE VECTORS ARE COLINEAR |

### Related Subroutines

| | |
|---|---|
| **GPLT** | Set Linetype |
| **GPLWSC** | Set Linewidth Scale Factor |
| **GPPLCD** | Set Polyline Color Direct |
| **GPPLCI** | Set Polyline Color Index |
| **GPPLET** | Set Polyline End Type |
| **GPPLI** | Set Polyline Index |

**RCP code**

201328149 (X'0C000615')

## GPELA2 - Elliptical Arc 2

**GPELA2 (*center*, *refv1*, *refv2*, *startv*, *endv*)**

**Purpose**

Use **GPELA2** to insert an Elliptical Arc 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Elliptical Arc 2 structure element depending on the current edit mode.

The graPHIGS API draws the portion of an ellipse (defined by the major and minor axes) specified by the start and end (*startv* and *endv* respectively) parameters.

The major and minor axis (*refv1* and *refv2* respectively) parameters define a parallelogram that circumscribes the ellipse and is anchored at the center of the ellipse. Two sides are parallel to the major axis. The other two sides are parallel to the minor axis. The major and minor axes extend from the center of the ellipse to midpoints of sides of the parallelogram. The ellipse is tangent to the sides of the parallelogram at their midpoints.

The elliptical arc is defined parametrically as the image of an arc of the unit circle in a two-dimensional parameter space. The major and minor axes are images of unit parameter vectors in the *x* and *y* directions respectively. The graPHIGS API draws elliptical drawn from the starting angle to the end angle through increasing parameter values.

Polyline attributes are applied to this primitive.

**GPELA2** is identified as GDP 1009.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*center* **— specified by user, 2 short floating-point numbers (MC)**
Center of arc.

*refv1* **— specified by user, 2 short floating-point numbers (MC)**
Major axis of ellipse. This direction vector is anchored at the center of the ellipse.

*refv2* **— specified by user, 2 short floating-point numbers (MC)**
Minor axis of ellipse. This direction vector is anchored at the center of the ellipse.

*startv* **— specified by user, short floating-point number**
Start value for creation of arc (specified in radians).

*endv* **— specified by user, short floating-point number**
End value for creation of arc (specified in radians).

**Error Codes**

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **107** | REFERENCE VECTORS ARE COLINEAR |

**Related Subroutines**

| | |
|---|---|
| **GPLT** | Set Linetype |
| **GPLWSC** | Set Linewidth Scale Factor |
| **GPPLCD** | Set Polyline Color Direct |
| **GPPLCI** | Set Polyline Color Index |
| **GPPLET** | Set Polyline End Type |
| **GPPLI** | Set Polyline Index |

**RCP code**

201328150 (X'0C000616')

## GPELA3 - Elliptical Arc 3

**GPELA3 (*center*, *refv1*, *refv1*, *startv*, *endv*)**

**Purpose**

Use **GPELA3** to insert an Elliptical Arc 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Elliptical Arc 3 structure element depending on the current edit mode. This is the fully specified, three-dimensional form of the elliptical arc primitive.

The major and minor axis (*refv1* and *refv2* respectively) parameters define a parallelogram that circumscribes the ellipse and is anchored at the center of the ellipse (*center*). Two sides are parallel to the major axis. The other two sides are parallel to the minor axis. The major and minor axes extend from the center of the ellipse to midpoints of sides of the parallelogram. The ellipse is tangent to the sides of the parallelogram at their midpoints.

The elliptical arc is defined parametrically as the image of an arc of the unit circle in a two-dimensional parameter space. The major and minor axes are images of unit parameter vectors in the *x* and *y* directions respectively. The graPHIGS API draws the elliptical from the starting angle to the end angle through increasing parameter values.

Polyline attributes are applied to this primitive.

**GPELA3** is identified as GDP 1010.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*center* — **specified by user, 3 short floating-point numbers (MC)**
Center of arc.

*refv1* — **specified by user, 3 short floating-point numbers (MC)**
Major axis of ellipse. This direction vector is anchored at the center of the ellipse.

*refv2* — **specified by user, 3 short floating-point numbers (MC)**
Minor axis of ellipse. This direction vector is anchored at the center of the ellipse.

*startv* — **specified by user, short floating-point number**
Start value for creation of arc (specified in radians).

*endv* — **specified by user, short floating-point number**
End value for creation of arc (specified in radians).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **107** | REFERENCE VECTORS ARE COLINEAR |

### Related Subroutines

| | |
|---|---|
| **GPLT** | Set Linetype |
| **GPLWSC** | Set Linewidth Scale Factor |
| **GPPLCD** | Set Polyline Color Direct |
| **GPPLCI** | Set Polyline Color Index |
| **GPPLET** | Set Polyline End Type |
| **GPPLI** | Set Polyline Index |

### RCP code

201328151 (X'0C000617')

## GPLG2 - Line Grid 2

**GPLG2 (*point*, *refv1*, *refv2*, *imin*, *imax*, *jmin*, *jmax*)**

### Purpose

Use **GPLG2** to insert a Line Grid 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Line Grid 2 structure element depending on the current edit mode.

When encountered during structure traversal, this element generates a grid of lines in the z=0 plane of modelling coordinates. Each grid line is defined by the following parametric equations and is subject to all transformations and clipping.

The endpoints of the generated line segments are defined by the following equations. A line segment is generated for each pair of endpoints ($\mathbf{P}_1$, $\mathbf{P}_2$) for each value of i and j.

Line segments parallel to $\mathbf{V}_1$
$\mathbf{P}_1(j) = \mathbf{P}_0 + i_{min}\mathbf{V}_1 + j\ \mathbf{V}_2$
$\mathbf{P}_2(j) = \mathbf{P}_0 + i_{max}\ \mathbf{V}_1 + j\ \mathbf{V}_2$
where $j_{min} <= j <= j_{max}$

Line segments parallel to $\mathbf{V}_2$
$\mathbf{P}_1(i) = \mathbf{P}_0 + i\ \mathbf{V}_1 + j_{min}\ \mathbf{V}_2$
$\mathbf{P}_2(i) = \mathbf{P}_0 + i\ \mathbf{V}_1 + j_{max}\mathbf{V}_2$
where $i_{min} <= i <= i_{max}$

Polyline attributes are applied to this primitive.

**GPLG2** is identified as GDP 1023.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine (page GPQGD - Inquire List of Generalized Drawing Primitives) to determine the GDPs supported by an opened workstation. See also the workstation description in The graPHIGS Programming Interface: Technical Reference.

**Parameters**

*point* **— specified by user, 2 short floating-point numbers (MC)**
origin of grid ($\mathbf{P}_0$)

*refv1* **— specified by user, 2 short floating-point numbers (MC)**
*i* reference vector ($\mathbf{V}_1$). This vector defines the direction of grid lines of constant *j* parameter and the spacing between grid lines of constant *i* parameter.

*refv2* **— specified by user, 2 short floating-point number (MC)**
*j* reference vector ($\mathbf{V}_2$). This vector defines the direction of grid lines of constant *i* parameter and the spacing between grid lines of constant *j* parameter.

*imin* **— specified by user, fullword integer**
Minimum *i* parameter ($i_{min}$). This value defines the smallest value of *i* that is part of the grid (<= *imax*).

*imax* **— specified by user, fullword integer**
Maximum *i* parameter. This parameter defines the largest value of *i* that is part of the grid (>= *imin*).

*jmin* **— specified by user, fullword integer**
Minimum *j* parameter. This parameter defines the smallest value of *j* that is part of the grid (<= *jmax*).

*jmax* **— specified by user, fullword integer**
Maximum *j* parameter. This parameter defines the largest value of *j* that is part of the grid (>= *jmin*)

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**107** REFERENCE VECTORS ARE COLINEAR

**340** MINIMUM GRID LIMIT > MAXIMUM

**Related Subroutines**

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLET**
Set Polyline End Type

**GPPLI** Set Polyline Index

**RCP code**

201344260 (X'0C004504')

# GPLG3 - Line Grid 3

**GPLG3 (*point*, *refv1*, *refv2*, *imin*, *imax*, *jmin*, *jmax*)**

### Purpose

Use **GPLG3** to insert a Line Grid 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Line Grid 3 structure element depending on the current edit mode.

When encountered during structure traversal, this element generates a grid of lines in the plane defined by *point*, *refv1*, *refv2*. Each grid line is defined by the following parametric equations and is subject to all transformations and clipping.

The endpoints of the generated line segments are defined by the following equations. A line segment is generated for each pair of endpoints ($P_1$, $P_2$) for each value of *i* and *j*.

Line segments parallel to $\mathbf{V}_1$
$\mathbf{P}_1(j) = \mathbf{P}_0 + i_{min}\mathbf{V}_1 + j\ \mathbf{V}_2$
$\mathbf{P}_2(j) = \mathbf{P}_0 + i_{max}\ \mathbf{V}_1 + j\ \mathbf{V}_2$
where $j_{min} <= j <= j_{max}$

Line segments parallel to $\mathbf{V}_2$
$\mathbf{P}_1(i) = \mathbf{P}_0 + i\ \mathbf{V}_1 + j_{min}\ \mathbf{V}_2$
$\mathbf{P}_2(i) = \mathbf{P}_0 + i\ \mathbf{V}_1 + j_{max}\mathbf{V}_2$
where $i_{min} <= i <= i_{max}$

Polyline attributes are applied to this primitive.

**GPLG3** is identified as GDP 1022.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine(page GPQGD - Inquire List of Generalized Drawing Primitives) to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference.*

### Parameters

*point* — **specified by user, 3 short floating-point numbers (MC)**
origin of grid ($\mathbf{P}_0$).

*refv1* — **specified by user, 3 short floating-point numbers (MC)**
*i* reference vector ($\mathbf{V}_1$). This vector defines the direction of grid lines of constant *j* parameter and the spacing between grid lines of constant *i* parameter.

*refv2* — **specified by user, 3 short floating-point numbers (MC)**
*j* reference vector ($\mathbf{V}_2$). This vector defines the direction of grid lines of constant *i* parameter and the spacing between grid lines of constant *j* parameter.

*imin* — **specified by user, fullword integer**
Minimum *i* parameter ($i_{min}$). This value defines the smallest value of *i* that is part of the grid (<=*imax*).

*imax* — **specified by user, fullword integer**
Maximum *i* parameter ($i_{max}$). This parameter defines the largest value of *i* that is part of the grid (>=*imin*).

*jmin* **— specified by user, fullword integer**
Minimum *j* parameter ($j_{min}$). This parameter defines the smallest value of *j* that is part of the grid (<=*jmax*).

*jmax* **— specified by user, fullword integer**
Maximum *j* parameter ($j_{max}$). This parameter defines the largest value of *j* that is part of the grid (>=*jmin*).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**107** REFERENCE VECTORS ARE COLINEAR

**340** MINIMUM GRID LIMIT > MAXIMUM

**Related Subroutines**

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLET**
Set Polyline End Type

**GPPLI** Set Polyline Index

**RCP code**

201344259 (X'0C004503')

## GPMG2 - Marker Grid 2

**GPMG2 (*point*, *refv1*, *refv2*, *imin*, *imax*, *jmin*, *jmax*)**

**Purpose**

Use **GPMG2** to insert a Marker Grid 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Marker Grid 2 structure element depending on the current edit mode.

When encountered during structure traversal, this element generates a grid of markers in the *z*=0 plane of modelling coordinates. Each grid location is defined by the following parametric equation and is subject to all transformations and clipping. Each position on the grid is defined by the following parametric vector equation:

$\mathbf{P}(i,j) = \mathbf{P}_0 + i\boldsymbol{V}_1 + j\boldsymbol{V}_{\#EMPTY>2}$

$i_{min} <= i <= i_{max}$

$j_{min} <= j <= j_{max}$

Polymarker attributes are applied to this primitive.

**GPMG2** is identified as GDP 1021.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*point* **— specified by user, 2 short floating-point numbers (MC)**
origin of grid ($\mathbf{P}_0$).

*refv1* **— specified by user, 2 short floating-point numbers (MC)**
*i* reference vector ($\mathbf{V}_1$). This vector defines the relative location of the next grid location that is obtained by incrementing the *i* parameter.

*refv2* **— specified by user, 2 short floating-point numbers (MC)**
*j* reference vector ($\mathbf{V}_2$). This vector defines the relative location of the next grid location that is obtained by incrementing the *j* parameter.

*imin* **— specified by user, fullword integer**
Minimum *i* parameter limit ($i_{min}$). This parameter specifies the smallest value of the *i* parameter at which markers are generated (<= imax).

*imax* **— specified by user, fullword integer**
Maximum *i* parameter limit ($i_{max}$). This parameter specifies the largest value of the *i* parameter at which markers are generated (>=imin).

*jmin* **— specified by user, fullword integer**
Minimum *j* parameter limit ($j_{min}$). This parameter specifies the smallest value of the *j* parameter at which markers are generated (<=jmax).

*jmax* **— specified by user, fullword integer**
Maximum *j* parameter limit ($j_{max}$). This parameter specifies the largest value of the *j* parameter at which markers are generated (>=jmin).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**107** REFERENCE VECTORS ARE COLINEAR

**340** MINIMUM GRID LIMIT > MAXIMUM

### Related Subroutines

**GPMSSC**
Set Marker Size Scale Factor

**GPMT** Set Marker Type

**GPPMCD**
Set Polymarker Color Direct

**GPPMCI**
Set Polymarker Color Index

**GPPMI**
Set Polymarker Index

### RCP code

201344258 (X'0C004502')

# GPMG3 - Marker Grid 3

**GPMG3 (*point*, *refv1*, *refv2*, *imin*, *imax*, *jmin*, *jmax*)**

### Purpose

Use **GPMG3** to insert a Marker Grid 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Marker Grid 3 structure element depending on the current edit mode.

When encountered during structure traversal, this element generates a grid of markers in the plane defined by *point*, *refv1*, *refv2*. Each grid location is defined by the following parametric equation and is subject to all transformations and clipping.

Each position on the grid is defined by the following parametric vector equation:

$\mathbf{P}(i,j) = \mathbf{P}_0 + i\mathbf{V}_1 + j\mathbf{V}_2$

$i_{min} <= i <= i_{max}$

$i_{min} <= j <= j_{max}$

Polymarker attributes are applied to this primitive.

**GPMG3** is identified as GDP 1020.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*point* — **specified by user, 3 short floating-point numbers (MC)**
origin of grid ($\mathbf{P}_0$)

*refv1* — **specified by user, 3 short floating-point numbers (MC)**
*i* reference vector ($\mathbf{V}_1$). This vector defines the relative location of the next grid location that is obtained by incrementing the *i* parameter.

*refv2* — **specified by user, 3 short floating-point numbers (MC)**
*j* reference vector ($\mathbf{V}_2$). This vector defines the relative location of the next grid location that is obtained by incrementing the *j* parameter.

*imin* — **specified by user, fullword integer**
Minimum *i* parameter limit ($i_{min}$). This parameter specifies the smallest value of the *i* parameter at which markers are generated (<=imax).

*imax* — **specified by user, fullword integer**
Maximum *i* parameter limit ($i_{max}$). This parameter specifies the largest value of the *i* parameter at which markers are generated (>=imin).

*jmin* — **specified by user, fullword integer**
Minimum *j* parameter limit ($j_{min}$). This parameter specifies the smallest value of the *j* parameter at which markers are generated (<=jmax).

*jmax* — **specified by user, fullword integer**
Maximum *j* parameter limit ($j_{max}$). This parameter specifies the largest value of the *j* parameter at which markers are generated (>=jmin).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**107** REFERENCE VECTORS ARE COLINEAR

**340** MINIMUM GRID LIMIT > MAXIMUM

**Related Subroutines**

**GPMSSC**
Set Marker Size Scale Factor

**GPMT** Set Marker Type

**GPPMCD**
Set Polymarker Color Direct

**GPPMCI**
Set Polymarker Color Index

**GPPMI**
Set Polymarker Index

**RCP code**

201344257 (X'0C004501')

## GPNBC2 - Non-Uniform B-Spline Curve 2

**GPNBC2 (*order*, *npoint*, *knot*, *tflags*, *tdata*, *cflags*, *cwidth*, *ctlpts*, *tmin*, *tmax*)**

**Purpose**

Use **GPNBC2** to insert a Non-Uniform B-Spline Curve 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Non-Uniform B-Spline Curve 2 structure element depending on the current edit mode.

During structure traversal, the graPHIGS API uses the specified coefficients to generates a a non-uniform B-spline curve of the specified order. The graPHIGS API draws the curve only over the parameter range specified.

Polyline attributes are applied to this primitive.

**GPNBC2** is identified as GDP 1034.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*order* **— specified by user, fullword integer**
Curve order (>=2).

*npoint* **— specified by user, fullword integer**
Number of control points (>= *order*).

*knot* **— specified by user, array of short floating-point numbers**
Knot Vector. This array must contain *order*+ *npoint* floating-point numbers representing a non-decreasing sequence of knot values in the parametric space.

*tflags* **— specified by user, fullword integer**
Tessellation quality value flags. This parameter shows whether or not the tessellation quality value is specified in the *tdata* parameter.

| Value | Meaning |
|---|---|
| **0** | Not specified. |
| **1** | Specified. |

*tdata* **— specified by user, array of short floating-point numbers**
Array of tessellation quality values. When the *tflags* parameter is set to a value of one, this array must contain *npoint*- *order*+1 tessellation quality values. These value are used in conjunction with Curve Approximation Criteria method 8 to control the number of line segments that are generated for each curve span. The number of line segments generated for this span is approximately the product of this value and the Curve Approximation Criteria control value in the traversal state list.

*cflags* **— specified by user, fullword integer**
Control point optional data flags. This parameter shows what data is specified for each control point. The value specified should be the sum of the following values based on the fields that are specified in the *ctlpts* parameter.

| Value | Meaning |
|---|---|
| **0** | Control point coordinates. |
| **1** | Weight for each control point. This produces the rational form of the Non-Uniform B-Spline Curve. |

*cwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *ctlpts* parameter.

*ctlpts* **— specified by user, array of short floating-point numbers (MC)**
Control point data. This array must contain *npoint* entries. The minimum required width and content of each entry depends on the value of parameter *cflags*. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Control point coordinates — 2 short floating-point numbers**
This field must always be present. Therefore, the *cwidth* parameter must be at least two.

**Weight — 1 short floating-point number**
Each weight *W* must be greater than zero when specified. If this field is specified, the *cwidth* parameter must be at least three.

> **Note:** When *W* is specified, the control points are not in homogeneous form (i.e., *XW*, *YW*, *W*). They are specified after division by *W* or (*X*, *Y*, *W*)

*tmin* **— specified by user, short floating-point number**
The minimum parameter value at which the curve is evaluated. The curve is evaluated at parameter values between *tmin* and *tmax* inclusive. This value must be greater than or equal to the value of knot *order*.

*tmax* **— specified by user, short floating-point number**
The maximum parameter value at which the curve is evaluated. The curve is evaluated at parameter values between *tmin* and *tmax* inclusive. This value must be less than or equal to the value of knot *npoint*+1.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**341** ORDER OF BASIS FUNCTION < TWO

**342** ORDER IS GREATER THAN NUMBER OF CONTROL POINTS

**343** KNOT VECTOR IS INVALID

**345** WEIGHT IN CONTROL POINT IS <= ZERO

**347** PARAMETER LIMITS ARE OUTSIDE VALID PARAMETER RANGE

**348** MINIMUM PARAMETER LIMIT > MAXIMUM

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**362** TESSELLATION CONTROL VALUE IS INVALID

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**Related Subroutines**

**GPCAC**
Set Curve Approximation Criteria

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLET**
Set Polyline End Type

**GPPLI** Set Polyline Index

**GPQCDF**
Inquire Curve Display Facilities

**RCP code**

201341954 (X'0C003C02')

## GPNBC3 - Non-Uniform B-Spline Curve 3

**GPNBC3 (*order*, *npoint*, *knot*, *tflags*, *tdata*, *cflags*, *cwidth*, *ctlpts*, *tmin*, *tmax*)**

**Purpose**

Use **GPNBC3** to insert a Non-Uniform B-Spline Curve 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Non-Uniform B-Spline Curve 3 structure element depending on the current edit mode.

During structure traversal, the graPHIGS API uses the specified coefficients to generate a a non-uniform B-spline curve of the specified order. The graPHIGS API draws the curve only over the parameter range specified.

Polyline attributes are applied to this primitive.

**GPNBC3** is identified as GDP 1033.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*order* **— specified by user, fullword integer**
Curve order (>=2).

*npoint* **— specified by user, fullword integer**
Number of control points (>= *order*).

*knot* **— specified by user, array of short floating-point numbers**
Knot Vector. This array must contain *order*+*npoint* floating-point numbers representing a non-decreasing sequence of knot values in the parametric space.

*tflags* **— specified by user, fullword integer**
Tessellation quality value flags. This parameter shows whether the tessellation quality value is specified or not in the *tdata* parameter.

| Value | Meaning |
|---|---|
| **0** | Not specified. |
| **1** | Specified. |

*tdata* **— specified by user, array of short floating-point numbers**
Array of tessellation quality values. When the *tflags* parameter is 1, this array must contain *npoint*-*order*+1 tessellation quality values. These values are used in conjunction with Curve Approximation Criteria method 8 to control the number of line segments that are generated for each curve span. The number of line segments generated for this span is approximately the product of this value and the Curve Approximation Criteria control value in the traversal state list.

*cflags* **— specified by user, fullword integer**
Control point optional data flags. This parameter shows what data is specified for each control point. The value specified should be the sum of the following values based on the fields that are specified in the *ctlpts* parameter:

| Value | Meaning |
|---|---|
| **0** | Control point coordinates. |
| **1** | Weight for each control point. This produces the rational form of the Non-Uniform B-Spline Curve. |

*cwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *ctlpts* parameter.

*ctlpts* **— specified by user, array of short floating-point numbers (MC)**
Control point data. This array must contain *npoint* entries. The minimum required width and content of each entry depends on the value of parameters *cflags*. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Control point coordinates — 3 short floating-point numbers**
This field must always present. Therefore, the *cwidth* parameter must be at least three.

**Weight — 1 short floating-point number**
Each weight *W* must be greater than zero when specified. If this field is specified, the *cwidth* parameter must be at least four.

**Note:** When *W* is specified, the control points are not in homogeneous form (i.e.,*XW*, *YW*, *ZW*, *W*). They are specified after division by *W* or (*X*, *Y*, *Z*, *W*).

*tmin* **— specified by user, short floating-point number**
The minimum parameter value at which the curve is evaluated. The curve is evaluated at parameter values between *tmin* and *tmax* inclusive. This value must be greater than or equal to the value of knot *order*.

*tmax* **— specified by user, short floating-point number**
The maximum parameter value at which the curve is evaluated. The curve is evaluated at parameter values between *tmin* and *tmax* inclusive. This value must be less than or equal to the value of knot *npoint*+1.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**341** ORDER OF BASIS FUNCTION < TWO

**342** ORDER IS GREATER THAN NUMBER OF CONTROL POINTS

**343** KNOT VECTOR IS INVALID

**345** WEIGHT IN CONTROL POINT IS <= ZERO

**347** PARAMETER LIMITS ARE OUTSIDE VALID PARAMETER RANGE

**348** MINIMUM PARAMETER LIMIT > MAXIMUM

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**362** TESSELLATION CONTROL VALUE IS INVALID

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**Related Subroutines**

**GPCAC**
Set Curve Approximation Criteria

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLET**
Set Polyline End Type

**GPPLI** Set Polyline Index

**GPQCDF**
Inquire Curve Display Facilities

**RCP code**

201341953 (X'0C003C01')

## GPNBS - Non-Uniform B-Spline Surface

**GPNBS (*uorder*, *vorder*, *unum*, *vnum*, *uknots*, *vknots*, *tflag*, *utdata*, *vtdata*, *cflags*, *cwidth*, *ctlpts*, *umin*, *umax*, *vmin*, *vmax*)**

### Purpose

Use **GPNBS** to insert a Non-Uniform B-Spline Surface structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Non-Uniform B-Spline Surface structure element depending on the current edit mode.

During structure traversal, the graPHIGS API uses the specified control points to generate a non-uniform parametric surface of the specified *uorder* and *vorder*. The graPHIGS API renders only the portion of the surface within the parameter limits.

If the specified workstation does not support the requested orders for the basis functions of the surface, then this primitive does not generate any output.

The lines of constant parameter at the parameter limits of the surface are rendered as edges.

Polygon and surface attributes are applied to this primitive.

**GPNBS** is identified as GDP 1035.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*uorder* **— specified by user, fullword integer**
Order of the basis functions for the *u* parameter (>=2).

*vorder* **— specified by user, fullword integer**
Order of the basis functions for the *v* parameter (>=2).

*unum* **— specified by user, fullword integer**
Number of surface control points for the *u* direction (>=*uorder*).

*vnum* **— specified by user, fullword integer**
Number of surface control points for the *v* direction (>=*vorder*).

*uknots* **— specified by user, array of short floating-point numbers**
Knot values for the *u* parameter. The length of this array must be *uorder*+ *unum*. This parameter must be a non-decreasing knot value sequence.

*vknots* **— specified by user, array of short floating-point numbers**
Knot values for the *v* parameter. The length of this array must be *vorder*+ *vnum*. This parameter must be a non-decreasing knot value sequence.

*tflag* **— specified by user, fullword integer**
Surface tessellation quality value flag. This parameter shows whether the tessellation quality values are specified or not.

**0** Not specified.

**1** Specified.

*utdata* **— specified by user, array of short floating-point numbers**
Tessellation quality values for the *u* direction. When the *tflag* parameter is set to a value of one, this parameter must contain *unum- uorder*+1 quality values. These value are used in conjunction with Surface Approximation Criteria method 8 to control the number of sub-divisions made in the *u* direction. The number of sub-divisions that are performed for a patch is approximately the product of this value and the Surface Approximation Criteria control value (*u*) in the traversal state list.

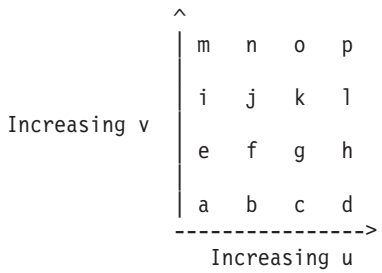
*vtdata* **— specified by user, array of short floating-point numbers**
Tessellation quality values for the *v* direction. When the *tflag* parameter is set to a value of one, this parameter must contain *vnum- vorder*+1 quality values. These value are used in conjunction with Surface Approximation Criteria method 8 to control the number of sub-divisions made in the *v* direction. The number of sub-divisions that are performed for a patch is approximately the product of this value and the Surface Approximation Criteria control value (*v*) in the traversal state list.

*cflags* **— specified by user, fullword integer**
Control point optional data flags. This parameter shows what data is specified for each control point. The value specified should be the sum of the following values based on the fields that are specified in the *ctlpts* parameter.

| Value | Meaning |
|---|---|
| **0** | Control point coordinates. |
| **1** | Weights are specified with each control point. This produces the rational form of the Non-Uniform B-Spline Surface. |

*cwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of control points array *ctlpts*

*ctlpts* **— specified by user, array of short floating-point numbers.**
Grid of control points. The control points are stored by row where a row is considered to be the direction associated with the *u* parameter. For example, the set of control points




would be stored in the order a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, and p. The *cwidth* parameter must be at least three. If *cflags* specifies that weights are included with each control point, then the *cwidth* parameter must be at least four. Each weight *W* must be greater than zero when specified.

> **Note:** When *W* is specified, the control points are not in homogeneous form (i.e.,*XW*, *YW*, *ZW*, *W*). They are specified after division by *W* or (*X*, *Y*, *Z*, *W*).

*umin* **— specified by user, short floating-point number**
The minimum parameter value in the *u* dimension at which the surface is evaluated. This value must be greater than or equal to the value of knot *uorder* in parameter *uknots*.

*umax* **— specified by user, short floating-point number**
The maximum parameter value in the *u* dimension at which the surface is evaluated. This value must be less than or equal to the value of knot *unum*+1 in parameter *uknots*

*vmin* **— specified by user, short floating-point number**
The minimum parameter value in the *v* dimension at which the surface is evaluated. This value must be greater than or equal to the value of knot *vorder* in parameter *vknots*

*vmax* **— specified by user, short floating-point number**
The maximum parameter value in the *v* dimension at which the surface is evaluated. This value must be less than or equal to the value of knot *vnum*+1 in parameter *vknots*

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**341** ORDER OF BASIS FUNCTION < TWO

**342** ORDER IS GREATER THAN NUMBER OF CONTROL POINTS

**343** KNOT VECTOR IS INVALID

**345** WEIGHT IN CONTROL POINT IS <= ZERO

**347** PARAMETER LIMITS ARE OUTSIDE VALID PARAMETER RANGE

**348** MINIMUM PARAMETER LIMIT > MAXIMUM

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**362** TESSELLATION CONTROL VALUE IS INVALID

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**Related Subroutines**

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPLMO**
Set Lighting Calculation Mode

**GPLSS**
Set Light Source State

**GPPGC**
Set Polygon Culling

**GPQSDF**
Inquire Surface Display Facilities

**GPSAC**
Set Surface Approximation Criteria

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCAC**
Set Trimming Curve Approximation Criteria

### RCP code

201342721 (X'0C003F01')

## GPPGD2 - Polygon 2 With Data

**GPPGD2 (*pflags*, *pdata*, *saflags*, *sawidth*, *sadata*, *vxflags*, *vxwidth*, *vxdata*)**

### Purpose

Use **GPPGD2** to insert a Polygon 2 With Data structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polygon 2 With Data structure element depending on the current edit mode.

When encountered during structure traversal, this element generates a polygon with the specified number of subareas. You may specify optional data to further control the rendering of this primitive. The optional data consists of:

- Convexity flag.

  The convexity flag indicates that the application determined the convexity of the polygon primitive. Therefore, the system rendering code does not have to determine the convexity every time the primitive is rendered. To determine the convexity of a set of polygons, the graPHIGS API on the RS/6000® contains a sample program under the directory: **/usr/lpp/graPHIGS/samples/convexcheck**

- Vertex colors.

  When rendering this primitive, if the primitive is not to be highlighted, then the graPHIGS API uses the specified vertex color. The colors are used in the lighting process to produce more realistic effects.
- Boundary flags.

  When rendering this primitive, if the current edge flag attribute is set to 2=ON, then the graPHIGS API renders as part of the edge, only the parts of the polygon defined by the boundary flags to be part of the edge. If there are no boundary flags specified, then the graPHIGS API treats it the same as a polygon (i.e., all boundaries are rendered as edges)
- Transparency coefficients.

  You can specify a transparency coefficient per vertex. The graPHIGS API uses these values when producing transparency effects for the rendered primitive.
- Vertex morphing vectors.

  You can supply vertex morphing vectors per vertex. The graPHIGS API combines these vectors with the vertices and vertex morphing scale factors (**GPVMF**)to create new vertex coordinate values for the rendered primitive.
- Data mapping data.

  You can specify data mapping data per vertex. The graPHIGS API uses these values to determine the colors of the rendered primitive.
- Data morphing vectors.

  You can specify data morphing vectors per vertex. The graPHIGS API combines these vectors with the data morphing scale factors (**GPDMF**) and (**GPBDMF**) and the vertex data mapping values to create new data mapping data values for the rendered primitive.

See *The graPHIGS Programming Interface: Understanding Concepts* for a more complete explanation of how the graPHIGS API uses the various optional data values.

> **Note:** This note applies ONLY to applications which will be run on the High Performance 3D Color Graphics Processor (8 or 24 bit). Use of any optional data other than the convexity flag, vertex colors, and boundary flags may cause unpredictable results (including locking the display) on this graphics processor. If only the High Performance 3D Color Graphics Processor is used, you should include only the supported optional data values. If your application must support multiple graphics processors INCLUDING this particular processor, the Inquire Workstation Description (**GPQWDT**) subroutine must be used to determine the functions that each workstation supports. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference* for the High Performance 3D Color Graphics Processor.

The normal of a Polygon 2 With Data is (0,0,1). All points specified are placed in the *x-y* plane (z=0). Polygon attributes are applied to this primitive.

**GPPGD2** is identified as GDP 1017.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*pflags* **— specified by user, fullword integer**

Shows what optional data is specified for the primitive. The value specified should be the sum of the following values based on the fields that are included in the *pdata* parameter.

| Value | Corresponding Field |
|---|---|
| **0** | Number of subareas. |

**2** Convexity flag is specified.

**4** Count of vertex morphing vectors is specified.

**8** Count of data mapping data is specified.

**16** Count of data morphing vectors is specified.

*pdata* **— specified by user, array of primitive data**
Contains specific information about the entire primitive. The presence of optional fields is determined by the value of the *pflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Number of Subareas — fullword integer (>=0)**
Number of contours in the polygon definition. This field is required.

**Convexity flag — fullword integer**
This data indicates that the application determined the convexity of the polygon (`0=CONCAVE, 1=CONVEX`). This field is optional.

**Vertex Morphing Vector Count — fullword integer (>=0)**
The number of vertex morphing vectors specified at each vertex. The number of fullwords of vertex morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

**Data Mapping Data Count — fullword integer (>=0)**
The number of data mapping values specified at each vertex. The number of data mapping values added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

**Data Morphing Vector Count — fullword integer (>=0)**
The number of data morphing vectors specified at each vertex. The number of fullwords of data morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

*saflags* **— specified by user, fullword integer**
Shows what optional data is specified for each subarea. The value specified should be the sum of the following values based on the fields that are included in the *sadata* parameter.

**Value** **Corresponding Field**

**0** Number of vertices. (There is no optional data currently defined. This field must be set to zero)

*sawidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *sadata* parameter array (>=1)

*sadata* **— specified by user, array of per subarea data**
Contains specific information about each subarea in the primitive. The length of this array is defined by the contents of the `number of subareas` field in the *pdata* parameter. The presence of optional fields is determined by the value of the *saflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Number of Vertices — fullword integer (>=0)**
Number of vertices in each subarea. If the number is less than three, the result is workstation dependent. This field is required for each subarea.

*vxflags* **— specified by user, fullword integer**
Shows what optional data is specified for each vertex. The value specified should be the sum of the following values based on the fields that are specified in the *vxdata* parameter.

| Value | Meaning |
|---|---|
| **0** | Vertex coordinates are specified. |
| **2** | Vertex colors are specified. |
| **4** | Boundary flags are specified. |
| **8** | Transparency coefficient is specified. |
| **16** | Vertex morphing vectors are specified. |
| **32** | Data mapping data is specified. |
| **64** | Data morphing vectors are specified (valid only if data mapping data is specified also). |

*vxwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *vxdata* parameter array (>=2)

*vxdata* **— specified by user, array of vertex data**
Contains specific information about each vertex in the primitive. The length of this array is equal to the sum of the `number of vertices` fields in the *sadata* parameter. The presence of optional fields is determined by the value of the *vxflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Coordinates — 2 short floating-point numbers (MC)**
*x* and *y* coordinates of a vertex. This field is required.

**Color — 3 short floating-point numbers**
The three components of a color in the current color model as contained in the graPHIGS API state list. This field is optional.

**Boundary Flags — fullword integer**
Specifies whether the corresponding boundary is to be treated as an edge of the polygon (`1=NOT_AN_EDGE, 2=IS_AN_EDGE`). These flags allow control over which parts of the polygon's boundary are to be treated as edges. The edge attributes are only applied to boundary segments that have a boundary flag set to a value of `2=IS_AN_EDGE`.

Each entry of this array specifies whether the graPHIGS API draws the boundary from the corresponding vertex to the following vertex. The last entry for each subarea corresponds to the boundary from the last vertex to the first. This field is optional.

**Transparency Coefficient — short floating-point number (0.0 <= transparency coefficient <=1.0)**
The transparency coefficient value used when performing transparency processing. A value of 0.0 is fully opaque; a value of 1.0 is fully transparent. This field is optional.

**Vertex Morphing Vectors — array of short floating-point numbers**
The vertex morphing vectors $dx_1$, $dy_{\#EMPTY>1}$, $dx_2$, $dy_2$, ..., $dx_n$, $dy_n$. The number, *n*, of vectors in this array is specified in the *pdata* parameter as the vertex morphing vector count. The array must be the same length for every vertex. This field is optional.

**Data Mapping Data — array of short floating-point numbers**
The data mapping data values $x_1$, $x_{\#EMPTY>2}$, $x_3$, ..., $x_{\#EMPTY>n}$. The number, *n*, of values in this array is specified in the *pdata* parameter as the data mapping data count. The array must be the same length for every vertex. This field is optional.

**Data Morphing Vectors — array of short floating-point numbers**
The data morphing vectors $d_{11}$>, $d_{\#EMPTY>12}$>, $d_{13}$>, ..., $d_{\#EMPTY>1n}$>, $d_{21}$>, $d_{\#EMPTY>22}$>, $d_{23}$>, ..., $d_{\#EMPTY>2n}$>, ..., $d_{m1}$, $d_{\#EMPTY>m2}$, $d_{m3}$, ..., $d_{\#EMPTY>mn}$. The number, *n*, is specified in the

*pdata* parameter as the data mapping data count, and the number, *m* is specified in the *pdata* parameter as the data morphing vector count. The array must be the same length for every vertex. This field is optional.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**115** TRANSPARENT COEFFICIENT IS INVALID

**198** NUMBER OF SUBAREAS < ZERO

**199** POLYGON SUBAREA HAS < ZERO POINTS

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**352** BOUNDARY FLAG IS INVALID

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**636** FULLWORDS OF VERTEX DATA EXCEEDS MAXIMUM OF 255

**Related Subroutines**

**GPBDMF**
Set Back Data Morphing Factors

**GPBDMI**
Set Back Data Mapping Index

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBISM**
Set Back Interior Shading Method

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPBTCO**
Set Back Transparency Coefficient

**GPDMF**
Set Data Morphing Factors

**GPDMI**
Set Data Mapping Index

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPISM**
Set Interior Shading Method

**GPPGC**
Set Polygon Culling

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCO**
Set Transparency Coefficient

**GPVMF**
Set Vertex Morphing Factors

### RCP code

201342210 (X'0C003D02')

## GPPGD3 - Polygon 3 With Data

**GPPGD3 (*pflags*, *pdata*, *saflags*, *sawidth*, *sadata*, *vxflags*, *vxwidth*, *vxdata*)**

### Purpose

Use **GPPGD3** to insert a Polygon 3 With Data structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Polygon 3 With Data structure element depending on the current edit mode.

When encountered during structure traversal, this element generates a polygon with the specified number of subareas. You may specify optional data to further control the rendering of this primitive. The optional data consists of:

- Convexity flag.

The convexity flag indicates that the application determined the convexity of the polygon primitive. Therefore, the system rendering code does not have to determine the convexity every time the primitive is rendered. To determine the convexity of a set of polygons, the graPHIGS API on the RS/6000® contains a sample program under the directory: **/usr/lpp/graPHIGS/samples/convexcheck**

- Normals.

  You can specify a geometric normal of the polygon and/or a normal for each vertex of the polygon. The normals are used in the lighting process to produce more realistic effects.

- Vertex colors.

  When rendering this primitive, if the primitive is not to be highlighted, then the graPHIGS API uses the specified vertex color. The colors are used in the lighting process to produce more realistic effects.

- Boundary flags.

  When rendering this primitive, if the current edge flag attribute is set to `2=ON`, then the graPHIGS API renders as part of the edge, only the parts of the polygon defined by the boundary flags to be part of the edge. If there are no boundary flags specified, then it is treated the same as a polygon (i.e. all boundaries are rendered as edges)

- Transparency Coefficients.

  You can specify a transparency coefficient per vertex. The graPHIGS API uses these values when producing transparency effects for the rendered primitive.

- Vertex morphing vectors.

  You can supply vertex morphing vectors per vertex. The graPHIGS API combines these vectors with the vertices and vertex morphing scale factors (**GPVMF**) to create new vertex coordinate values for the rendered primitive.

- Data mapping data.

  You can specify data mapping data per vertex. The graPHIGS API uses these values to determine the colors of the rendered primitive.

- Data morphing vectors.

  You can specify data morphing vectors per vertex. The graPHIGS API combines these vectors with the data morphing scale factors (**GPDMF**) and (**GPBDMF**) and the vertex data mapping values to create new data mapping data values for the rendered primitive.

See *The graPHIGS Programming Interface: Understanding Concepts* for a more complete explanation of how the graPHIGS API uses the various optional data values.

> **Note:** This note applies ONLY to applications which will be run on the High Performance 3D Color Graphics Processor (8 or 24 bit). Use of any optional data other than the convexity flag, polygon normal, vertex normals, vertex colors, and boundary flags may cause unpredictable results (including locking the display) on this graphics processor. If only the High Performance 3D Color Graphics Processor is used, you should include only the supported optional data values. If your application must support multiple graphics processors INCLUDING this particular processor, the Inquire Workstation Description (**GPQWDT**) subroutine must be used to determine the functions that each workstation supports. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference* for the High Performance 3D Color Graphics Processor.

All points specified must lie in the same plane, the graPHIGS API makes no check to verify this. The system behavior is undefined in this case.

Polygon attributes are applied to this primitive.

**GPPGD3** is identified as GDP 1016.

**Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*pflags* **— specified by user, fullword integer**
Shows what optional data is specified for the primitive. The value specified should be the sum of the following values based on the fields that are included in the *pdata* parameter.

| Value | Corresponding Field |
|---|---|
| **0** | Number of subareas |
| **1** | Geometric normal is specified. |
| **2** | Convexity flag is specified. |
| **4** | Count of vertex morphing vectors is specified. |
| **8** | Count of data mapping data is specified. |
| **16** | Count of data morphing vectors is specified. |

*pdata* **— specified by user, array of primitive data**
Contains specific information about the entire primitive. The presence of optional fields is determined by the value of the *pflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Number of Subareas — fullword integer (>=0)**
Number of contours in the polygon definition. This field is required.

**Geometric Normal — 3 short floating-point numbers (MC)**
Geometric normal to be used in processing this polygon. This field is optional.

**Convexity Flag — fullword integer**
This data indicates that the application determined the convexity of the polygon (`0=CONCAVE, 1=CONVEX`). This field is optional.

**Vertex Morphing Vector Count — fullword integer (>=0)**
This number of vertex morphing vectors specified at each vertex. The number of fullwords of vertex morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

**Data Mapping Data Count — fullword integer (>=0)**
The number of data mapping values specified at each vertex. The number of data mapping values added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

**Data Morphing Vector Count — fullword integer (>=0)**
The number of data morphing vectors specified at each vertex. The number of fullwords of data morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

*saflags* **— specified by user, fullword integer**
Shows what optional data is specified for each subarea. The value specified should be the sum of the following values based on the fields that are included in the *sadata* parameter.

| Value | Corresponding Field |
|---|---|
| **0** | Number of vertices. (There is no optional data currently defined. This field must be set to zero.) |

*sawidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *sadata* parameter array (>=1)

*sadata* **— specified by user, array of per subarea data**
Contains specific information about each subarea in the primitive. The length of this array is defined by the contents of the `number of subareas` field in the *pdata* parameter. The presence of optional fields is determined by the value of the *saflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Number of Vertices — fullword integer (>=0)**
Number of vertices in each subarea. If the number is less than three, the result is workstation dependent. This field is required for each subarea.

*vxflags* **— specified by user, fullword integer**
Shows what optional data is specified for each vertex. The value specified should be the sum of the following values based on the fields that are specified in the *vxdata* parameter.

| Value | Meaning |
|---|---|
| **0** | Vertex coordinates are specified. |
| **1** | Vertex normals are specified. |
| **2** | Vertex colors are specified. |
| **4** | Boundary flags are specified. |
| **8** | Transparency coefficient is specified. |
| **16** | Vertex morphing vectors are specified. |
| **32** | Data mapping data is specified. |
| **64** | Data morphing vectors are specified (valid only if data mapping data is specified also). |

*vxwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *vxdata* parameter array (>=3)

*vxdata* **— specified by user, array of vertex data**
Contains specific information about each vertex in the primitive. The length of this array is equal to the sum of the `number of vertices` fields in the *sadata* parameter. The presence of optional fields is determined by the value of the *vxflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Coordinates — 3 short floating-point numbers (MC)**
*x, y*, and *z* coordinates of a vertex. This field is required.

**Normal — 3 short floating-point numbers (MC)**
The three components of a vector that is to be used as the normal of the polygon at the corresponding vertex. This field is optional.

**Color — 3 short floating-point numbers**
The three components of a color in the current color model as contained in the graPHIGS API state list. This field is optional.

**Boundary Flags — fullword integer**
Specifies whether the corresponding boundary is to be treated as an edge of the polygon (`1=NOT_AN_EDGE, 2=IS_AN_EDGE`). These flags allow control over which parts of the polygon's boundary are to be treated as edges. The edge attributes are only applied to boundary segments that have a boundary flag set to the value of `2=IS_AN_EDGE`.

Each flag specifies whether the boundary from the corresponding vertex to the following vertex is to be drawn. The last entry for each subarea corresponds to the boundary from the last vertex to the first. This field is optional.

**Transparency Coefficient — short floating-point number (0.0 <= transparency coefficient <=1.0)** The transparency coefficient value used when performing transparency processing. A value of 0.0 is fully opaque; a value of 1.0 is fully transparent. This field is optional.

**Vertex Morphing Vectors — array of short floating-point numbers**
The vertex morphing vectors $dx_1$, $dy_1$, $dz_1$, $dx_2$, $dy_2$, $dz_2$, ..., $dx_n$, $dy_n$, $dz_n$. The number, *n*, of vectors in this array is specified in the *pdata* parameter as the vertex morphing vector count. The array must be the same length for every vertex. This field is optional.

**Data Mapping Data — array of short floating-point numbers**
The data mapping data values $x_1$, $x_2$, $x_3$, ..., $x_n$. The number, *n*, of values in this array is specified in the *pdata* parameter as the data mapping data count. The array must be the same length for every vertex. This field is optional.

**Data Morphing Vectors — array of short floating-point numbers**
The data morphing vectors $d_{11}$>, $d_{12}$>, $d_{13}$>, ..., $d_{1n}$>, $d_{21}$>, $d_{22}$>, $d_{23}$>, ..., $d_{2n}$>, ..., $d_{m1}$, $d_{m2}$, $d_{m3}$, ..., $d_{mn}$. The number, *n*, is specified in the *pdata* parameter as the data mapping data count, and the number, *m* is specified in the *pdata* parameter as the data morphing vector count. The array must be the same length for every vertex. This field is optional.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**115** TRANSPARENT COEFFICIENT IS INVALID

**198** NUMBER OF SUBAREAS < ZERO

**199** POLYGON SUBAREA HAS < ZERO POINTS

**349** NORMAL VECTOR HAS ZERO LENGTH

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**352** BOUNDARY FLAG IS INVALID

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**636** FULLWORDS OF VERTEX DATA EXCEEDS MAXIMUM OF 255

### Related Subroutines

**GPBDMI**
Set Back Data Mapping Index

**GPBDMF**
Set Back Data Morphing Factors

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBISM**
Set Back Interior Shading Method

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPBTCO**
Set Back Transparency Coefficient

**GPDMI**
Set Data Mapping Index

**GPDMF**
Set Data Morphing Factors

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPISM**
Set Interior Shading Method

**GPPGC**
Set Polygon Culling

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCO**
Set Transparency Coefficient

**GPVMF**
Set Vertex Morphing Factors

**RCP code**

201342209 (X'0C003D01')

## GPPG2 - Polygon 2

**GPPG2 (*areas*, *npoint*, *width*, *pointlist*)**

### Purpose

Use **GPPG2** to specify a two-dimensional polygon primitive and insert it into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polygon 2 structure element depending on the current edit mode.

This structure element defines the boundary of contours which may be hollow or filled with a uniform color, a pattern, or a hatch style. You can display the boundary of the primitive with or without an edge. The normal on Polygon 2 is (0,0,1).

All points specified are placed in the *x*, *y* plane. Polygon attributes are applied to this primitive.

### Parameters

*areas* **— specified by user, fullword integer**
Number of subareas (>=0).

*npoint* **— specified by user, array of fullword integers**
Number of points per subarea (>=0).

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values (>=2).

*pointlist* **— specified by user, array of short floating-point numbers (MC)**
Array of points specified in row order.

The input array contains a list of points in which subsequent *x* values are separated by *width* fullwords.

For the $i^{th}$ subarea, the array contains 2 x npoint(i) short floating-point numbers. The pointlist is a continuous list of points with the points for subarea *i*+1 immediately following those of subarea *i*.

All points in the pointlist must be coplanar.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**198** NUMBER OF SUBAREAS < ZERO

**199** POLYGON SUBAREA HAS < ZERO POINTS

**557** WIDTH PARAMETER < MINIMUM ALLOWED

### Related Subroutines

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPLMO**
Set Lighting Calculation Mode

**GPLSS**
Set Light Source State

**GPPGC**
Set Polygon Culling

**GPSAC**
Set Surface Approximation Criteria

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCAC**
Set Trimming Curve Approximation Criteria

**RCP code**

201328136 (X'0C000608')

## GPPG3 - Polygon 3

**GPPG3 (*areas*, *npoint*, *width*, *pointlist*)**

### Purpose

Use **GPPG3** to specify a three-dimensional polygon primitive element and insert it into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polygon 3 structure element depending on the current edit mode.

This structure element defines the boundary of contours which may be hollow or filled with a uniform color pattern, or hatch style. You can display the boundary of the primitive with or without an edge.

All points specified must lie in the same plane, but the graPHIGS API does not check to verify this. The system behavior is undefined in this case.

Polygon attributes are applied to this primitive.

### Parameters

*areas* **— specified by user, fullword integer**
Number of subareas (>=0).

*npoint* **— specified by user, array of fullword integers**
Number of points per subarea (>=0).

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values (>=3).

*pointlist* **— specified by user, array of short floating-point numbers (MC)**
Array of points specified in row order. The input array contains a list of points in which subsequent *x* values are separated by *width* fullwords.

For the $i^{th}$ subarea, the array contains 3 x npoint(i) short floating-point numbers. The pointlist is a continuous list of points with the points for subarea *i*+1 immediately following those of subarea *i*. All points in the pointlist must be coplaner.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**198** NUMBER OF SUBAREAS < ZERO

**199** POLYGON SUBAREA HAS < ZERO POINTS

**557** WIDTH PARAMETER < MINIMUM ALLOWED

### Related Subroutines

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPLMO**
Set Lighting Calculation Mode

**GPLSS**
Set Light Source State

**GPPGC**
Set Polygon Culling

**GPSAC**
Set Surface Approximation Criteria

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCAC**
Set Trimming Curve Approximation Criteria

### RCP code

201328135 (X'0C000607')

## GPPHE - Polyhedron Edge

**GPPHE (*nedge*, *edgelist*)**

### Purpose

Use **GPPHE** to insert a Polyhedron Edge structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polyhedron Edge structure element depending on the current edit mode.

You can use this primitive to simulate the edge of a polyhedron type object. The two normals identify the orientation of the faces which intersect and result in the line segment defined by the two endpoints.

During structure traversal, the graPHIGS API uses the polyline attributes and Polyhedron Edge Culling mode (**GPPHEC**) from the traversal state list to render each line segment defined by two end points. The Polyhedron Edge Culling mode along with two associated normals within an *edgelist* entry may suppress rendering of this line segment.

All normal vectors are normalized by the graPHIGS API. If the application later inquires the content of this structure element, then the graPHIGS API returns the normalized vectors *not* the original vectors specified on this subroutine.

**GPPHE** is identified as GDP 1037.

> **Note:** Not all GDPs are supported on all workstations. (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*nedge* **— specified by user, fullword integer**
Number of edges (>=1)

*edgelist* **— specified by user, array of 12 short floating-point numbers (MC)**
Array of polyhedron edge descriptors. Each polyhedron descriptor consists of the *x*, *y*, and *z* coordinates of two normal vectors and two end points in this order.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**349** NORMAL VECTOR HAS ZERO LENGTH

**363** NUMBER OF EDGES < ONE

**Related Subroutines**

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPHEC**
Set Polyhedron Edge Culling

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLET**
Set Polyline End Type

**GPPLI** Set Polyline Index

**RCP code**

201344001 (X'0C004401')

# GPPL2 - Polyline 2

**GPPL2 (*npoint*, *width*, *pointlist*)**

### Purpose

Use **GPPL2** to create a two-dimensional polyline element and insert it into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polyline 2 structure element depending on the current edit mode.

This structure element defines a list of two-dimensional points (*x*, *y*) (the *z* coordinate is assumed to be zero) that the graPHIGS API is to connect by straight lines starting with the first point and ending with the last point.

If the application specifies one point or less, then no output is generated. If the application specifies two contiguous points as the same point, then the graPHIGS API generates a point of one pixel in size.

Polyline attributes are applied to this primitive.

### Parameters

*npoint* **— specified by user, fullword integer**
Number of points (>=0).

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values (>=2).

*pointlist* **— specified by user, array of short floating-point numbers (MC)**
Array of points specified in row order.

The input array contains a list of points in which subsequent *x* values are separated by *width* fullwords.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**100** NUMBER OF POINTS < ZERO

**557** WIDTH PARAMETER < MINIMUM ALLOWED

### Related Subroutines

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLET**
Set Polyline End Type

**GPPLI** Set Polyline Index

### RCP code

201328130 (X'0C000602')

## GPPL3 - Polyline 3

**GPPL3 (*npoint, width, pointlist*)**

### Purpose

Use **GPPL3** to create a three-dimensional polyline element and insert it into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polyline 3 structure element depending on the current edit mode.

This structure element defines a list of three-dimensional points (*x*, *y*, and *z*) that the graPHIGS API is to connect by straight lines starting with the first point and ending with the last point.

If the application specifies one point or less, then no output is generated. If the application specifies two contiguous points as the same point, then the graPHIGS API generates a point of one pixel in size.

Polyline attributes are applied to this primitive.

### Parameters

*npoint* **— specified by user, fullword integer**
Number of points (>=0).

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values (>=3).

*pointlist* **— specified by user, array of short floating-point numbers (MC)**
Array of points specified in row order. The input array contains a list of points in which subsequent *x* values are separated by *width* fullwords.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**100** NUMBER OF POINTS < ZERO

**557** WIDTH PARAMETER < MINIMUM ALLOWED

### Related Subroutines

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLET**
Set Polyline End Type

**GPPLI** Set Polyline Index

### RCP code

201328129 (X'0C000601')

# GPPLD3 - Polyline Set 3 With Data

**GPPLD3 (*pflags*, *pdata*, *plflags*, *plwidth*, *pldata*, *vxflags*, *vxwidth*, *vxdata*)**

### Purpose

Use **GPPLD3** to insert a Polyline Set 3 With Data structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polyline Set 3 With Data structure element depending on the current edit mode.

This structure element defines multiple three-dimensional polylines within one structure element. This is similar to the Disjoint Polyline 2 (**GPDPL2**) subroutine and the Disjoint Polyline 3 (**GPDPL3**) subroutine, except in its specification. When encountered during structure traversal, this element generates an unconnected sequence of polylines from the list of points specified.

Your application can specify optional data to further control the rendering of this primitive. The optional data consists of:

- Vertex colors.

  When rendering this primitive, if the primitive is not to be highlighted, then the graPHIGS API uses the specified vertex colors. If the current polyline shading method is `1=POLYLINE_SHADING_NONE` (default), then the graPHIGS API uses the $i^{th}$ vertex color to color the $i^{th}$ line of the polyline. If the current polyline shading method is `2=POLYLINE_SHADING_COLOR`, then the graPHIGS API interpolates the color along each line between the colors specified at the endpoints of the line. If your application does not specify the vertex color data in this primitive definition, then the graPHIGS API uses the current polyline color to render **GPPLD3**.
- Vertex morphing vectors.

  You can supply vertex morphing vectors per vertex. The graPHIGS API combines these vectors with the vertices and vertex morphing scale factors (**GPVMF**) to create new vertex coordinate values for the rendered primitive.

See *The graPHIGS Programming Interface: Understanding Concepts* for a more complete explanation of how the graPHIGS API uses the various optional data values.

> **Note:** This note applies ONLY to applications which will be run on the High Performance 3D Color Graphics Processor (8 or 24 bit). Use of any optional data other than vertex colors may cause unpredictable results (including locking the display) on this graphics processor. If only the High Performance 3D Color Graphics Processor is used, you should include only the supported optional data values. If your application must support multiple graphics processors INCLUDING this particular processor, the Inquire Workstation Description (**GPQWDT**) subroutine must be used to determine the functions that each workstation supports. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference* for the High Performance 3D Color Graphics Processor.

Polyline attributes are applied to this primitive.

**GPPLD3** is identified as GDP 1014.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*pflags* **— specified by user, fullword integer**
Shows what optional data is specified for the primitive. The value specified should be the sum of the following values based on the fields that are included in the *pdata* parameter.

| Value | Corresponding Field |
|---|---|
| **0** | Number of polylines. |
| **4** | Count of vertex morphing vectors is specified. |

*pdata* **— specified by user, array of primitive data**
Contains specific information about the entire primitive. The presence of optional fields is determined by the value of the *pflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Number of Polylines — fullword integer (>=0)**
Number of specified polylines. This field is required.

**Vertex Morphing Vector Count — fullword integer (>=0)**
The number of vertex morphing vectors specified at each vertex. The number of fullwords of vertex morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

*plflags* **— specified by user, fullword integer**
Shows what optional data is specified for each polyline. The value specified should be the sum of the following values based on the fields that are included in the *pldata* parameter.

| Value | Corresponding Field |
|---|---|
| **0** | Number of vertices in each polyline. (There is no optional data currently defined. This field must be set to zero). |

*plwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *pldata* parameter array (>=1).

*pldata* **— specified by user, array of per polyline data**
Contains specific information about each polyline in the primitive. The length of this array is defined by the contents of the `number of polylines` field in the *pdata* parameter. The presence of optional fields is determined by the value of the *plflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Number of Vertices — fullword integer (>=0)**
Number of vertices in each polyline. If the number is less than two, then no lines are generated. This field is required for each polyline.

*vxflags* **— specified by user, fullword integer**
Shows what optional data is specified for each vertex. The value specified should be the sum of the following values based on the fields that are specified in the *vxdata* parameter.

| Value | Meaning |
|---|---|
| **0** | Vertex coordinates are specified. |
| **2** | Vertex colors are specified. |
| **16** | Vertex morphing vectors are specified. |

*vxwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *vxdata* parameter array (>=3).

*vxdata* **— specified by user, array of vertex data**
Contains specific information about each vertex in the primitive. The length of this array is equal to the sum of the `number of vertices` fields in the *pldata* parameter. The presence of optional fields is determined by the value of the *vxflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Coordinates — 3 short floating-point numbers (MC)**
*x, y* and *z* coordinates of each vertex. This field is required.

**Color — 3 short floating-point numbers (0.0<=component<=1.0)**
The three components of a color in the current color model as contained in the graPHIGS API state list. This field is optional.

**Vertex Morphing Vectors — array of short floating-point numbers**
The vertex morphing vectors $dx_1$, $dy_1$, $dz_1$, $dx_2$, $dy_2$, $dz_2$, ..., $dx_n$, $dy_n$, $dz_n$. The number, *n*, of vectors in this array is specified in the *pdata* parameter as the vertex morphing vector count. The array must be the same length for every vertex. This field is optional.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**100** NUMBER OF POINTS < ZERO

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**356** NUMBER OF POLYLINES < ZERO

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**636** FULLWORDS OF VERTEX DATA EXCEEDS MAXIMUM OF 255

**Related Subroutines**

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLET**
Set Polyline End Type

**GPPLI** Set Polyline Index

**GPPLSM**
Set Polyline Shading Method

**GPTCO**
Set Transparency Coefficient

**GPVMF**
Set Vertex Morphing Factors

**RCP code**

201342211 (X'0C003D03')

## GPPM2 - Polymarker 2

**GPPM2 (*npoint*, *width*, *pointlist*)**

**Purpose**

Use **GPPM2** to create a two-dimensional polymarker element and insert it into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polymarker 2 structure element depending on the current edit mode.

This structure element defines a list of two-dimensional points (*x*, *y*) that the graPHIGS API identifies by markers and renders in Device Coordinate (DC) space parallel to the display surface.

If the primitive does not specify any points, it is ignored.

Polymarker attributes are applied to this primitive.

**Parameters**

*npoint* **— specified by user, fullword integer**
Number of points (>=0)

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values (>=2)

*pointlist* **— specified by user, array of short floating-point numbers (MC)**
Array of points specified in row order. The input array contains a list of points in which subsequent *x* values are separated by *width* fullwords.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**100** NUMBER OF POINTS < ZERO

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**Related Subroutines**

**GPMSSC**
Set Marker Size Scale Factor

**GPMT** Set Marker Type

**GPPMCD**
Set Polymarker Color Direct

**GPPMCI**
Set Polymarker Color Index

**GPPMI**
Set Polymarker Index

**RCP code**

201328132 (X'0C000604')

# GPPM3 - Polymarker 3

**GPPM3 (*npoint*, *width*, *pointlist*)**

### Purpose

Use **GPPM3** to create a three-dimensional polymarker element and insert it into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polymarker 3 structure element depending on the current edit mode.

This structure element defines a list of three-dimensional points (*x*, *y*, and *z*) that the graPHIGS API identifies by markers and renders in Device Coordinate (DC) space parallel to the display surface.

If the primitive does not specify any points, it is ignored.

Polymarker attributes are applied to this primitive.

### Parameters

*npoint* **— specified by user, fullword integer**
Number of points (>=0)

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values (>=3)

*pointlist* **— specified by user, array of short floating-point values (MC)**
Array of points specified in row order.

The input array contains a list of points in which subsequent *x* values are separated by *width* fullwords.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**100** NUMBER OF POINTS < ZERO

**557** WIDTH PARAMETER < MINIMUM ALLOWED

### Related Subroutines

**GPMSSC**
Set Marker Size Scale Factor

**GPMT** Set Marker Type

**GPPMCD**
Set Polymarker Color Direct

**GPPMCI**
Set Polymarker Color Index

**GPPMI**
Set Polymarker Index

### RCP code

201328131 (X'0C000603')

# GPPXL2 - Pixel 2

**GPPXL2 (*point*, *pack*, *numrow*, *numcol*, *startrow*, *startcol*, *nrow*, *ncol*, *array*)**

### Purpose

Use **GPPXL2** to insert a Pixel 2 element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Pixel 2 structure element depending on the current edit mode.

This structure element defines a two-dimensional rectangular array of pixels that the application writes into the frame buffer of a workstation. The pixel values specify indexes into the workstation's display color table. Only the low order byte of the pixel array is used and preserved by this primitive. The graPHIGS API assumes the primitive exists in the *z*=0 plane.

**GPPXL2** is identified as GDP 1002.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*point* **— specified by user, 2 short floating-point numbers (MC)**
*x* and *y* coordinates of the upper left corner for start of pixel rectangle.

*pack* **— specified by user, fullword integer**
Specifies the packing factor of the pixel color indexes in the input array (`8 BITS/PIXEL`, `16 BITS/PIXEL`, `32 BITS/PIXEL`). The interpretation of the following parameters depends on this value. The graPHIGS API uses the least significant 8 bits only.

*numrow* **— specified by user, fullword integer**
Number of rows in the input array (>=1).

*numcol* **— specified by user, fullword integer**
Number of columns in the input array (>=1).

*startrow* **— specified by user, fullword integer**
Row within array that is the start of the data (>=1).

*startcol* **— specified by user, fullword integer**
Column within array that is the start of the data (>=1).

*nrow* **— specified by user, fullword integer**
Number of rows within array to be used for display beginning at the starting position (>=1).

*ncol* **— specified by user, fullword integer**
Number of columns within the array to be used for display beginning at the starting position (>=1).

*array* **— specified by user, array of integers (as defined by the packing factor)**
A grid of *numrow* by *numcol* color indexes. The array must be in row order.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**91** STARTING POINT OR DIMENSION < ONE

**537** PATTERN OR PIXEL ARRAY EXCEEDS INPUT ARRAY SIZE

**549** INVALID PIXEL PACK FACTOR

**Related Subroutines**

None

**RCP code**

201328143 (X'0C00060F')

## GPPXL3 - Pixel 3

**GPPXL3 (*point*, *pack*, *numrow*, *numcol*, *startrow*, *startcol*, *nrow*, *ncol*, *array*)**

**Purpose**

Use **GPPXL3** to insert a Pixel 3 element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Pixel 3 structure element depending on the current edit mode.

This structure element defines a two-dimensional rectangular array of pixels that the application writes into the frame buffer of a workstation. The pixel values specify indexes into the workstation's display color table. Only the low order byte of the pixel array is used and preserved by this primitive.

**GPPXL3** is identified as GDP 1001.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*point* **— specified by user, 3 short floating-point numbers (MC)**
*x*, *y* and *z* coordinates of the upper left corner for start of pixel rectangle.

*pack* **— specified by user, fullword integer**
Specifies the packing factor of the pixel color indexes in the input array (`8 BITS/PIXEL`, `16 BITS/PIXEL`, `32 BITS/PIXEL`). The interpretation of the following parameters depend on this value. The graPHIGS API uses the least significant 8 bits only.

*numrow* **— specified by user, fullword integer**
Number of rows in the input array (>=1).

*numcol* **— specified by user, fullword integer**
Number of columns in the input array (>=1).

*startrow* **— specified by user, fullword integer**
Row within the array that is the start of the data (>=1).

*startcol* **— specified by user, fullword integer**
Column within array that is the start of the data (>=1).

*nrow* **— specified by user, fullword integer**
Number of rows within the array to be used for display beginning at the starting position (>=1).

*ncol* **— specified by user, fullword integer**
Number of columns within the array to be used for display beginning at the starting position (>=1).

*array* **— specified by user, array of integers (as defined by the packing factor)**
A grid of *numrow* by *numcol* color indexes. The array must be in row order.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**91** STARTING POINT OR DIMENSION < ONE

**537** PATTERN OR PIXEL ARRAY EXCEEDS INPUT ARRAY SIZE

**549** INVALID PIXEL PACK FACTOR

**Related Subroutines**

None

**RCP code**

201328142 (X'0C00060E')

## GPQM3 - Quadrilateral Mesh 3

**GPQM3 (*mflags*, *mdata*, *qflags*, *qwidth*, *qdata*, *vxflags*, *vxwidth*, *vxdata*)**

**Purpose**

Use **GPQM3** to insert a Quadrilateral Mesh 3 structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Quadrilateral Mesh 3 structure element, depending on the current edit mode.

When encountered during structure traversal, this element generates an array of (M minus 1) x (N minus 1) quadrilaterals from a two-dimensional array of M x N vertices. Each quadrilateral is generated by four neighboring points in the vertex array, vertices *v(i, j)*, v(i + 1, j), v(i + 1, j + 1), and v(i, j + 1), where 1<= *i*<= *M* and 1<= *j*<= *N*. Each quadrilateral is a facet of the primitive. For more information on quardrilaterals, see *The graPHIGS Programming Interface: Understanding Concepts*.

Your application can specify optional data to further control the rendering of this primitive.

- Convexity flag.

  The convexity flag indicates that the application determined the convexity of the quadrilateral mesh primitive. Therefore, the system rendering code does not have to determine the convexity every time the primitive is rendered.
- Normals.

  You can specify a geometric normal for each quadrilateral and a reflectance normal for each vertex. The normals are used in the lighting process to produce more realistic effects.
- Colors.

  You can specify color for each quadrilateral and/or each vertex. When rendering this primitive, if the primitive is to be highlighted, then the graPHIGS API uses the highlight color instead. The colors are used in the lighting process to produce more realistic effects.
- Boundary flags.

  The edges of this primitive consist of line segments forming the boundary of each quadrilateral in the mesh. Use boundary flags to identify the boundaries that you want rendered as edges. More boundary flags are specified than actually used, but the graPHIGS API ignores the unused boundary flags. If there are no boundary flags specified, then all boundaries are rendered as edges. If the edge flag is set to `2=ON` and the line type is *not* set to `1=SOLID_LINE` then the results are unpredictable due to the potential double drawing of some edges on some workstations.
- Transparency coefficients.

You can specify a transparency coefficient per vertex. The graPHIGS API uses these values when producing transparency effects for the rendered primitive.

- Vertex morphing vectors.

  You can supply vertex morphing vectors per vertex. The graPHIGS API combines these vectors with the vertices and vertex morphing scale factors (**GPVMF**)to create new vertex coordinate values for the rendered primitive.
- Data mapping data.

  You can specify data mapping data per vertex. The graPHIGS API uses these values to determine the colors of the rendered primitive.
- Data morphing vectors.

  You can specify data morphing vectors per vertex. The graPHIGS API combines these vectors with the data morphing scale factors (**GPDMF**)and (**GPBDMF**)and the vertex data mapping values to create new data mapping data values for the rendered primitive.

See *The graPHIGS Programming Interface: Understanding Concepts* for a more complete explanation of how the graPHIGS API uses the various optional data values.

> **Note:** ***This note applies ONLY to applications which will be run on the High Performance 3D Color Graphics Processor (8 or 24 bit).*** Use of any optional data other than the convexity flag, vertex normals, quadrilateral normals, quadrilateral colors, vertex colors, and boundary flags may cause unpredictable results (including locking the display) on this graphics processor. If only the High Performance 3D Color Graphics Processor is used, you should include only the supported optional data values. If your application must support multiple graphics processors INCLUDING this particular processor, the Inquire Workstation Description (**GPQWDT**) subroutine must be used to determine the functions that each workstation supports. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference* for the High Performance 3D Color Graphics Processor.

When the vertices of a quadrilateral are not coplanar, then the method of rendering the quadrilateral is workstation dependent.

During structure traversal, the graPHIGS API ignores quadrilateral meshes with less than two vertices in either direction, i.e., there is no visual effect.

Polygon attributes are applied to this primitive.

**GPQM3** is identified as GDP 1031.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*mflags* **— specified by user, fullword integer**

Shows what optional data is specified for the primitive. The value specified should be the sum of the following values based on the fields that are included in the *mdata* parameter.

| Value | Corresponding Field |
|---|---|
| **0** | Dimensions of vertex array |
| **2** | Convexity flag is specified. |
| **4** | Count of vertex morphing vectors is specified. |
| **8** | Count of data mapping data is specified. |
| **16** | Count of data morphing vectors is specified. |

*mdata* **— specified by user, array of primitive data**

Contains specific information about the entire primitive. The presence of optional fields is determined by the value of the *mflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Dimensions of vertex array — 2 fullword integers (>=0)**

The dimensions of the vertex array. The first value (*row_dim*) defines the number of vertices in a row of the array. The second value (*col_dim*) defines the number of vertices in a column of the array. The number of quadrilaterals generated in each dimension is one less than its corresponding vertex array dimension. This field is required.

**Convexity flag — Fullword integer**

This data indicates that your application determined the convexity of the mesh (`0=CONCAVE`, `1=CONVEX`). If any individual quadrilateral is concave, then set the flag to `0=CONCAVE`. Set the flag to `1=CONVEX` only if all the quadrilaterals are convex. This allows the workstation to optimize processing of the primitive. This field is optional.

**Vertex Morphing Vector Count — fullword integer (>=0)**

The number of vertex morphing vectors specified at each vertex. The number of fullwords of vertex morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

**Data Mapping Data Count — fullword integer (>=0)**

The number of data mapping values specified at each vertex. The number of data mapping values added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

**Data Morphing Vector Count — fullword integer (>=0)**

The number of data morphing vectors specified at each vertex. The number of fullwords of data morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

*qflags* **— specified by user, fullword integer**

Shows what optional data is specified for each quadrilateral. The value specified should be the sum of the following values based on the fields that are included in the *qdata* parameter.

| Value | Corresponding Field |
|---|---|
| **0** | Null, *qdata* is not referenced. |
| **1** | Geometric normals. |
| **2** | Color. |

*qwidth* **— specified by user, fullword integer (>=0)**

Number of words between subsequent entries of the *qdata* parameter array. The minimum value for this parameter is dependent on the optional data that is specified.

*qdata* **— specified by user, array of per quadrilateral data**

Contains specific information about each quadrilateral in the primitive. The length of this array is (row_dim minus 1) times (col_dim minus 1) where *row_dim* and *col_dim* are the vertex array dimensions as defined in *mdata*. The entries of this array are stored in row major order. The presence of optional fields is determined by the value of the *qflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Geometric normal — 3 short floating-point numbers (MC)**

Geometric normal to be used in processing this quadrilateral. This field is optional.

**Color — 3 short floating-point numbers (0.0<=component<=1.0)**
The three components of a color in the current color model as contained in the graPHIGS API state list. This field is optional.

*vxflags* **— specified by user, fullword integer**
Shows what optional data is specified for each vertex. The value specified should be the sum of the following values based on the fields that are specified in the *vxdata* parameter.

| Value | Meaning |
|---|---|
| **0** | Vertex coordinates are specified. |
| **1** | Vertex normals are specified. |
| **2** | Vertex colors are specified. |
| **4** | Boundary flags are specified. |
| **8** | Transparency coefficient is specified. |
| **16** | Vertex morphing vectors are specified. |
| **32** | Data mapping data is specified. |
| **64** | Data morphing vectors are specified (valid only if date mapping data is specified also). |

*vxwidth* **— specified by user, fullword integer (>=3)**
Number of words between subsequent data entries of the *vxdata* parameter array. The minimum value for this parameter depends on the optional data that is specified.

*vxdata* **— specified by user, array of vertex data**
Contains specific information about each vertex in the primitive. The length of the array is *row_dim* [default] *col_dim*, where *row_dim* and *col_dim* are the dimensions of the vertex array as specified in the *mdata* parameter. The entries are stored in row major order. The presence of optional fields is determined by the value of the *vxflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Coordinates — 3 short floating-point numbers (MC)**
*x*, *y*, and *z* coordinates of a vertex. This field is required.

**Normal — 3 short floating-point numbers (MC)**
The three components of a vector that is to be used as the normal of the quadrilateral at the corresponding vertex. This field is optional.

**Color — 3 short floating-point numbers (0.0<=component<=1.0)**
The three components of a color in the current color model as contained in the graPHIGS API state list. This field is optional.

**Boundary flags — 2 fullword integers**
Specifies whether the corresponding boundary is to be treated as an edge of the quadrilateral (`1=NOT_AN_EDGE`, `2=IS_AN_EDGE`). These flags allow control over which parts of the quadrilateral's boundary are to be treated as edges. The edge attributes are only applied to boundary segments that have a boundary flag set to a value of `2=IS_AN_EDGE`.

Each vertex *v(i, j)* has two boundary flags which specify whether the graPHIGS API draws the boundary from the specified vertex to an adjacent vertex. The first boundary flag applies to the boundary from vertex *v(i, j)* to vertex *v(i + 1, j)* and the second boundary flag applies to the boundary from vertex *v(i, j)* to vertex *v(i, j + 1)*, where 1<= *i*<= *col_dim* and 1<= *j*<= *row_dim*. There are more vertices than boundary flags, but the graPHIGS API ignores the unused boundary flags. This field is optional.

**Transparency Coefficient — short floating-point number (0.0 <= transparency coefficient <=1.0)** The transparency coefficient value used when performing transparency processing. A value of 0.0 is fully opaque; a value of 1.0 is fully transparent. This field is optional.

**Vertex Morphing Vectors — array of short floating-point numbers**
The vertex morphing vectors $dx_1$, $dy_{\#EMPTY>1}$, $dz_1$, $dx_2$, $dy_2$, $dz_2$, ..., $dx_n$, $dy_n$, $dz_{\#EMPTY>n}$. The number, *n*, of vectors in this array is specified in the *pdata* parameter as the vertex morphing vector count. The array must be the same length for every vertex. This field is optional.

**Data Mapping Data — array of short floating-point numbers**
The data mapping data values $x_1$, $x_{\#EMPTY>2}$, $x_3$, ..., $x_{\#EMPTY>n}$. The number, *n*, of values in this array is specified in the *pdata* parameter as the data mapping data count. The array must be the same length for every vertex. This field is optional.

**Data Morphing Vectors — array of short floating-point numbers**
The data morphing vectors $d_1$ 1, $d_{\#EMPTY>1}$ 2, $d_1$ 3, ..., $d_{\#EMPTY>1}$ n, $d_2$ 1, $d_2$ 2, $d_2$ 3, ..., $d_2$ n, ..., $d_{m1}$ , $d_{m2}$, $d_{\#EMPTY>m3}$, ..., $d_{mn}$

The data morphing vectors $d_{11}$>, $d_{\#EMPTY>12}$>, $d_{13}$>, ..., $d_{\#EMPTY>1n}$>, $d_{21}$>, $d_{\#EMPTY>22}$>, $d_{23}$>, ..., $d_{\#EMPTY>2n}$>, ..., $d_{m1}$, $d_{\#EMPTY>m2}$, $d_{m3}$, ..., $d_{\#EMPTY>mn}$. The number, *n*, is specified in the *pdata* parameter as the data mapping data count, and the number, *m* is specified in the *pdata* parameter as the data morphing vector count. The array must be the same length for every vertex. This field is optional.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**115** TRANSPARENT COEFFICIENT IS INVALID

**349** NORMAL VECTOR HAS ZERO LENGTH

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**352** BOUNDARY FLAG IS INVALID

**357** DIMENSION OF VERTEX ARRAY < ZERO

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**636** FULLWORDS OF VERTEX DATA EXCEEDS MAXIMUM OF 255

### Related Subroutines

**GPBDMF**
Set Back Data Morphing Factors

**GPBDMI**
Set Back Data Mapping Index

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBISM**
Set Back Interior Shading Method

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPBTCO**
Set Back Transparency Coefficient

**GPDMF**
Set Data Morphing Factors

**GPDMI**
Set Data Mapping Index

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPISM**
Set Interior Shading Method

**GPPGC**
Set Polygon Culling

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCO**
Set Transparency Coefficient

**GPVMF**
Set Vertex Morphing Factors

**RCP code**

201343747 (X'0C004303')

## GPSPH - Polysphere

**GPSPH (*nsphere*, *pflags*, *pdata*, *width*, *spherelist*)**

### Purpose

Use **GPSPH** to insert a Polysphere structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Polysphere structure element depending on the current edit mode.

When encountered during structure traversal, this element generates a sphere or a sequence of spheres each defined by a center point and a radius in Modelling Coordinates (MC). Each sphere is subject to all transformations and clipping.

All polygon interior and surface attributes are applied to this primitive.

Since the sphere has no defined boundary or edge, edge attributes do not apply to this primitive (i.e., `1=HOLLOW` and `5=EMPTY` have no visual effect) Highlight color and pick echo for `1=HOLLOW` and `5=EMPTY` which normally are applied to edges, are applied in a workstation dependent manner (i.e., highlight color and pick echo may be applied to tessellation lines).

**GPSPH** is identified as GDP 1046.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*nsphere* **— specified by user, fullword integer**
Number of spheres to be generated (>=0).

*pflags* **— specified by user, fullword integer**
Indicates what optional data is specified for this primitive. (There is no optional data currently defined. This field must be set to zero).

*pdata* **— specified by user, array of primitive data**
Contains specific information about the primitive. The presence of optional fields is determined by the value of the *pflags* parameter. (Since there is no optional data currently defined, this field must be set to zero).

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values (>=4).

*spherelist* **— specified by user, array of short floating point numbers (MC)**
Array of floating point numbers defining the sphere. The number of arrays is equal to the specified number of spheres. Each array contains:

**point** 3 short floating point numbers (MC)

*x*, *y* and *z* coordinates of the center of a sphere.

**radius** a short floating point number (MC)

Radius of the sphere (>=0)

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**101** NUMBER OF SPHERES < ZERO

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**582** RADIUS SPECIFIED < ZERO

**Related Subroutines**

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPLMO**
Set Lighting Calculation Mode

**GPLSS**
Set Light Source State

**GPPGC**
Set Polygon Culling

**GPSAC**
Set Surface Approximation Criteria

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCAC**
Set Trimming Curve Approximation Criteria

**RCP code**

201346817 (X'0C004F01')

## GPTNBS - Trimmed Non-Uniform B-Spline Surface

**GPTNBS (*uorder*, *vorder*, *unum*, *vnum*, *uknots*, *vknots*, *tflag*, *utess*, *vtess*, *cflags*, *cwidth*, *ctlpts*, *ncontour*, *ncurve*, *curveinfo*, *tknot*, *ttess*, *cdwidth*, *cddata*)**

### Purpose

Use **GPTNBS** to insert a Trimmed Non-Uniform B-Spline Surface structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Trimmed Non-Uniform B-Spline Surface structure element depending on the current edit mode.

At structure traversal time, a non-uniform parametric surface of the specified *uorder* and *vorder* are generated using the specified control points. Only the region of the surface which is bounded by an odd number of trimming curves is rendered.

This primitive generates no output if any of the following are true:

- The requested orders for the basis functions of the surface or trimming curves are not supported by the workstation.
- No trimming curves are specified.

The trimming curves *must* adhere to the following rules; otherwise, the results are unpredictable:

- Each loop must be explicitly closed.
- Curves cannot go outside the parameter range of the surface.
- The trimming curves within a loop must be connected in a head to tail fashion. They may not be randomly specified. The end of one curve must coincide with the beginning of the next curve.
- The trimming curves may not cross other trimming curves in the same or different loop.

The trimming curves have the following *optional* capability:

- A mix of rational and non-rational curves can be used on the same surface.
- The trimming curves can form multiple closed loops.
- Each curve is parameterized independently.

Polygon and surface attributes are applied to this primitive.

**GPTNBS** is identified as GDP 1036.

> **Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*uorder* **— specified by user, fullword integer**
Order of the basis functions for the *u* parameter (>=2).

*vorder* **— specified by user, fullword integer**
Order of the basis functions for the *v* parameter (>=2).

*unum* **— specified by user, fullword integer**
Number of surface control points for the *u* direction (>=uorder).

*vnum* **— specified by user, fullword integer**
Number of surface control points for the *v* direction (>=vorder).

*uknots* **— specified by user, array of short floating-point numbers**
Knot values for the *u* parameter. The length of this array must be *uorder*+ *unum*. This parameter must be a non-decreasing knot value sequence.

*vknots* **— specified by user, array of short floating-point numbers**
Knot values for the *v* parameter. The length of this array must be *vorder*+ *vnum* This parameter must be a non-decreasing knot value sequence.

*tflag* **— specified by user, fullword integer**
Surface tessellation quality value flag. This parameter shows whether the tessellation quality values are specified or not.

| Value | Meaning |
|---|---|
| **0** | Not specified. |
| **1** | Specified. |

*utess* **— specified by user, array of short floating-point numbers**
Tessellation quality values for the *u* direction. When the *tflag* parameter is set to a value of one, this parameter must contain *unum*- *uorder*+1 quality values. These values are used in conjunction with Surface Approximation Criteria method 8 to control the number of sub-divisions made in the *u* direction. The number of sub-divisions that are performed for a patch is approximately the product of this value and the Surface Approximation Criteria control value (*u*) in the traversal state list.

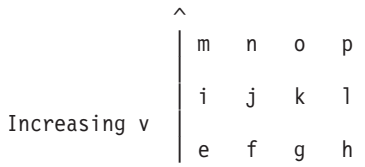
*vtess* **— specified by user, array of short floating-point numbers**
Tessellation quality values for the *v* direction. When the *tflag* parameter is set to a value of one, this parameter must contain *vnum*- *vorder*+1 quality values. These values are used in conjunction with Surface Approximation Criteria method 8 to control the number of sub-divisions made in the *v* direction. The number of sub-divisions that are performed for a patch is approximately the product of this value and the Surface Approximation Criteria control value (*v*) in the traversal state list.

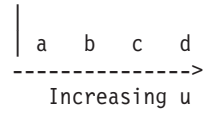
*cflags* **— specified by user, fullword integer**
Control point optional data flags. This parameter shows what data is specified for each control point. The value specified should be the sum of the following values based on the fields that are specified in the *ctlpts* parameter.

| Value | Meaning |
|---|---|
| **0** | Control point coordinates. |
| **1** | Weights are specified with each control point. This produces the rational form of the Non-Uniform B-Spline Surface. |

*cwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of control points array *ctlpts*.

*ctlpts* **— specified by user, array of short floating-point numbers.**
Grid of control points. The control points are stored by row where a row is considered to be the direction associated with the *u* parameter. For example, the set of control points






would be stored in the order a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, and p. The *cwidth* parameter must be at least three. If *cflags* specifies that weights are included with each control point, the *cwidth* parameter must be at least four. Each weight *W* must be greater than zero when specified.

**Note:** When *W* is specified, the control points are not in homogeneous form (i.e.,*XW*, *YW*, *ZW*, *W*). They are specified after division by *W* or (*X*, *Y*, *Z*, *W*).

*ncontour* **— specified by user, fullword integer**
Number of contours to be generated (>=0).

*ncurve* **— specified by user, array of fullword integers**
Number of curves in each contour. Each entry must be greater than or equal to one. The length of this array is defined by the value of the *ncontour* parameter.

*curveinfo* **— specified by user, array of curve data**
Array containing information about each curve. Each entry of this parameter must have the following fields in this order:

**Type of Curve — fullword integer**
This field must contain

**3 -** Non-Uniform B-Spline curve

**Options — fullword integer**
This parameter specifies various options of the curve. Each option is specified by a bit in this word and the following bits are currently defined.

| **Bit** | **Meaning** |
|---|---|
| **0-28** | Reserved. Must be set to zero. |
| **29** | Tessellation quality flag.<br><br>If set, a tessellation quality value for each span of this curve is specified in the *ttess* parameter. |
| **30** | Weight flag.<br><br>If set, the curve is rational and the weight is specified for each control point in the *cddata* parameter. |
| **31** | Boundary flag.<br><br>If set, the curve is treated as an edge of the composite fill area. |

**Order — fullword integer**
Order of the curve (>=2).

**Number of Data — fullword integer**
Number of entries of the *cddata* parameter used to define the curve. This parameter corresponds to the *npoint* parameter of the Non-Uniform B-Spline Curve 2. The specified number's entries of the *cddata* parameter are used as its *ctlpts* parameter.

**Start — short floating-point number**
The parameter value representing the start point of the curve.

**End — short floating-point number**
The parameter value representing the end point of the curve.

*tknot* **— specified by user, array of short floating-point numbers**
Array of knot values for the *t* direction of the curve. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo*. The length of each list equals the *order* + `number of data` for the curve.

*ttess* **— specified by user, array of short floating-point numbers**
Array of tessellation quality values. This array must contain one list for each Non-Uniform B-Spline curve with a tessellation quality flag set to a value of one (specified). For other curves, this array is not referenced. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo*. The length of each list equals the `number of data` - *order* + 1 of the curve.

*cdwidth* **— specified by user, fullword integer.**
Specifies the number of fullwords between each entry of the array in *cddata*. If there is any rational curve in the *curveinfo* parameter, this parameter must be at least three. Otherwise, it must be larger than or equal to two.

*cddata* **— specified by user, array of short floating-point numbers**
This array must contain one list for each curve. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo*. The length of each list equals the `number of data` field specified in the *curveinfo* parameter.

For each entry, the following fields are defined and the fields must be specified in this order without any gap.

**U, V components —**
2 short floating-point numbers

**weight —**
short floating-point number

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**341** ORDER OF BASIS FUNCTION < TWO

**342** ORDER IS GREATER THAN NUMBER OF CONTROL POINTS

**343** KNOT VECTOR IS INVALID

**345** WEIGHT IN CONTROL POINT IS <= ZERO

**347** PARAMETER LIMITS ARE OUTSIDE VALID PARAMETER RANGE

**348** MINIMUM PARAMETER LIMIT > MAXIMUM

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**353** NUMBER OF CONTOURS < ZERO

**354** NUMBER OF CURVES PER CONTOUR < ONE

**361** CURVE OPTIONS FIELD IS INVALID

**362** TESSELLATION CONTROL VALUE IS INVALID

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**Related Subroutines**

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPLMO**
Set Lighting Calculation Mode

**GPLSS**
Set Light Source State

**GPPGC**
Set Polygon Culling

**GPQTDF**
Inquire Trimming Curve Display Facilities

**GPSAC**
Set Surface Approximation Criteria

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCAC**
Set Trimming Curve Approximation Criteria

**RCP code**

201345027 (X'0C004803')

# GPTS3 - Triangle Strip 3

**GPTS3 (*pflags*, *pdata*, *tflags*, *twidth*, *tdata*, *vxflags*, *vxwidth*, *vxdata*)**

### Purpose

Use **GPTS3** to insert a Triangle Strip 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Triangle Strip 3 structure element depending on the current edit mode.

When encountered during structure traversal, this element generates n - 2 triangles from *n* vertices. Each triangle is generated by vertices k, k + 1, k + 2. Your application can specify optional data to further control the rendering of this primitive. The optional data consists of:

- Normals.

  You can specify a geometric normal for each triangle and/or a normal for each vertex. The normals are used in the lighting process to produce more realistic effects.
- Vertex colors.

  You can specify color for each vertex. If the current color source attribute indicates using the color defined in the primitive, and the primitive is not to be highlighted, then the graPHIGS API uses the specified color. The colors are used in the lighting process to produce more realistic effects.
- Boundary flags.

  The edges of this primitive consist of the line segments forming the boundary of each triangle in the strip. You may specify boundary flags to identify the boundaries that are to be rendered as edges. (More boundary flags are specified than are actually used; the unused boundary flags are ignored.). Whether the edges between triangles are drawn once or twice is workstation dependent.
- Transparency coefficients.

  You can specify a transparency coefficient per vertex. The graPHIGS API uses these values when producing transparency effects for the rendered primitive.
- Vertex morphing vectors.

  You can supply vertex morphing vectors per vertex. The graPHIGS API combines these vectors with the vertices and vertex morphing scale factors (**GPVMF**) to create new vertex coordinate values for the rendered primitive.
- Data mapping data.

  You can specify data mapping data per vertex. The graPHIGS API uses these values to determine the colors of the rendered primitive.
- Data morphing vectors.

  You can specify data morphing vectors per vertex. The graPHIGS API combines these vectors with the data morphing scale factors (**GPDMF**) and (**GPBDMF**) and the vertex data mapping values to create new data mapping data values for the rendered primitive.

See *The graPHIGS Programming Interface: Understanding Concepts* for a more complete explanation of how the graPHIGS API uses the various optional data values.

> **Note:** ***This note applies ONLY to applications which will be run on the High Performance 3D Color Graphics Processor (8 or 24 bit)***. Use of any optional data other than the vertex normals, triangle normals, vertex colors, and boundary flags may cause unpredictable results (including locking the display) on this graphics processor. If only the High Performance 3D Color Graphics Processor is used, you should include only the supported optional data values. If your application must support multiple graphics processors INCLUDING this particular processor, the Inquire Workstation Description (**GPQWDT**) subroutine must be used to determine the functions that each

workstation supports. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference* for the High Performance 3D Color Graphics Processor.

When rendering this primitive, if the current edge flag attribute is set to `2=ON` and the current line type is *not* set to `1=SOLID_LINE`, then the results are unpredictable due to the potential double drawing of some edges on some workstations.

The graPHIGS API ignores triangle strips with less than three vertices.

Polygon attributes are applied to this primitive.

**GPTS3** is identified as GDP 1029.

**Note:** Not all GDPs are supported on all workstations. Use the Inquire List of Generalized Drawing Primitives (**GPQGD**) subroutine to determine the GDPs supported by an opened workstation. See also the workstation description in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*pflags* **— specified by user, fullword integer**
Shows what optional data is specified for the primitive. The value specified should be the sum of the following values based on the fields that are included in the *pdata* parameter.

| Value | Corresponding Field. |
|---|---|
| **0** | Number of vertices. |
| **4** | Count of vertex morphing vectors is specified. |
| **8** | Count of data mapping data is specified. |
| **16** | Count of data morphing vectors is specified. |

*pdata* **— specified by user, array of primitive data**
Contains specific information about the entire primitive. The presence of optional fields is determined by the value of the *pflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Number of Vertices — fullword integer (>=0)**
Number of vertices in the triangle strip. This field is required.

**Vertex Morphing Vector Count — fullword integer (>=0)**
The number of vertex morphing vectors specified at each vertex. The number of fullwords of vertex morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

**Data Mapping Data Count — fullword integer (>=0)**
The number of data mapping values specified at each vertex. The number of data mapping values added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

**Data Morphing Vector Count — fullword integer (>=0)**
The number of data morphing vectors specified at each vertex. The number of fullwords of data morphing vector data added to the other fullwords of vertex data specified per vertex cannot exceed 255 fullwords. This field is optional.

*tflags* **— specified by user, fullword integer**
Shows what optional data is specified for each triangle. The value specified should be the sum of the following values based on the fields that are included in the *tdata* parameter.

| Value | Corresponding Field |
|---|---|
| **0** | Null, *tdata* is not referenced. |
| **1** | Geometric Normals. |

*twidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *tdata* parameter array (>=0).

*tdata* **— specified by user, array of per triangle data**
Contains specific information about each triangle in the primitive. The length of this array is defined by the contents of the `number of vertices`-2. The presence of optional fields is determined by the value of the *tflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Geometric Normal — 3 short floating-point numbers (MC)**
Geometric normal to be used in processing the triangle. This field is optional.

*vxflags* **— specified by user, fullword integer**
Shows what optional data is specified for each vertex. The value specified should be the sum of the following values based on the fields that are specified in the *vxdata* parameter.

| Value | Meaning |
|---|---|
| **0** | Vertex coordinates are specified. |
| **1** | Vertex normals are specified. |
| **2** | Vertex colors are specified. |
| **4** | Boundary flags are specified. |
| **8** | Transparency coefficient is specified. |
| **16** | Vertex morphing vectors are specified. |
| **32** | Data mapping data is specified. |
| **64** | Data morphing vectors are specified (valid only if date mapping data is specified also). |

*vxwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of the *vxdata* parameter array (>=3).

*vxdata* **— specified by user, array of vertex data**
Contains specific information about each vertex in the primitive. The length of this array is equal to `number of vertices` field in the *pdata* parameter. The presence of optional fields is determined by the value of the *vxflags* parameter. The fields must be specified in the order defined below and no space is allowed between those that are present.

**Coordinates — 3 short floating-point numbers (MC)**
*x, y*, and *z* coordinates of a vertex. This field is required.

**Normal — 3 short floating-point numbers (MC)**
The three components of a vector that is to be used as the normal of the polygon at the corresponding vertex. This field is optional.

**Color — 3 short floating-point numbers**
The three components of a color in the current color model as contained in the graPHIGS API state list. This field is optional.

**Boundary Flags — 2 fullword integers**
Specifies whether the corresponding boundary is to be treated as an edge of the polygon

(1=NOT_AN_EDGE, 2=IS_AN_EDGE). The edge attributes are only applied to boundary segments that have a boundary flag set to a value of 2=IS_AN_EDGE.

Each vertex *v(i)* has two boundary flags which specify whether the boundary from the specified vertex to an adjacent vertex is to be drawn as an edge. The first boundary flag applies to the boundary from vertex *v(i)* to vertex *v(i+1)* and the second boundary flag applies to the boundary from vertex *v(i)* to vertex *v(i+2)*, where *1<=i<=number_of_vertices*. As there are more boundary flags than edges, unused boundary flags are ignored. This field is optional.

**Transparency Coefficient — short floating-point number (0.0 <= transparency coefficient <=1.0)** The transparency coefficient value used when performing transparency processing. A value of 0.0 is fully opaque; a value of 1.0 is fully transparent. This field is optional.

**Vertex Morphing Vectors — array of short floating-point numbers.**
The vertex morphing vectors $dx_1$, $dy_{\#EMPTY>1}$, $dz_1$, $dx_2$, $dy_2$, $dz_2$, ..., $dx_n$, $dy_n$, $dz_{\#EMPTY>n}$. The number, *n*, of vectors in this array is specified in the *pdata* parameter as the vertex morphing vector count. The array must be the same length for every vertex. This field is optional.

**Data Mapping Data — array of short floating-point numbers.**
The data mapping data values $x_1$, $x_{\#EMPTY>2}$, $x_3$, ..., $x_{\#EMPTY>n}$. The number, *n*, of values in this array is specified in the *pdata* parameter as the data mapping data count. The array must be the same length for every vertex. This field is optional.

**Data Morphing Vectors — array of short floating-point numbers.**
The data morphing vectors $d_{11}$>, $d_{\#EMPTY>12}$>, $d_{13}$>, ..., $d_{\#EMPTY>1n}$>, $d_{21}$>, $d_{\#EMPTY>22}$>, $d_{23}$>, ..., $d_{\#EMPTY>2n}$>, ..., $d_{m1}$, $d_{\#EMPTY>m2}$, $d_{m3}$, ..., $d_{\#EMPTY>mn}$.

The number, *n*, is specified in the *pdata* parameter as the data mapping data count, and the number, *m* is specified in the *pdata* parameter as the data morphing vector count. The array must be the same length for every vertex. This field is optional.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**100** NUMBER OF POINTS < ZERO

**115** TRANSPARENT COEFFICIENT IS INVALID

**349** NORMAL VECTOR HAS ZERO LENGTH

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**352** BOUNDARY FLAG IS INVALID

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**636** FULLWORDS OF VERTEX DATA EXCEEDS MAXIMUM OF 255

### Related Subroutines

**GPBDMF**
Set Back Data Morphing Factors

**GPBDMI**
Set Back Data Mapping Index

**GPBICD**
Set Back Interior Color Direct

**GPBICI**
Set Back Interior Color Index

**GPBISM**
Set Back Interior Shading Method

**GPBSCD**
Set Back Specular Color Direct

**GPBSCI**
Set Back Specular Color Index

**GPBSPR**
Set Back Surface Properties

**GPBTCO**
Set Back Transparency Coefficient

**GPDMF**
Set Data Morphing Factors

**GPDMI**
Set Data Mapping Index

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPISM**
Set Interior Shading Method

**GPPGC**
Set Polygon Culling

**GPSCD**
Set Specular Color Direct

**GPSCI**
Set Specular Color Index

**GPSPR**
Set Surface Properties

**GPTCO**
Set Transparency Coefficient

**GPVMF**
Set Vertex Morphing Factors

**RCP code**

201343745 (X'0C004301')

## GPTX2 - Geometric Text 2

**GPTX2 (*point*, *length*, *text*)**

**Purpose**

Use **GPTX2** to insert a two-dimensional, geometric text element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Geometric Text 2 structure element depending on the current edit mode.

This structure element specifies a string of geometric text that the graPHIGS API draws at the specified location in the *x*, *y* plane.

When you create this structure element, the current Text Character Set value in the graPHIGS API State List is bound to this character string.

**Parameters**

*point* **— specified by user, 2 short floating-point numbers (MC)**
*x* and *y* coordinates of the text position.

*length* **— specified by user, fullword integer**
Length of text string in bytes (>=0).

*text* **— specified by user, variable length character string**
Text to be displayed.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**108** NUMBER OF CHARACTERS IN TEXT STRING < ZERO

**Related Subroutines**

**GPCHH**
Set Character Height

**GPCHPM**
Set Character Positioning Mode

**GPCHSP**
Set Character Spacing

**GPCHUB**
Set Character Up and Base Vectors

**GPCHUP**
Set Character Up Vector

**GPCHXP**
Set Character Expansion Factor

**GPQGFC**
Inquire Geometric Font Characteristics

**GPQXTX**
Inquire Extended Text Facilities

**GPTXAL**
Set Text Alignment

**GPTXCD**
Set Text Color Direct

**GPTXCI**
Set Text Color Index

**GPTXFO**
Set Text Font

**GPTXI** Set Text Index

**GPTXPR**
Set Text Precision

**GPTXPT**
Set Text Path

**RCP code**

201328134 (X'0C000606')

## GPTX3 - Geometric Text 3

**GPTX3 (*point*, *length*, *text*, *refv1*, *refv2*)**

**Purpose**

Use **GPTX3** to insert a three-dimensional, geometric text element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Geometric Text 3 structure element depending on the current edit mode.

This structure element specifies a string of geometric text that the graPHIGS API draws on the plane defined by the specified text position and reference vectors.

When you create this structure element, the current Text Character Set value in the graPHIGS API State List is bound to this character string.

Two vectors definitions orient a local coordinate system, within which the text is positioned. The two reference vectors and the text position define the plane in which the text is drawn. The first vector defines the x-axis of the local coordinate system. The second reference vector defines the half plane of the text in which the positive y-axis lies. The directions specified by Character Up Vector and Text Path attributes are relative to this coordinate system.

**Parameters**

*point* **— specified by user, 3 short floating-point numbers (MC)**
*x*, *y*, and *z* coordinates of the text position. This position serves as the origin of the local coordinate system within which three-dimensional text is defined.

*length* **— specified by user, fullword integer**
Length of text string in bytes (>=0).

*text* **— specified by user, variable length character string**
Text to be displayed.

*refv1* **— specified by user, 3 short floating-point numbers (MC)**
Directional components of the first text reference vector.

*refv2* **— specified by user, 3 short floating-point numbers (MC)**
Directional components of the second reference vector.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**107** REFERENCE VECTORS ARE COLINEAR

**108** NUMBER OF CHARACTERS IN TEXT STRING < ZERO

**Related Subroutines**

**GPCHH**
Set Character Height

**GPCHPM**
Set Character Positioning Mode

**GPCHSP**
Set Character Spacing

**GPCHUB**
Set Character Up and Base Vectors

**GPCHUP**
Set Character Up Vector

**GPCHXP**
Set Character Expansion Factor

**GPQGFC**
Inquire Geometric Font Characteristics

**GPQXTX**
Inquire Extended Text Facilities

**GPTXAL**
Set Text Alignment

**GPTXCD**
Set Text Color Direct

**GPTXCI**
Set Text Color Index

**GPTXFO**
Set Text Font

**GPTXI** Set Text Index

**GPTXPR**
Set Text Precision

**GPTXPT**
Set Text Path

**RCP code**

201328133 (X'0C000605')

# Chapter 4. Attribute Structure Elements

Attribute values describe the appearance of output primitives, including size, shape, style, and color.

This group of subroutines creates structure elements and requires that the structure state is Structure Open (STOP). When the graPHIGS API encounters the elements in this section at traversal time, it modifies the current traversal time registers.

Your application can specify some attribute values directly through a structure element or indirectly by using an index to a bundle table in the Workstation State List (WSL). During structure traversal, the current Attribute Source Flag (ASF) setting determines whether the graPHIGS API draws a primitive using an individual or bundled value of an attribute. For more information about attributes, see *The graPHIGS Programming Interface: Understanding Concepts*.

For attribute values supported on a specific workstation, use the appropriate Inquiry programming subroutines or see *The graPHIGS Programming Interface: Technical Reference*.

## GPAAL - Set Annotation Alignment

**GPAAL (*horiz*, *vert*)**

### Purpose

Use **GPAAL** to insert a Set Annotation Alignment structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Annotation Alignment structure element depending on the current edit mode.

This structure element specifies the alignment the graPHIGS API uses at structure traversal time to render all subsequent annotation text primitives.

The alignment values affect the position of the annotation text extent rectangle in relation to the text position.

The traversal default for annotation alignment is `1=NORMAL`. If the specified alignment values are not within the allowable range, then the graPHIGS API uses a default alignment of `1=NORMAL` for both horizontal and vertical.

For more information concerning annotation text and annotation text attributes, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*horiz* **— specified by user, fullword integer**
Horizontal alignment (`1=NORMAL, 2=LEFT_ALIGN, 3=CENTER, 4=RIGHT_ALIGN`)

*vert* **— specified by user, fullword integer**
Vertical alignment (`1=NORMAL, 2=TOP, 3=CAP, 4=HALF, 5=BASE, 6=BOTTOM`)

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **309** | TEXT ALIGNMENT COMPONENT IS INVALID |

### Related Subroutines

**GPTXPR** Set Text Precision

### RCP code

201328659 (X'0C000813')

## GPADCN - Add Class Name to Set

**GPADCN (*number*, *names*)**

### Purpose

Use **GPADCN** to insert an Add Class Name to Set structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Add Class Name to Set structure element depending on the current edit mode.

During structure traversal, this structure element adds the specified class names to the current class set. The traversal default is a null name set.

Class names let an application control the eligibility of a primitive for pickability (detectability), highlighting, and invisibility by associating the primitive with a class set.

When the graPHIGS API encounters a primitive during structure traversal, the primitive belongs to the classes contained in the current class set. If the workstation does not support a specified name, then the graPHIGS API ignores the name and the name has no affect on the primitive.

Also use class names to create inclusion and exclusion filters for the specified workstation. The graPHIGS API uses these filters in conjunction with the class set traversal state to determine if pickability, highlighting, and visibility apply. The filters act independently of each other. During structure traversal, the graPHIGS API compares the current class set to the current filters.

For a complete discussion of class names and filters, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*number* **— specified by user, fullword integer**
Number of class names to add to the class set (>=0)

*names* **— specified by user, array of fullword integers**
Array of class names to add to the class set (class names must be >=0)

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **320** | CLASS NAME VALUE IS INVALID |
| **530** | NUMBER OF CLASS NAMES < ZERO |

### Related Subroutines

**GPHLF** Set Highlighting Filter

**GPIVF** Set Invisibility Filter
**GPPKF** Set Pick Filter
**GPQNCN** Inquire Number of Available Class Names
**GPRCN** Remove Class Name from Set

### RCP code

201334785 (X'0C002001')

## GPAH - Set Annotation Height

**GPAH (*height*)**

### Purpose

Use **GPAH** to insert a Set Annotation Height structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Annotation Height structure element depending on the current edit mode.

The application specifies the annotation text character height with respect to the annotation text local coordinate system; that is, a two-dimensional coordinate system parallel to the NPC (Normalized Projection Coordinates) *x*, *y* plane. The graPHIGS API multiplies the specified height by the scale factor of the current workstation transformation, and then maps the result to the closest available height on the workstation.

The current annotation character height entry in the graPHIGS API traversal state list is also set by the Set Annotation Height Scale Factor (**GPAHSC**) structure element. As a result, the traversal default annotation height corresponds to a default annotation height scale factor of 1.0.

For more information concerning annotation text and annotation text attributes, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*height* **— specified by user, short floating-point number (NPC)**
Annotation text character height (>0)

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **78** | CHARACTER HEIGHT VALUE <= ZERO |

### Related Subroutines

**GPAHSC** Set Annotation Height Scale Factor
**GPQXAF** Inquire Extended Annotation Font Characteristics

### RCP code

201328660 (X'0C000814')

## GPAHSC - Set Annotation Height Scale Factor

**GPAHSC (*factor*)**

### Purpose

Use **GPAHSC** to insert a Set Annotation Height Scale Factor structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Annotation Height Scale Factor structure element depending on the current edit mode.

The specified annotation height scale factor specifies a ratio of the annotation character height to the workstation's nominal character height. Because the current annotation text character height in the graPHIGS API traversal state list is expressed in Device Coordinates (DC), the graPHIGS API multiplies the specified value by the nominal character height and sets the specified value to the entry.

The current annotation character height entry in the graPHIGS API traversal state list is also set by the Set Annotation Height (**GPAH**) structure element.

The traversal default value for annotation height scale factor is 1.0.

For more information concerning annotation text and annotation text attributes, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*factor* **— specified by user, short floating-point number**
This attribute specifies the height scale factor. This factor is multiplied by the workstation dependent nominal annotation text character height.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |

### Related Subroutines

| | |
|---|---|
| **GPAH** | Set Annotation Height |
| **GPQXAF** | Inquire Extended Annotation Font Characteristics |

### RCP code

201328655 (X'0C00080F')

## GPAID - Set Antialiasing Identifier

**GPAID (*antid*)**

### Purpose

Use **GPAID** to insert a Set Antialiasing Identifier structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Antialiasing Identifier structure element depending on the current edit mode.

During structure traversal, the graPHIGS API sets the antialiasing identifier entry of graPHIGS API traversal state list to the value specified by the parameter. To create subsequent output primitives, use this value. This value supplies antialiasing information to the workstation.

The traversal default for the antialiasing identifier is 1=NONE.

If the workstation does not support the specified antialiasing identifier, then the antialiasing identifier defaults to 1=NONE.

### Parameters

*antid* **— specified by user, fullword integer**
Antialiasing identifier (1=NONE, 2=PERFORM)

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **252** | ANTIALIASING IDENTIFIER IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPQAMO** | Inquire Available Antialiasing Modes |
| **GPQCVR** | Inquire Current View Representation |
| **GPQRVR** | Inquire Requested View Representation |
| **GPXVR** | Set Extended View Representation |

### RCP code

201343242 (X'0C00410A')

## GPAPT - Set Annotation Path

**GPAPT (*path*)**

### Purpose

Use **GPAPT** to insert a Set Annotation Path structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Annotation Path structure element depending on the current edit mode.

This structure element specifies the writing direction of characters in a text string relative to the Annotation Up vector. At structure traversal time, the graPHIGS API uses this path value to render all subsequent annotation text primitives.

The traversal default for annotation path is 1=RIGHT.

If the workstation does not support the specified path value, then the annotation path value defaults to 1=RIGHT.

For more information concerning annotation text and annotation text attributes, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*path* — **specified by user, fullword integer**
Specifies the path for annotation text primitives (1=RIGHT, 2=LEFT, 3=UP, 4=DOWN).

**Error Codes**

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **306** | TEXT PATH VALUE IS INVALID |

**Related Subroutines**

| | |
|---|---|
| **GPTXPR** | Set Text Precision |

**RCP code**

201328667 (X'0C00081B')

## GPAS - Set Annotation Style

**GPAS (*style*)**

**Purpose**

Use **GPAS** to insert a Set Annotation Style structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Annotation Style structure element depending on the current edit mode.

During structure traversal, this structure element sets the current annotation style entry in the graPHIGS API traversal state list to the value specified by the annotation style (*style*) parameter to render subsequent annotation text primitives. For annotation style 2=LEAD_LINE, the current polyline attributes are used to render the lead line.

The traversal default for annotation style is 1=UNCONNECTED.

If the workstation does not support the specified annotation style, then the annotation style defaults to 1=UNCONNECTED.

**Parameters**

*style* — **specified by user, fullword integer**
Annotation style (1=UNCONNECTED, 2=LEAD_LINE).

**Error Codes**

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **82** | ANNOTATION STYLE IS INVALID |

**Related Subroutines**

| | |
|---|---|
| **GPANR2** | Annotation Text Relative 2 |
| **GPANR3** | Annotation Text Relative 3 |

**RCP code**

201328668 (X'0C00081C')

## GPASF - Attribute Source Flag Setting

**GPASF (*number*, *id*, *flag*)**

### Purpose

Use **GPASF** to insert a Set Attribute Source Flag (ASF) structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Attribute Source Flag (ASF) structure element depending on the current edit mode.

At structure traversal time, the current ASF setting determines the `1=BUNDLED` or `2=INDIVIDUAL` attributes that the graPHIGS API uses to draw an output primitive.

The traversal default for all attributes is `2=INDIVIDUAL`.

If any attribute identifier in the list is invalid, then the graPHIGS API ignores that entry. If any attribute source flag is invalid, then it defaults to `2=INDIVIDUAL`.

### Parameters

*number* **— specified by user, fullword integer**
The number of entries in the lists (>=0).

*id* **— specified by user, array of fullword integers**
List of attribute identifiers. Valid values are:

| | |
|---|---|
| 1 | =Linewidth scale factor |
| 3 | =Polyline color |
| 4 | =Marker type |
| 5 | =Marker size scale factor |
| 6 | =Polymarker color |
| 7 | =Text font |
| 8 | =Text precision |
| 9 | =Character expansion factor |
| 11 | =Text color |
| 12 | =Interior style |
| 13 | =Style index |
| 14 | =Interior color |
| 15 | =Edge flag |
| 16 | =Edge linetype |
| 17 | =Edge color |
| 18 | =Edge scale factor |

*flag* **— specified by user, array of fullword integers**
Attribute source flag list corresponding to the attribute identifiers (`1=BUNDLED`, `2=INDIVIDUAL`).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **314** | AN ATTRIBUTE IDENTIFIER IS INVALID |
| **315** | ATTRIBUTE SOURCE IS INVALID |

| | |
|---|---|
| **523** | NUMBER OF ASFS < ZERO |

### Related Subroutines

None

### RCP code

201329153 (X'0C000A01')

## GPAUP - Set Annotation Up Vector

**GPAUP (*vector*)**

### Purpose

Use **GPAUP** to insert a Set Annotation Up Vector structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Annotation Up Vector structure element depending on the current edit mode.

At structure traversal time, this structure element specifies the y-axis direction for characters in a text string that the graPHIGS API uses to render all subsequent annotation text primitives. When rendering annotation text primitives, the graPHIGS API uses the annotation up vector along with a default annotation base vector set at right angles in the clockwise direction to the annotation up vector.

The traversal default value for annotation up vector is 0.0, 1.0, and for annotation base vector the traversal default is 1.0, 0.0.

If the annotation up vector is invalid, then a vector value defaults to a value of 0.0, 1.0, and a base vector value of 1.0, 0.0.

The graPHIGS API normalizes the specified vectors. If the application later inquires the content of this structure element, then the graPHIGS API returns the normalized vector, *not* the original vector specified by this subroutine.

For more information concerning annotation text and annotation text attributes, see *The graPHIGS Programming Interface: Subroutine Reference*.

### Parameters

*vector* **— specified by user, 2 short floating-point numbers (NPC)**
Directional components of the annotation text character up vector. Magnitude must be greater than zero.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **79** | CHARACTER UP VECTOR HAS LENGTH ZERO |

### Related Subroutines

| | |
|---|---|
| **GPTXPR** | Set Text Precision |

**RCP code**

201328666 (X'0C00081A')

# GPBBLF - Set Back Blending Function

**GPBBLF (*srcf, destf*)**

### Purpose

Use **GPBBLF** to insert a Set Back Blending Function structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Back Blending Function structure element, depending upon the current edit mode.

Blending is a method of combining the colors of the primitive being rendered (called the *source*) with previously rendered output (called the *destination*) to create a new color for the destination. The blending function specifies how you combine the source and destination colors. Conceptually:

*COLOR = ( srcf x* $COLOR_{src}$ *) + ( destf x* $COLOR_{dest}$ *)*

where *srcf* and *destf* are the source and destination blending functions respectively.

This element specifies the source blending function (*srcf*) and the destination blending function (*destf*) you use when you blend subsequent primitives with previously rendered output. Blending occurs when the Transparency Processing Mode of the view table entry is `3=BLEND` or `4=BLEND_ALL`.

The traversal default for *srcf* is `3=SRCBF_SRC_ALPHA` and for *destf* is `4=DSTBF_ONE_MINUS_SRC_ALPHA`.

This element specifies the methods you use to control the blending calculations performed on back facing portions of subsequent area primitives. The graPHIGS API uses the specified methods if the face distinguish mode (**GPFDMO**) is set to `2=COLOR_SURFACE_PROPERTIES`.

The source and destination blending functions are selected by your application using the *srcf* and *destf* parameters for **GPBBLF** as indicated in the following table:

| Source function ID | *srcf* | Destination function ID | *destf* |
|---|---|---|---|
| 1 | 0 | 1 | 0 |
| 2 | 1 | 2 | 1 |
| 3 | $alpha_{src}$ | 3 | $alpha_{src}$ |
| 4 | 1 - $alpha_{src}$ | 4 | 1 - $alpha_{src}$ |
| 5 | $alpha_{dest}$ | 5 | $alpha_{dest}$ |
| 6 | 1 - $alpha_{dest}$ | 6 | 1 - $alpha_{dest}$ |
| 7 | $R_{dest}$, $G_{dest}$, $B_{dest}$, or $alpha_{dest}$ | 7 | $R_{src}$, $G_{src}$, $B_{src}$, or $alpha_{src}$ |
| 8 | 1 - ($R_{dest}$, $G_{dest}$, $B_{dest}$, or $alpha_{dest}$) | 8 | 1 - ($R_{src}$, $G_{src}$, $B_{src}$, or $alpha_{src}$) |
| 9 | min ($alpha_{src}$, 1 - $alpha_{dest}$) | | |

For example, if *srcf* has the value `2=SRCBF_ONE` and *destf* has the value `1=DSTBF_ZERO`, then the blending function becomes:

*COLOR = (1.0 x* $COLOR_{src}$*) + ( 0.0 x* $COLOR_{dest}$*)* which causes the source color to replace the destination color.

In the table blending coefficients, also called *alpha values* (alpha), are floating-point numbers in the range [0.0,1.0] where 0.0 is fully transparent and 1.0 is fully opaque (opposite of the definition of a transparency coefficient). The source blending coefficient ($alpha_{src}$ ) is calculated from the source transparency coefficient (*coeff*) as follows:

$alpha_{src}$ = 1.0 - *coeff*

where coeff is in the range [0.0,1.0] and is specified by the Set Back Transparency Coefficient procedure (**GPBTCO**) or by the transparency coefficient of the Set Back Surface Properties subroutine (**GPBSPR**). (These two subroutines set the same transparency coefficient). You can also specify the transparency coefficient as part of the vertex information of certain primitives. Alternately, you can directly supply alpha values and transparency coefficients as part of a texture map. Where available, the alpha buffer provides the destination blending coefficient ($alpha_{dest}$) defined by some of the blending functions. If a blending function is not supported by the workstation, or if a destination blending function is specified on a workstation that does not have alpha buffers, then the element is ignored. Use **GPQWDT** to inquire the transparency facilities of a specified workstation.

A new blending coefficient is calculated from the blending equation and stored with each color (pixel) on those workstations which support alpha buffers. Those workstations which do not support alpha buffers do not support blending functions from the table which include $alpha_{dest}$. The $alpha_{dest}$ value is typically stored as an integer in the range [0,255]. The initial $alpha_{dest}$ value for a view's shield can be specified using the Set Extended View Representation (**GPXVR**).

### Parameters

*srcf*— **specified by user, fullword integer**
Source blending function (1=SRCBF_ZERO, 2=SRCBF_ONE, 3=SRCBF_SRC_ALPHA, 4=SRCBF_ONE_MINUS_SRC_ALPHA, 5=SRCBF_DST_ALPHA, 6=SRCBF_ONE_MINUS_DST_ALPHA, 7=SRCBF_DST_COLOR, 8=SRCBF_ONE_MINUS_DST_COLOR, 9=SRCBF_MIN_SRC_ALPHA_ONE_MINUS_DST_ALPHA).

*destf*— **specified by user, fullword integer**
Destination blending function (1=DSTBF_ZERO, 2=DSTBF_ONE, 3=DSTBF_SRC_ALPHA, 4=DSTBF_ONE_MINUS_SRC_ALPHA, 5=DSTBF_DST_ALPHA, 6=DSTBF_ONE_MINUS_DST_ALPHA, 7=DSTBF_SRC_COLOR, 8=DSTBF_ONE_MINUS_SRC_COLOR).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **629** | BLENDING FUNCTION IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPBLF** | Set Blending Function |
| **GPBSPR** | Set Back Surface Properties |
| **GPBTCO** | Set Back Transparency Coefficient |
| **GPQWDT** | Inquire Workstation Description |
| **GPSPR** | Set Surface Properties |
| **GPTCO** | Set Transparency Coefficient |
| **GPXVR** | Set Extended View Representation |

### RCP code

# GPBDFM - Set Back Data Filtering Method

**GPBDFM (*minfm, magfm, boundu, boundv*)**

### Purpose

Use **GPBDFM** to insert a Set Back Data Filtering Method structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Data Filtering Method structure element, depending upon the current edit mode.

This element specifies the back data filtering method that the graPHIGS API uses to perform data mapping on back facing portions of subsequent area primitives.

If the face distinguish mode (**GPFDMO**) is `1=NONE`, then the graPHIGS API uses the current data filtering method for data mapping on the front and back facing portions of area primitives. If the face distinguish mode is `2=COLOR_SURFACE_PROPERTIES`, then the graPHIGS API uses the specified back data filtering method for data mapping on only back facing portions of area primitives. When the data mapping method is `2=SINGLE_VALUE_UNIFORM`, the graPHIGS API ignores the *boundv* parameter.

The traversal default back data filtering methods are:

| | |
|---|---|
| **Minification:** | `1=SAMPLE_IN_BASE` |
| **Magnification:** | `1=SAMPLE_IN_BASE` |
| **U-dimension:** | `1=CLAMP` |
| **V-dimension:** | `1=CLAMP` |

If any specified value is not supported on the workstation, then the default value is used.

### Parameters

*minfm* **— specified by user, fullword integer**
Minification filtering method (`1=SAMPLE_IN_BASE, 2=INTERP_IN_BASE, 3=SAMPLE_IN_SQUARE_MM, 4=SAMPLE_IN_AND_INTERP_BTWN_SQUARE_MM, 5=INTERP_IN_SQUARE_MM, 6=INTERP_IN_AND_BTWN_SQUARE_MM, 7=SAMPLE_IN_RECT_MM, 8=SAMPLE_IN_AND_INTERP_BTWN_RECT_MM, 9=INTERP_IN_RECT_MM`).

*magfm* **— specified by user, fullword integer**
Magnification filtering method (`1=SAMPLE_IN_BASE, 2=INTERP_IN_BASE`).

*boundu* **— specified by user, fullword integer**
U-dimension bounding method (`1=CLAMP, 2=REPEAT`).

*boundv* **— specified by user, fullword integer**
V-dimension bounding method (`1=CLAMP, 2=REPEAT`).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **631** | FILTERING METHOD IS INVALID |
| **632** | BOUNDING METHOD IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPDFM** | Set Data Filtering Method |
| **GPDMR** | Set Data Mapping Representation |

### RCP code

201343516 (X'0C00421C')

## GPBDMI - Set Back Data Mapping Index

*GPBDMI (index*)

### Purpose

Use **GPBDMI** to insert a Set Back Data Mapping Index structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Back Data Mapping Index structure element, depending upon the edit mode.

During traversal, the back data mapping index specifies the entry in the data mapping table used to perform data mapping on subsequent primitives that support data mapping.

If the face distinguish mode (**GPFDMO**) is `1=NONE`, then the graPHIGS API uses the current data mapping index for data mapping on the front and back facing portions of area primitives. If the face distinguish mode is `2=COLOR_SURFACE_PROPERTIES`, then the graPHIGS API uses the specified back data mapping index for data mapping on only back facing portions of area primitives.

The data mapping table is zero based. Entry zero always contains a data mapping entry of `1=DM_METHOD_COLOR` Use **GPQWDT** to determine the number of definable data mapping table entries.

The traversal default back data mapping index is zero. If the specified index is not supported on the workstation, then the graPHIGS API uses a default index of zero.

### Parameters

*index* **— specified by user, fullword integer**
Data mapping table index (>=0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **630** | DATA MAPPING INDEX < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPDMI** | Set Data Mapping Index |
| **GPDMR** | Set Data Mapping Representation |
| **GPQWDT** | Inquire Workstation Description |

### RCP code

201343514 (X'0C00421A')

## GPBDM2 - Set Back Data Matrix 2

**GPBDM2 (*matrix*)**

### Purpose

Use **GPBDM2** to insert a Set Back Data Matrix 2 structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Back Data Matrix 2 structure element, depending upon the current edit mode.

This element specifies the transformation matrix that the graPHIGS API uses to modify the data mapping values on back facing portions of primitives that support data mapping. The specified matrix is used if the face distinguish mode (**GPFDMO**) is `2=COLOR_SURFACE_PROPERTIES`.

The data mapping values are treated as homogeneous points of the form (*u, v,* 1.0). When used with a data mapping method of `2=SINGLE_VALUE_UNIFORM`, the graPHIGS API sets the *v* coordinate value to 0.0. Otherwise, the graPHIGS API uses indexed vertex data values for the *u* and *v* coordinate values. This point is multiplied by the specified matrix. The resulting values are used to perform the data mapping.

The last column of the matrix must have the values 0.0, 0.0, 1.0.

The traversal default back data matrix is the identity matrix.

### Parameters

*matrix* **— specified by user, 9 short floating-point numbers**
Back data modification matrix.

The elements of the matrix must be specified as follows:

```
|m11  M12  0.0|
|m21  M22  0.0| ---> (m11, m12, 0.0, m21, m22, 0.0, m31, m32, 1.0)
|m31  M32  1.0|
```

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **633** | MATRIX VALUE IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPDM2** | Set Data Matrix 2 |
| **GPDMR** | Set Data Mapping Representation |

### RCP code

201343518 (X'0C00421E')

## GPBICD - Set Back Interior Color Direct

**GPBICD (*color*)**

### Purpose

Use **GPBICD** to insert a Set Back Interior Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Back Interior Color Direct structure element depending on the current edit mode.

This structure element specifies direct color values. During structure traversal, the graPHIGS API uses these values to render the back facing portions of all following area definitions if the interior style is set to `1=HOLLOW` (see GPIS - Set Interior Style) and the edge flag is set to `1=OFF` (see GPEF - Set Edge Flag) or if the interior style is set to `1=HOLLOW` the interior style is set to `2=SOLID` or `4=HATCH`.

Face distinguish mode must be set to `2=COLOR_SURFACE_PROPERTIES` If the face distinguish mode is `1=NONE`, then the graPHIGS API uses the current interior color to render the back facing portions of all following area definitions.

This attribute sets the same traversal state as the Set Back Interior Color Index (**GPBICI**) subroutine. The traversal default for back interior color is the content of entry 1 of the rendering color table.

### Parameters

*color* **— specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **96** | COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL |

### Related Subroutines

| | |
|---|---|
| **GPBICI** | Set Back Interior Color Index |
| **GPEF** | Set Edge Flag |
| **GPFDMO** | Set Face Distinguish Mode |
| **GPICD** | Set Interior Color Direct |
| **GPICI** | Set Interior Color Index |
| **GPIS** | Set Interior Style |

### RCP code

201343499 (X'0C00420B')

## GPBICI - Set Back Interior Color Index

**GPBICI (*index*)**

### Purpose

Use **GPBICI** to insert a Set Back Interior Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Back Interior Color Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that defines the color values. During structure traversal, the graPHIGS API uses these values to render the back facing portions of all following area definitions if the interior style is set to 1=HOLLOW and the edge flag is set to 1=OFF or if the interior style is 2=SOLID or 4=HATCH.

Face distinguish mode must be set to 2=COLOR_SURFACE_PROPERTIES If the face distinguish mode is 1=NONE, then the graPHIGS API uses the current interior color to render the back facing portions of all following area definitions.

This attribute sets the same traversal state as the Set Back Interior Color Direct (**GPBICD**) subroutine. The traversal default value for back interior color is an index value of 1.

If the workstation does not support the specified back interior color index value, then the back interior color index defaults to a value of 1.

### Parameters

*index* **— specified by user, fullword integer**
Color index (>=0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **92** | COLOR INDEX < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPBICD** | Set Back Interior Color Direct |
| **GPEF** | Set Edge Flag |
| **GPFDMO** | Set Face Distinguish Mode |
| **GPICD** | Set Interior Color Direct |
| **GPICI** | Set Interior Color Index |
| **GPIS** | Set Interior Style |

### RCP code

201343498 (X'0C00420A')

## GPBISM - Set Back Interior Shading Method

**GPBISM (*method*)**

### Purpose

Use **GPBISM** to insert a Set Back Interior Shading Method structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Back Interior Shading Method element, depending on the current edit mode.

The graPHIGS API uses this element to specify the shading method used for the interior of back facing portions of subsequent area primitives.

This element specifies the way to shade the interior of back facing portions of subsequent area primitives. The specified method is used if the face distinguish mode (**GPFDMO**) is set to

2=COLOR_SURFACE_PROPERTIES. The interior shading methods include 1=SHADING_NONE, which is also known as *flat* shading, 2=SHADING_COLOR, traditionally known as *Gourand* shading, and 3=SHADING_DATA. See *The graPHIGS Programming Interface: Understanding Concepts* for information on the interactions between lighting, shading and data mapping.

The traversal default for interior shading method is 2=SHADING_COLOR. If the workstation does not support the specified shading method, then the graPHIGS API uses 2=SHADING_COLOR.

### Parameters

*method* **— specified by user, fullword integer**
Interior shading method (1=SHADING_NONE, 2=SHADING_COLOR, 3=SHADING_DATA).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **512** | METHOD NOT SUPPORTED |

### Related Subroutines

| | |
|---|---|
| **GPISM** | Set Interior Shading Method |
| **GPQAAF** | Inquire Advanced Attribute Facilities |

### RCP code

201343512 (X'0C004218')

## GPBLF - Set Blending Function

**GPBLF (*srcf, destf*)**

### Purpose

Use **GPBLF** to insert a Set Blending Function structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Blending Function structure element, depending upon the current edit mode.

Blending is a method of combining the colors of the primitive being rendered (called the *source*) with previously rendered output (called the *destination*) to create a new color for the destination. The blending function specifies how you combine the source and destination colors. Conceptually:

$COLOR = ( srcf \times COLOR_{src} ) + ( destf \times COLOR_{dest} )$

where *srcf* and *destf* are the source and destination blending functions respectively.

This element specifies the source blending function (*srcf*) and the destination blending function (*destf*) you use when you blend subsequent primitives with previously rendered output. Blending occurs when the Transparency Processing Mode of the view table entry is 3=BLEND or 4=BLEND_ALL.

The traversal default for *srcf* is 3=SRCBF_SRC_ALPHA and for *destf* is 4=DSTBF_ONE_MINUS_SRC_ALPHA.

If face distinguish mode is 1=NONE, then use these values to calculate the blending effects on both front and back facing portions of area primitives. If face distinguish mode is 2=COLOR_SURFACE_PROPERTIES, then use these values to calculate the blending effects on only front facing portions of area primitives.

The source and destination blending functions are selected by your application using the *srcf* and *destf* parameters for **GPBLF** as indicated in the following table:

| Source function ID | *srcf* | Destination function ID | *destf* |
|---|---|---|---|
| 1 | 0 | 1 | 0 |
| 2 | 1 | 2 | 1 |
| 3 | $alpha_{src}$ | 3 | $alpha_{src}$ |
| 4 | 1 - $alpha_{src}$ | 4 | 1 - $alpha_{src}$ |
| 5 | $alpha_{dest}$ | 5 | $alpha_{dest}$ |
| 6 | 1 - $alpha_{dest}$ | 6 | 1 - $alpha_{dest}$ |
| 7 | $R_{dest}$, $G_{\#EMPTY>dest}$, $B_{dest}$, or $alpha_{\#EMPTY>dest}$ | 7 | $R_{src}$, $G_{\#EMPTY>src}$, $B_{src}$, or $alpha_{\#EMPTY>src}$ |
| 8 | 1 - ( $R_{dest}$, $G_{\#EMPTY>dest}$, $B_{dest}$, or $alpha_{\#EMPTY>dest}$ ) | 8 | 1 - ( $R_{src}$, $G_{\#EMPTY>src}$, $B_{src}$, or $alpha_{\#EMPTY>src}$ ) |
| 9 | min ( $alpha_{src}$, 1 - $alpha_{dest}$) | | |

For example, if *srcf* has the value 2=SRCBF_ONE and *destf* has the value 1=DSTBF_ZERO, then the blending function becomes:

*COLOR* = ( 1.0 x $COLOR_{src}$ ) + ( 0.0 x $COLOR_{dest}$ )

which causes the source color to replace the destination color.

In the table the blending coefficients, also called *alpha values* (alpha), are floating point numbers in the range [0.0,1.0] where 0.0 is fully transparent and 1.0 is fully opaque (opposite of the definition of a transparency coefficient). The source blending coefficient ($alpha_{\#EMPTY>src}$) is calculated from the source transparency coefficient (*coeff*) as follows:

$alpha_{src}$ = 1.0 - coeff

where coeff is in the range [0.0,1.0] and is specified by the Set Transparency Coefficient procedure (**GPTCO**), or by the transparency coefficient of the Set Surface Properties subroutine (**GPSPR**). (These two subroutines set the same transparency coefficient). You can also specify the transparency coefficient as part of the vertex information of certain primitives. Alternately, you can directly supply alpha values and transparency coefficients as part of a texture map. Where available, the alpha buffer provides the destination blending coefficient ($alpha_{\#EMPTY>dest}$) defined by some of the blending functions. If a blending function is not supported by the workstation, or if a destination blending function is specified on a workstation that does not have alpha buffers, then the element is ignored. Use **GPQWDT** to inquire the transparency facilities of a specific workstation.

A new blending coefficient is calculated from the blending equation and stored with each color (pixel) on those workstations which support alpha buffers. Those workstations which do not support alpha buffers do not support blending functions which include $alpha_{dest}$. The $alpha_{dest}$ value is typically stored as an integer in the range [0,255]. The initial $alpha_{dest}$ value for a view's shield can be specified using the Set Extended View Representation (**GPXVR**).

**Parameters**

*srcf*— **specified by user, fullword integer**
Source blending function (1=SRCBF_ZERO, 2=SRCBF_ONE, 3=SRCBF_SRC_ALPHA, 4=SRCBF_ONE_MINUS_SRC_ALPHA, 5=SRCBF_DST_ALPHA, 6=SRCBF_ONE_MINUS_DST_ALPHA, 7=SRCBF_DST_COLOR, 8=SRCBF_ONE_MINUS_DST_COLOR, 9=SRCBF_MIN_SRC_ALPHA_ONE_MINUS_DST_ALPHA).

*destf*— **specified by user, fullword integer**
Destination blending function (1=DSTBF_ZERO, 2=DSTBF_ONE, 3=DSTBF_SRC_ALPHA, 4=DSTBF_ONE_MINUS_SRC_ALPHA, 5=DSTBF_DST_ALPHA, 6=DSTBF_ONE_MINUS_DST_ALPHA, 7=DSTBF_SRC_COLOR, 8=DSTBF_ONE_MINUS_SRC_COLOR).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **629** | BLENDING FUNCTION IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPBBLF** | Set Back Blending Function |
| **GPBSPR** | Set Back Surface Properties |
| **GPBTCO** | Set Back Transparency Coefficient |
| **GPQWDT** | Inquire Workstation Description |
| **GPSPR** | Set Surface Properties |
| **GPTCO** | Set Transparency Coefficient |
| **GPXVR** | Set Extended View Representation |

### RCP code

201343507 (X'0C004213')

## GPBRMO - Set Back Reflectance Model

**GPBRMO (*model*)**

### Purpose

Use **GPBRMO** to insert a Set Back Reflectance Model structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Reflectance Model structure element, depending on the current edit mode.

This element specifies the lighting reflectance model that the graPHIGS API uses when performing lighting calculations on back facing portions of subsequent area primitives. If the face distinguish mode (**GPFDMO**) is set to 2=COLOR_SURFACE_PROPERTIES, then the graPHIGS API uses the specified back reflectance model.

The defined back reflectance models and their effect are as follows:

| | |
|---|---|
| **1=REFLECTANCE_NONE** | No reflectance calculation is performed. |
| **2=AMB** | Ambient reflectance effects are computed. |
| **3=AMB_DIFF** | Ambient and diffuse reflectance effects are computed. |
| **4=AMB_DIFF_SPEC** | Ambient, diffuse, and specular reflectance effects are computed. |

The traversal default for the reflectance model is 1=REFLECTANCE_NONE. If the workstation does not support the specified model, then the graPHIGS API uses 1=REFLECTANCE_NONE.

### Parameters

*model* **— specified by user, fullword integer**
Back reflectance model (1=REFLECTANCE_NONE, 2=AMB, 3=AMB_DIFF, 4=AMB_DIFF_SPEC).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **110** | REFLECTANCE MODEL IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPQAAF** | Inquire Advanced Attribute Facilities |
| **GPRMO** | Set Reflectance Model |

### RCP code

201343510 (X'0C004216')

## GPBSCD - Set Back Specular Color Direct

**GPBSCD (*color*)**

### Purpose

Use **GPBSCD** to insert a Set Back Specular Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Back Specular Color Direct structure element depending on the current edit mode.

This structure element specifies the color values that the graPHIGS API uses for the specular highlights at structure traversal time. The specular highlights are produced by lighting calculations in area geometries.

Face distinguish mode (**GPFDMO**) must be set to 2=COLOR_SURFACE_PROPERTIES. If the face distinguish mode is 1=NONE, then the graPHIGS API uses the current specular color in lighting calculations for back facing portions of subsequent surfaces.

This attribute sets the same traversal state as the Set Back Specular Color Index (**GPBSCI**) subroutine. The traversal default for back specular color is the content of entry 1 of the rendering color table.

### Parameters

*color* **— specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **96** | COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL |

### Related Subroutines

| | |
|---|---|
| **GPBSCI** | Set Back Specular Color Index |
| **GPFDMO** | Set Face Distinguish Mode |

### RCP code

201343501 (X'0C00420D')

## GPBSCI - Set Back Specular Color Index

**GPBSCI (*index*)**

### Purpose

Use **GPBSCI** to insert a Set Back Specular Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Back Specular Color Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that defines the color values. At structure traversal time, the graPHIGS API uses these values for the specular highlights. The specular highlights are produced by lighting calculations in area geometries.

Face distinguish mode must be set to `2=COLOR_SURFACE_PROPERTIES`. If face distinguish mode is `1=NONE`, then the graPHIGS API uses the current specular color in lighting calculations for the back facing portions of the surfaces.

This attribute sets the same traversal state as the Set Back Specular Color Direct (**GPBSCD**) subroutine. The traversal default for back specular color is a color index of 1.

If the workstation does not support the specified color index value or the specified index is outside the color table limit, then the color index defaults to a value of 1.

### Parameters

*index* **— specified by user, fullword integer**
Color index (>=0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **92** | COLOR INDEX < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPBSCD** | Set Back Specular Color Direct |
| **GPFDMO** | Set Face Distinguish Mode |

### RCP code

201343500 (X'0C00420C')

# GPBSPR - Set Back Surface Properties

**GPBSPR (*amb*, *diff*, *spec*, *exp*, *trans*)**

### Purpose

Use **GPBSPR** to insert a Set Back Surface Properties structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Back Surface Properties structure element depending on the current edit mode.

During structure traversal, this structure element specifies the coefficients and exponents used in calculating lighting and transparency effects.

Face distinguish mode (**GPFDMO**) must be `2=COLOR_SURFACE_PROPERTIES` for the graPHIGS API to use these values to calculate the lighting and transparency effects on the back facing portions of a surface. If face distinguish mode is `1=NONE`, then the graPHIGS API uses the current surface properties to calculate the effects on the back facing portions of a surface.

The traversal default coefficients and exponent for the back surface properties are as follows:

- *amb* = 1.0 (back ambient reflection coefficient)
- *diff* = 1.0 (back diffuse reflection coefficient)
- *spec* = 1.0 (back specular reflection coefficient)
- *exp* = 0.0 (back specular reflection exponent) (i.e., no specular effect)
- *trans* = 0.0 (back transparency coefficient) (i.e., opaque)

To use the specified transparency coefficient, the transparency mode for the view must be a value other than `1=NONE`. The current back transparency coefficient in the graPHIGS API Traversal State List is also set by the back transparency coefficient of the Set Back Transparency Coefficient (**GPBTCO**) subroutine.

### Parameters

*amb* **— specified by user, short floating-point number**
Back ambient reflection coefficient (0<= *amb*<=1).

*diff* **— specified by user, short floating-point number**
Back diffuse reflection coefficient (0<= *diff*<=1).

*spec* **— specified by user, short floating-point number**
Back specular reflection coefficient (0<= *spec*<=1).

*exp* **— specified by user, short floating-point number**
Back specular reflection exponent (>=0).

*trans* **— specified by user, short floating-point number**
Back transparency coefficient (0<= *trans*<=1).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **111** | AMBIENT COEFFICIENT IS INVALID |
| **112** | DIFFUSE COEFFICIENT IS INVALID |
| **113** | SPECULAR COEFFICIENT IS INVALID |
| **114** | SPECULAR EXPONENT IS INVALID |
| **115** | TRANSPARENT COEFFICIENT IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPBRMO** | Set Back Reflectance Model |
| **GPBTCO** | Set Back Transparency Coefficient |
| **GPFDMO** | Set Face Distinguish Mode |
| **GPLMO** | Set Lighting Calculation Mode |
| **GPLSR** | Set Light Source Representation |
| **GPLSS** | Set Light Source State |
| **GPSPR** | Set Surface Properties |

### RCP code

201343494 (X'0C004206')

## GPBTCO - Set Back Transparency Coefficient

**GPBTCO (*coeff*)**

### Purpose

Use **GPBTCO** to insert a Set Back Transparency Coefficient structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Back Transparency Coefficient structure element, depending upon the current edit mode.

During traversal, the value of *coeff* specifies the source transparency coefficient of subsequent back facing portions of area primitives. The graPHIGS API uses the specified coefficient if the face distinguish mode (**GPFDMO**) is set to `2=COLOR_SURFACE_PROPERTIES`. The use of the transparency coefficient depends on the Transparency Processing Mode of the view table entry.

The current back transparency coefficient in the graPHIGS API Traversal State List is also set by the back transparency coefficient of the Set Back Surface Properties (**GPBSPR**) subroutine.

The traversal default for the back transparency coefficient value is 0.0.

### Parameters

*coeff* **— specified by user, short floating-point number**
Back transparency coefficient (0.0<=*coeff*<=1.0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **115** | TRANSPARENT COEFFICIENT IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPBSPR** | Set Back Surface Properties |
| **GPFDMO** | Set Face Distinguish Mode |
| **GPSPR** | Set Surface Properties |
| **GPTCO** | Set Transparency Coefficient |

### RCP code

# GPCAC - Set Curve Approximation Criteria

**GPCAC (*criteria*, *value*)**

### Purpose

Use **GPCAC** to insert a Set Curve Approximation Criteria structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Curve Approximation Criteria structure element depending on the current edit mode.

During structure traversal, this structure element determines how the graPHIGS API tessellates curves for subsequent curve primitives. *Tessellation* is the dividing of curves into a set of line geometries before they are processed.

Depending on the criteria selected, the graPHIGS API uses the control value by itself or the control value in conjunction with the tessellation vector in the curve primitive definition, to determine how curves will be tessellated.

The traversal default for curve approximation criteria is `1=WORKSTATION_DEPENDENT` and a control value of 1.0.

If the workstation does not support the specified curve approximation criteria or if the criteria value is outside the allowable range, then the criteria defaults to `1=WORKSTATION_DEPENDENT` and the control value defaults to 1.0

### Parameters

*criteria* **— specified by user, fullword integer**
Curve approximation criteria (`1=WORKSTATION_DEPENDENT`, `3=CONSTANT_SUBDIVISION_BETWEEN_KNOTS`, `8=VARIABLE_SUBDIVISION_BETWEEN_KNOTS`).

*value* **— specified by user, short floating-point number**
Control value (>=0) Depending on the *criteria* parameter values you specified, control values are as follows:

**If *criteria*=1 (`WORKSTATION_DEPENDENT`)**
A value of 1.0 specifies a nominal quality curve. A value greater than or less than 1.0 specifies a higher or lower quality curve respectively. Whether or not the tessellation vector in the curve definition is used is workstation dependent. A nominal quality of 1 will typically correspond to a chordal deviation of 1 pixel.

**If *criteria*=3 (`CONSTANT_SUBDIVISION_BETWEEN_KNOTS`)**
This value is rounded off to the nearest integer and specifies the fixed number of intervals to be used in rendering the corresponding span of the curve. If the tessellation vector is specified in the curve definition, it is ignored.

**If *criteria*=8 (`VARIABLE_SUBDIVISION_BETWEEN_KNOTS`)**
This value is multiplied by the tessellation vector in the curve definition and rounded off to the nearest integer in rendering the corresponding span of the curve. If the tessellation vector in the curve definition is not present, then the value is rounded off to the nearest integer and this becomes the number of intervals to be used in rendering the corresponding span of the curve (the same as criteria 3).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **65** | CURVE APPROXIMATION CRITERIA IS INVALID |
| **66** | CONTROL VALUE < ZERO |

### Related Subroutines

**GPQCDF** Inquire Curve Display Facilities

### RCP code

201343235 (X'0C004103')

## GPCHH - Set Character Height

**GPCHH (*height*)**

### Purpose

Use **GPCHH** to insert a Set Character Height structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Character Height structure element depending on the current edit mode.

This structure element specifies the character height in Modelling Coordinate (MC) space that the graPHIGS API uses when rendering subsequent geometric text primitives.

The traversal default for character height value is 0.01.

When the graPHIGS API encounters an element of this type, it uses the absolute value of the specified character height. If the workstation does not support a continuous range of character heights, then the graPHIGS API uses the closest supported value.

### Parameters

*height* **— specified by user, short floating-point number (MC)**
Character height (>0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **78** | CHARACTER HEIGHT VALUE <= ZERO |

### Related Subroutines

**GPQGFC** Inquire Geometric Font Characteristics

### RCP code

201328652 (X'0C00080C')

## GPCHLS - Set Character Line Scale Factor

**GPCHLS (*scale*)**

### Purpose

Use **GPCHLS** to insert a Set Character Line Scale Factor structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Character Line Scale Factor structure element depending on the current edit mode.

This structure element specifies a value that the graPHIGS API uses to determine the height of the characters when rendering all subsequent character line primitives.

This value is multiplied by the nominal height specified in the primitive definition to determine the actual height of the characters. The graPHIGS API maps the calculated value to the closest height available on the workstation.

The traversal default for character line scale factor is 1.0.

### Parameters

*scale* **— specified by user, short floating-point number**
Character line scale factor (>=0.0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **76** | CHARACTER LINE SCALE FACTOR < ZERO |

### Related Subroutines

**GPQGFC** Inquire Geometric Font Characteristics

### RCP code

201343241 (X'0C004109')

## GPCHPM - Set Character Positioning Mode

**GPCHPM (*posmode*)**

### Purpose

Use **GPCHPM** to insert a Set Character Positioning Mode structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Character Positioning Mode structure element depending on the current edit mode.

The character positioning mode determines whether the graPHIGS API uses the character positioning box for the specific character or the nominal positioning box for the font in rendering annotation and geometric text primitives.

In the symbol definition for fonts, character positioning boxes are defined for the font (all symbols in the font) and optionally for each individual symbol. This structure element only applies if the font contains character positioning boxes defined for each individual symbol. If the font only has a character positioning box defined for the font and not for the individual symbols, then this element does not affect the font.

The traversal default for character positioning mode is 1=CONSTANT.

### Parameters

*posmode* **— specified by user, fullword integer**
Character position mode (1=CONSTANT, 2=PROPORTIONAL).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **81** | CHARACTER POSITIONING MODE IS INVALID |

### Related Subroutines

**GPQXTX** Inquire Extended Text Facilities

### RCP code

201328661 (X'0C000815')

## GPCHSP - Set Character Spacing

**GPCHSP (*space*)**

### Purpose

Use **GPCHSP** to insert a Set Character Spacing structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Character Spacing structure element depending on the current edit mode.

This structure element specifies the additional amount of space the graPHIGS API inserts between characters at structure traversal time to render all subsequent text primitives when the character spacing attribute source flag is set to 2=INDIVIDUAL.

This value is expressed as a fraction of the height.

The traversal default for character spacing is 0.0.

### Parameters

*space* **— specified by user, short floating-point number**
Character spacing.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |

### Related Subroutines

| | |
|---|---|
| **GPASF** | Attribute Source Flag Setting |
| **GPQPTR** | Inquire Predefined Text Representation |
| **GPTXPR** | Set Text Precision |

### RCP code

201328650 (X'0C00080A')

## GPCHUB - Set Character Up and Base Vectors

**GPCHUB (*up*, *base*)**

### Purpose

Use **GPCHUB** to insert a Set Character Up and Base Vectors structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Character Up and Base Vectors structure element depending on the current edit mode.

During structure traversal, this structure element sets the current character up vector and base vector entries in the graPHIGS API traversal state list to the specified values.

The character up vector specifies the direction of the font coordinate y-axis within the text reference coordinate system. The character base vector specifies the direction of the font coordinate x-axis within the text reference coordinate system. The vectors need not be perpendicular to one another.

This attribute sets the same traversal state as the Set Character Up Vector (**GPCHUP**) subroutine.

The traversal default value for character up vector is 0.0, 1.0 and the traversal default value for character base vector is 1.0, 0.0.

If the character up and base vector are invalid, then the up vector defaults to 0.0, 1.0, and the base vector value defaults to 1.0, 0.0.

The graPHIGS API normalizes the specified vectors. If the application later inquires the content of this structure element, then the graPHIGS API returns the normalized vectors *not* the original vectors specified by this subroutine.

### Parameters

*up* **— specified by user, 2 short floating-point numbers**
Character up vector. (Magnitude must be >0).

*base* **— specified by user, 2 short floating-point numbers**
Character base vector. (Magnitude must be >0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **80** | CHARACTER UP AND BASE VECTORS ARE COLINEAR |

### Related Subroutines

**GPCHUP** Set Character Up Vector
**GPTXPR** Set Text Precision

**RCP code**

201328662 (X'0C000816')

# GPCHUP - Set Character Up Vector

**GPCHUP (*up*)**

## Purpose

Use **GPCHUP** to insert a Set Character Up Vector structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Character Up Vector structure element depending on the current edit mode.

During structure traversal, this structure element sets the current character up vector in the graPHIGS API traversal state list to the specified value. The base vector entries reset to the vector that is obtained by rotating the up vector 90 degrees clockwise.

The character up vector specifies the direction of the font coordinate y-axis within the text reference coordinate system. The character base vector specifies the direction of the font coordinate x-axis within the text reference coordinate system.

At structure traversal time, this structure element specifies the y-axis direction of the text coordinate system for characters in a text string that the graPHIGS API uses to render all subsequent geometric text primitives. The character up vector is a two-dimensional vector on the text plane specified by the text primitive. When rendering text primitives, the graPHIGS API uses the character up value along with a default character base vector set at right angles.

This attribute sets the same traversal state as the Set Character Up and Base Vector (**GPCHUB**) subroutine. The traversal default value for character up vector is 0.0, 1.0 and the traversal default value for character base vectors is 1.0, 0.0.

If the character up vector is invalid, then the up vector value defaults to 0.0, 1.0 and the base vector value defaults to 1.0, 0.0.

The graPHIGS API normalizes the specified vector. If the application later inquires the content of this structure element, then the graPHIGS API returns the normalized vector, *not* the original vector specified by this subroutine.

## Parameters

*up* **— specified by user, 2 short floating-point numbers**
Character up vector. (Magnitude must be >0).

## Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **79** | CHARACTER UP VECTOR HAS LENGTH ZERO |

## Related Subroutines

| | |
|---|---|
| **GPCHUB** | Set Character Up and Base Vectors |
| **GPTXPR** | Set Text Precision |

### RCP code

201328653 (X'0C00080D')

## GPCHXP - Set Character Expansion Factor

**GPCHXP (*expans*)**

### Purpose

Use **GPCHXP** to insert a Set Character Expansion Factor structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Character Expansion Factor structure element depending on the current edit mode.

This structure element specifies the character expansion factor that the graPHIGS API uses at structure traversal time to render all subsequent text primitives when the character expansion attribute source flag is set to `2=INDIVIDUAL`.

The value is a fraction of the width/height ratio that the font designer specified. A value of 1.0 reproduces the font designer's width/height ratio.

The traversal default value for character expansion factor is 1.0.

If the workstation does not support a the character expansion factor or the specified expansion factor is negative or zero, then the graPHIGS API uses a default expansion factor of 1.0.

### Parameters

*expans* **— specified by user, short floating-point number**
Character expansion factor (>0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **77** | CHARACTER EXPANSION FACTOR <= ZERO |

### Related Subroutines

| | |
|---|---|
| **GPASF** | Attribute Source Flag Setting |
| **GPQGFC** | Inquire Geometric Font Characteristics |
| **GPQPTR** | Inquire Predefined Text Representation |
| **GPQXAF** | Inquire Extended Annotation Font Characteristics |

### RCP code

201328649 (X'0C000809')

## GPCPI - Set Color Processing Index

**GPCPI (*index*)**

### Purpose

Use **GPCPI** to insert a Set Color Processing Mode structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Color Processing Mode structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's color processing table. The attributes defined in the entry control the color processing (color quantization) applied to subsequent primitives.

The traversal default color processing index value is zero.

If the workstation does not support the specified index value or the specified index is outside the color processing table size, then the color processing index value defaults to zero.

### Parameters

*index* **— specified by user, fullword integer**
Color processing mode index (>=0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **265** | COLOR PROCESSING INDEX < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPCPR** | Set Color Processing Representation |
| **GPDCI** | Set Depth Cue Index |
| **GPDCR** | Set Depth Cue Representation |
| **GPQCPF** | Inquire Color Processing Facilities |
| **GPQRCM** | Inquire Available Rendering Color Models |

### RCP code

201343492 (X'0C004204')

## GPDCI - Set Depth Cue Index

**GPDCI (*index*)**

### Purpose

Use**GPDCI** to insert a Set Depth Cue Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Depth Cue Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's depth cue table. The attributes defined in the entry control how depth cueing is applied to subsequent primitives. Each entry contains the depth cue mode, depth cue color, depth cue reference planes, and depth cue scale factors.

The traversal default depth cue index value is 0.

If the workstation does not support the specified index value or the specified index is outside the depth cue table size, then the depth cue index value defaults to 0.

### Parameters

*index* — **specified by user, fullword integer**
Depth cue table index (>=0)

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **261** | DEPTH CUE INDEX < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPCPI** | Set Color Processing Index |
| **GPCPR** | Set Color Processing Representation |
| **GPDCR** | Set Depth Cue Representation |
| **GPLMO** | Set Lighting Calculation Mode |
| **GPLSR** | Set Light Source Representation |
| **GPLSS** | Set Light Source State |
| **GPQDCF** | Inquire Depth Cue Facilities |

### RCP code

201343239 (X'0C004107')

## GPDFM - Set Data Filtering Method

**GPDFM (*minfm, magfm, boundu, boundv*)**

### Purpose

Use **GPDFM** to insert a Set Data Filtering Method structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Data Filtering Method structure element, depending upon the current edit mode.

This element specifies the filtering methods that the graPHIGS API uses to perform data mapping on subsequent area primitives. When the data mapping method is `2=SINGLE_VALUE_UNIFORM`, the *boundv* parameter is ignored.

The traversal default data filtering methods are:

| | |
|---|---|
| **Minification:** | `1=SAMPLE_IN_BASE` |
| **Magnification:** | `1=SAMPLE_IN_BASE` |
| **U-dimension:** | `1=CLAMP` |
| **V-dimension:** | `1=CLAMP` |

If any specified value is not supported on the workstation, then the default value is used.

### Parameters

*minfm* **— specified by user, fullword integer**
Minification filtering method (1=SAMPLE_IN_BASE, 2=INTERP_IN_BASE, 3=SAMPLE_IN_SQUARE_MM, 4=SAMPLE_IN_AND_INTERP_BTWN_SQUARE_MM, 5=INTERP_IN_SQUARE_MM, 6=INTERP_IN_AND_BTWN_SQUARE_MM, 7=SAMPLE_IN_RECT_MM, 8=SAMPLE_IN_AND_INTERP_BTWN_RECT_MM, 9=INTERP_IN_RECT_MM).

*magfm* **— specified by user, fullword integer**
Magnification filtering method (1=SAMPLE_IN_BASE, 2=INTERP_IN_BASE).

*boundu* **— specified by user, fullword integer**
*u*-dimension bounding method (1=CLAMP, 2=REPEAT).

*boundv* **— specified by user, fullword integer**
*v*-dimension bounding method (1=CLAMP, 2=REPEAT).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **631** | FILTERING METHOD IS INVALID |
| **632** | BOUNDING METHOD IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPBDFM** | Set Back Data Filtering Method |
| **GPDMR** | Set Data Mapping Representation |

### RCP code

201343515 (X'0C00421B')

## GPDMI - Set Data Mapping Index

*GPDMI (index*)

### Purpose

Use **GPDMI** to insert a Set Data Mapping Index structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Data Mapping Index structure element, depending upon the current edit mode.

During traversal, the data mapping index specifies the entry in the data mapping table used to perform data mapping on subsequent primitives that are to be data mapped.

The data mapping table is zero based. Entry zero always contains a data mapping entry of 1=DM_METHOD_COLOR. Use **GPQWDT** to determine the number of definable data mapping table entries.

The traversal default data mapping index is zero. If the specified index is not supported on the workstation, then graPHIGS API uses a default index of zero.

### Parameters

*index* **— specified by user, fullword integer**
Data mapping table index (>= 0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **630** | DATA MAPPING INDEX < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPBDMI** | Set Back Data Mapping Index |
| **GPDMR** | Set Data Mapping Representation |

### RCP code

201343513 (X'0C004219')

## GPDM2 - Set Data Matrix 2

**GPDM2 (*matrix*)**

### Purpose

Use **GPDM2** to insert a Set Data Matrix 2 structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Data Matrix 2 structure element, depending upon the current edit mode.

This element specifies the transformation matrix that the graPHIGS API uses to modify the data mapping values specified in primitives that support data mapping.

The data mapping values are treated as homogeneous points of the form (*u, v,* 1.0). When used with a data mapping method of `2=SINGLE_VALUE_UNIFORM`, the graPHIGS API sets the *v* coordinate value to 0.0. Otherwise, the graPHIGS API uses indexed vertex data values for the *u* and *v* coordinate values. This point is multiplied by the specified matrix. The resulting values are used to perform the data mapping.

The last column of the matrix must have the values 0.0, 0.0, 1.0.

The traversal default data matrix is the identity matrix.

### Parameters

*matrix* **— specified by user, 9 short floating-point numbers**
Data modification matrix.

The elements of the matrix must be specified as follows:

```
|m11  M12  0.0|
|m21  M22  0.0| ---> (m11, m12, 0.0, m21, m22, 0.0, m31, m32, 1.0)
|m31  M32  1.0|
```

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **633** | MATRIX VALUE IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPBDM2** | Set Back Data Matrix 2 |
| **GPDMR** | Set Data Mapping Representation |

### RCP code

201343517 (X'0C00421D')

## GPECD - Set Edge Color Direct

**GPECD (*color*)**

### Purpose

Use **GPECD** to insert a Set Edge Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Edge Color Direct structure element depending on the current edit mode.

This structure element specifies the direct color values that the graPHIGS API uses to render the edges of subsequent output primitives to which this attribute applies. The graPHIGS API uses these values at structure traversal time to render the edges of output primitives when the edge color attribute source flag is set to `2=INDIVIDUAL` and the edge flag is set to `2=ON`.

This attribute sets the same traversal state as the Set Edge Color Index (**GPECI**) subroutine. The traversal default for edge color is the content of entry 1 of the rendering color table.

### Parameters

*color* **— specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **96** | COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL |

### Related Subroutines

| | |
|---|---|
| **GPASF** | Attribute Source Flag Setting |
| **GPECI** | Set Edge Color Index |
| **GPQXER** | Inquire Extended Edge Representation |
| **GPXER** | Set Extended Edge Representation |

### RCP code

201328904 (X'0C000908')

## GPECI - Set Edge Color Index

**GPECI (*index*)**

### Purpose

Use **GPECI** to insert a Set Edge Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Edge Color Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that defines the color the graPHIGS API uses to render the edges of all output primitives to which this attribute applies. At structure traversal time, the graPHIGS API uses this index to render the edges of output primitives when the edge color aspect source flag is set to `2=INDIVIDUAL` and the edge flag is set to `2=ON`.

This attribute sets the same traversal state as the Set Edge Color Direct (**GPECD**) subroutine. The traversal default for edge color is a color index value of 1.

If the workstation does not support the specified color index value or the specified index is outside the color table limit, then the color index defaults to a value of 1.

### Parameters

*index* **— specified by user, fullword integer**
Color index (>=0).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **92** | COLOR INDEX < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPASF** | Attribute Source Flag Setting |
| **GPECD** | Set Edge Color Direct |
| **GPQPER** | Inquire Predefined Edge Representation |
| **GPQXER** | Inquire Extended Edge Representation |
| **GPXER** | Set Extended Edge Representation |

### RCP code

201328902 (X'0C000906')

## GPEF - Set Edge Flag

**GPEF (*edgefg*)**

### Purpose

Use **GPEF** to insert a Set Edge Flag structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Edge Flag structure element depending on the current edit mode.

This structure element indicates whether or not to draw the graPHIGS API draws the edge of subsequent polygon primitives during structure traversal. The graPHIGS API uses the specified value if the edge flag aspect source flag is set to 2=INDIVIDUAL.

An edge flag of 3=GEOMETRY_ONLY forces the graPHIGS API to raster edges of polygons using a line algorithm as opposed to an area algorithm. This guarantees that a line is drawn coincident with the boundary of a polygon after a polygon is drawn.

The traversal default edge flag is 1=OFF.

If the workstation does not support the specified edge flag value, then the edge flag defaults to 1=OFF.

### Parameters

*edgefg* **— specified by user, fullword integer**
Edge flag (1=OFF, 2=ON, 3=GEOMETRY_ONLY).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **311** | EDGE FLAG VALUE IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPASF** | Attribute Source Flag Setting |
| **GPQAAF** | Inquire Advanced Attribute Facilities |
| **GPQEF** | Inquire Edge Facilities |
| **GPQPER** | Inquire Predefined Edge Representation |

### RCP code

201328900 (X'0C000904')

## GPEI - Set Edge Index

**GPEI (*index*)**

### Purpose

Use **GPEI** to insert a Set Edge Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Edge Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's edge bundle table. The entry contains attribute settings for edge flag, edge line type, edge scale factor, and edge color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent polygon primitives for those attributes which have an aspect source flag value set to 1=BUNDLED.

The traversal default edge index value is 1.

If the workstation does not support the specified index, then the edge index defaults to a value of 1.

### Parameters

*index* **— specified by user, fullword integer**
Edge bundle table index (>=1).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **60** | BUNDLE INDEX VALUE < ONE |

### Related Subroutines

| | |
|---|---|
| **GPQLW** | Inquire Length of Workstation State Tables |
| **GPQXER** | Inquire Extended Edge Representation |
| **GPXER** | Set Extended Edge Representation |

### RCP code

201328388 (X'0C000704')

## GPELT - Set Edge Linetype

**GPELT (*edgelt*)**

### Purpose

Use **GPELT** to insert a Set Edge Linetype element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Edge Linetype element depending on the current edit mode.

This structure element specifies an index into a workstation line type table that contains line types. The graPHIGS API uses this index to render the edges of all subsequent output primitives if the corresponding edge flag is set to `2=ON`. At structure traversal time, the graPHIGS API uses this line type to render the edges of output primitives when the line type of an edge aspect source flag is set to `2=INDIVIDUAL`.

The traversal default for edge line type is `1=SOLID_LINE`.

If the workstation does not support the specified index, then the edge index defaults to `1=SOLID_LINE`.

### Parameters

*edgelt* **— specified by user, fullword integer**
Specifies an index into the workstation's edge line type table. The table size and specific entries supported are workstation dependent. Use the Inquire Edge Facilities (**GPQEF**) subroutine to determine the supported edge line types on your workstation. The default edge line type table for supported entries is defined with the following line types:

```
1=SOLID_LINE
2=DASHED
3=DOTTED
4=DASH_DOT
```

```
5=LONG_DASH
6=DOUBLE_DOT
7=DASH_DOUBLE_DOT
8-n=SOLID_LINE
```

Any entry may be changed by the Set Linetype Representation (**GPLTR**) subroutine except entry 1.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **63** | LINETYPE VALUE < ONE |

### Related Subroutines

| | |
|---|---|
| **GPASF** | Attribute Source Flag Setting |
| **GPLTR** | Set Linetype Representation |
| **GPQER** | Inquire Edge Representation |
| **GPQLTR** | Inquire Linetype Representation |
| **GPQPER** | Inquire Predefined Edge Representation |
| **GPQXER** | Inquire Extended Edge Representation |
| **GPXER** | Set Extended Edge Representation |

### RCP code

201328901 (X'0C000905')

## GPESC - Set Edge Scale Factor

**GPESC (*edgesf*)**

### Purpose

Use **GPESC** to insert a Set Edge Scale Factor structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Edge Scale Factor structure element depending on the current edit mode.

This structure element specifies a value that the graPHIGS API uses to determine how wide to draw the edges of subsequent output primitives to which this attribute applies. At structure traversal time, the graPHIGS API uses this scale factor to determine the width of the edge when the edge scale factor aspect source flag is set to `2=INDIVIDUAL` and additionally for polygon output primitives when the corresponding edge flag is set to `2=ON`.

The edge scale factor element specifies the edge's width as a fraction of the nominal edge width. The device support multiplies this scale factor by the nominal width of a line on the corresponding device to determine the requested width. The graPHIGS API maps the calculated value to the closest width available on the device. A scale factor of 1.0, which is the traversal default, generates a nominal size line on any workstation.

### Parameters

*edgesf* **— specified by user, short floating-point number**
Edge scale factor, specified as a fraction of the nominal linewidth.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPQEF**
Inquire Edge Facilities

**GPQPER**
Inquire Predefined Edge Representation

**GPQXER**
Inquire Extended Edge Representation

**GPXER**
Set Extended Edge Representation

### RCP code

201328903 (X'0C000907')

## GPFBC - Set Frame Buffer Comparison

**GPFBC (*op*, *mask*, *value*)**

### Purpose

Use **GPFBC** to insert a Set Frame Buffer Comparison structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Frame Buffer Comparison structure element depending on the current edit mode.

This structure element specifies a set of options to be taken after comparing the current frame buffer content with a comparison value using a comparison mask. When a bit is set to a value of one in the mask, the graPHIGS API uses the corresponding frame buffer contents in the comparison with the specified comparison value.

The traversal default for frame buffer operation is `1=NO_OPERATION`.

**GPFBC** is identified as GSE 1002.

> **Note:** Not all GSEs are supported on all workstations. Use the List of Available GSEs (**GPQGSE**) subroutine to determine the GSEs which are supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*op* **— specified by user, fullword integer**
Frame buffer options:

**`1=NO_OPERATION`**
No comparison is made (*mask* and *value* are not used)

**`2=WRITE_WHEN_EQUAL`**
Before updating the frame buffer with a new pixel value, the current contents of the frame buffer are compared with the specified comparison color value using the specified

comparison mask. If they are equal, the new color value is put into the frame buffer. If they are not equal, then the current frame buffer pixel is not changed at all. This option is also known as *underpaint*.

**3=WRITE_WHEN_NOT_EQUAL**
Before updating the frame buffer with a new pixel value, the current contents of the frame buffer are compared with the specified comparison color value using the specified comparison mask. If they are equal, the new pixel value is put into the frame buffer. If they are not equal, the pixel value produced by processing the current line-on-line color attribute is put into the frame buffer. This option is also known as *line-on-line highlighting*.

*mask* **— specified by user, fullword integer**
Comparison frame buffer mask. A bit set to 1 in the mask indicates the corresponding frame buffer bit plane that will be used in the comparison.

For an indexed frame buffer workstation, its least significant *n* bits (where *n* is the bit depth of the frame buffer) are used for the actual comparison mask. For a component frame buffer workstation, its least significant *n*1 bits, the next least significant *n*2 bits, and the next least significant *n*3 bits are used for each frame buffer component where *n*1, *n*2, *n*3 are the bit depths of the three frame buffer components.

*value* **— specified by user, fullword integer**
Comparison value. For an indexed frame buffer workstation, the least significant *n* bits (where *n* is the bit depth of the frame buffer) of the comparison value are used for the comparison. For a component frame buffer workstation, its least significant *n*1 bits, the next least significant *n*2 bits, and the next least significant *n*3 bits are used for each frame buffer component where *n*1, *n*2, and *n*3 are the bit depths of the three frame buffer components.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**118** FRAME BUFFER COMPARISON IS INVALID

### Related Subroutines

**GPLLCD**
Set Line-on-Line Color Direct

**GPLLCI**
Set Line-on-Line Color Index

**GPQFBC**
Inquire Frame Buffer Characteristics

### RCP code

201343234 (X'0C004102')

## GPFBM - Set Frame Buffer Protect Mask

**GPFBM (*mask*)**

### Purpose

Use **GPFBM** to insert a Set Frame Buffer Protect Mask structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Frame Buffer Protect Mask structure element depending on the current edit mode.

This structure specifies a write protect mask. The effect of the mask is to inhibit writing to the masked bit planes in the frame buffer during the rendering of output primitives.

The traversal default for frame buffer protect mask is zero. If a bit is set to a value of one, the corresponding bit plane is protected and cannot be modified.

**GPFBM** is identified as GSE 1001.

> **Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Available GSEs (**GPQGSE**) subroutine to determine the GSEs which are supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*mask* **— specified by user, fullword integer**
Frame buffer protect mask. A bit set to 1 in the mask indicates that the corresponding bit plane is protected and cannot be modified.

For an indexed frame buffer workstation, its least significant *n* bits (where *n* is the bit depth of the frame buffer) are used for the actual read/write mask. For a component frame buffer workstation, its least significant *n*1 bits, the next least significant *n*2 bits, and the next least significant *n*3 bits are used for each frame buffer component where *n*1, *n*2, and *n*3 are the bit depths of the three frame buffer components.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**Related Subroutines**

**GPQFBC**
Inquire Frame Buffer Characteristics

**RCP code**

201343233 (X'0C004101')

## GPFDMO - Set Face Distinguish Mode

**GPFDMO (*mode*)**

**Purpose**

Use **GPFDMO** to insert a Set Face Distinguish Mode structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Face Distinguish Mode structure element depending on the current edit mode.

During structure traversal, this structure element sets the current face distinguish mode entry in the graPHIGS API traversal state list to the value specified by the parameter. The graPHIGS API uses this value when creating subsequent area defining output primitives.

Face distinguish mode must be set to `2=COLOR_SURFACE_PROPERTIES` for the graPHIGS API to apply the front face color and surface properties attributes to the front face facets, and to apply the back face color and surface properties attributes to back face facets. If face distinguish mode is set to `1=NONE`, then the graPHIGS API applies front face color and surface properties attributes to both the front and back face facets.

The traversal default for face distinguish mode is 1=NONE.

### Parameters

*mode* **— specified by user, fullword integer**
Face distinguish mode (1=NONE, 2=COLOR_SURFACE_PROPERTIES).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**88** FACE DISTINGUISH MODE IS INVALID

### Related Subroutines

**GPQAAF**
Inquire Advanced Attribute Facilities

### RCP code

201343495 (X'0C004207')

## GPFLM - Set Face Lighting Method

**GPFLM (*flmeth*)**

### Purpose

Use **GPFLM** to insert a Set Face Lighting Method structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Face Lighting Method structure element depending on the current edit mode.

During structure traversal, this structure element sets the face lighting method entry of the graPHIGS API traversal state list to the value specified by the parameter. This value is used during the lighting of primitives which define area.

Face lighting options enable you to position an object in relation to a light source and a viewer. For instance, if a viewer is in front of a opaque object and the light source is on the other, the viewer does not see the light and the side of the viewer is dark or in the shadows. 2=FACE_DEPENDENT enables this effect. 1=FACE_INDEPENDENT disables the effect. For more information on face lighting options, see face-dependent lighting in *The graPHIGS Programming Interface: Understanding Concepts*.

The traversal default for face lighting method is 1=FACE_INDEPENDENT.

If the workstation does not support the specified face lighting method, then the face lighting method defaults to 1=FACE_INDEPENDENT.

### Parameters

*flmeth* **— specified by user, fullword integer**
Face lighting method (1=FACE_INDEPENDENT, 2=FACE_DEPENDENT).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**299** FACE LIGHTING METHOD IS INVALID

### Related Subroutines

None

### RCP code

201343503 (X'0C00420F')

## GPHID - Set HLHSR Identifier

**GPHID (*hlhsr*)**

### Purpose

Use **GPHID** to insert a Set HLHSR Identifier structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set HLHSR Identifier structure element depending on the current edit mode.

During structure traversal, this structure element sets the current HLHSR identifier entry of the graPHIGS API traversal state list to the specified parameter. The application uses this value when creating subsequent output primitives in a view with a HLHSR mode other than `1=OFF`.

If the workstation does not support the specified HLHSR identifier, then the HLHSR identifier defaults to a value of `1=VISUALIZE_IF_NOT_HIDDEN`. In a view with HLHSR mode set to `1=OFF`, this value is completely ignored and has no affect on the visualization of primitives.

HLHSR processing is often implemented by use of a z-buffer and a frame buffer. The following table summarizes the effect of the various HLHSR identifiers on the z-buffer and the frame buffer:

**Table 3. HLHSR Processing**

| Summary of when the frame buffer and the z-buffer are updated. | | |
|---|---|---|
| | **Frame buffer** | **Z-buffer** |
| `1=VISUALIZE_IF_NOT_HIDDEN` | $Z_{prim} >= Z_{buf}$ | $Z_{prim} >= Z_{buf}$ |
| `2=VISUALIZE_IF_HIDDEN` | $Z_{prim} < Z_{buf}$ | Never |
| `3=VISUALIZE_ALWAYS` | Always | Always |
| `4=NOT_VISUALIZE` | Never | $Z_{prim} >= Z_{buf}$ |
| `5=FACE_DEPENDENT_VISUALIZATION` | | |
| Front-facing Areas | $Z_{prim} >= Z_{buf}$ | $Z_{prim} >= Z_{buf}$ |
| Back-facing Areas | $Z_{prim} > Z_{buf}$ | $Z_{prim} > Z_{buf}$ |
| `6=NO_UPDATE` | Never | Never |
| `7=GREATER_THAN` | $Z_{prim} > Z_{buf}$ | $Z_{prim} > Z_{buf}$ |
| `8=EQUAL_TO` | $Z_{prim} = Z_{buf}$ | $Z_{prim} = Z_{buf}$ |
| `9=LESS_THAN` | $Z_{prim} < Z_{buf}$ | $Z_{prim} < Z_{buf}$ |
| `10=NOT_EQUAL` | $Z_{prim} <> Z_{buf}$ | $Z_{prim} <> Z_{buf}$ |
| `11=LESS_THAN_OR_EQUAL_TO` | $Z_{prim} <= Z_{buf}$ | $Z_{prim} <= Z_{buf}$ |

**Note:** The actual update of the z-buffer and/or the frame buffer may be prohibited by the use of the z-buffer protect mask and the frame buffer protect mask.

### Parameters

*hlhsr* **— specified by user, fullword integer**
HLHSR identifier (1=VISUALIZE_IF_NOT_HIDDEN, 2=VISUALIZE_IF_HIDDEN, 3=VISUALIZE_ALWAYS, 4=NOT_VISUALIZE, 5=FACE-DEPENDENT_VISUALIZATION, 6=NO_UPDATE, 7=GREATER_THAN, 8=EQUAL_TO, 9=LESS_THAN, 10=NOT_EQUAL, 11=LESS_THAN_OR_EQUAL_TO).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**250** HLHSR IDENTIFIER IS INVALID

### Related Subroutines

**GPQHMO**
Inquire Available HLHSR Modes

**GPXVR**
Set Extended View Representation

### RCP code

201343240 (X'0C004108')

## GPHLCD - Set Highlighting Color Direct

**GPHLCD (*color*)**

### Purpose

Use **GPHLCD** to insert a Set Highlighting Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Highlighting Color Direct structure element depending on the current edit mode.

This structure element specifies the direct color values the graPHIGS API uses to render all subsequent highlighted primitives. This color overrides any color specified by the attribute color for the primitive or specified in the primitive definition.

This attribute sets the same traversal state as the Set Highlighting Color Index (**GPHLCI**) subroutine. The traversal default for highlighting color is the content of entry 1 of the rendering color table.

### Parameters

*color* **— specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

### Related Subroutines

**GPHLCI**
Set Highlighting Color Index

**RCP code**

201328669 (X'0C00081D')

## GPHLCI - Set Highlighting Color Index

**GPHLCI (*index*)**

**Purpose**

Use **GPHLCI** to insert a Set Highlighting Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Highlighting Color Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that contains the color the graPHIGS API uses to render all subsequent highlighted primitives. This color overrides any color specified by the attribute color for the primitive or specified in the primitive definition.

This attribute sets the same traversal state as the Set Highlighting Color Direct (**GPHLCD**) subroutine. The traversal default for highlighting color is a color index value of 1.

If the workstation does not support the specified index, then the color index defaults to a value of 1.

**Parameters**

*index* **— specified by user, fullword integer**
Color index (>=0).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**92** COLOR INDEX < ZERO

**Related Subroutines**

**GPHLCD**
Set Highlighting Color Direct

**RCP code**

201328656 (X'0C000810')

## GPICD - Set Interior Color Direct

**GPICD (*color*)**

**Purpose**

Use **GPICD** to insert a Set Interior Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Interior Color Direct structure element depending on the current edit mode.

This structure element specifies the direct color values the graPHIGS API uses to fill all following polygon definitions if the interior style is set to 1=HOLLOW and edge is 1=OFF or if the interior style is set to 2=SOLID or 4=HATCH.

Face distinguish mode must be set to 2=COLOR_SURFACE_PROPERTIES for the graPHIGS API to use these values to fill the front facing portions only. If the face distinguish mode is 1=NONE, then the graPHIGS API uses these values to fill both the front facing and back facing portions of the area defining primitives.

At structure traversal time, the graPHIGS API uses these values to render the interiors when the interior color aspect source flag is set to 2=INDIVIDUAL.

This attribute sets the same traversal state as the Set Interior Color Index (**GPICI**) subroutine. The traversal default for interior color is the content of entry 1 of the rendering color table.

**Parameters**

*color* **— specified by user, 3 short floating-point numbers.**
Three color components of the current structure color model in the graPHIGS API state list (0.0<=component<=1.0).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPEF** Set Edge Flag

**GPFDMO**
Set Face Distinguish Mode

**GPICI** Set Interior Color Index

**GPQXIR**
Inquire Extended Interior Representation

**GPXIR**
Set Extended Interior Representation

**RCP code**

201328905 (X'0C000909')

## GPICI - Set Interior Color Index

**GPICI (*index*)**

**Purpose**

Use **GPICI** to insert a Set Interior Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Interior Color Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that contains the color values the graPHIGS API uses to fill all following polygon definitions if the interior style is set to 1=HOLLOW and edge is 1=OFF, or the interior style is set to 2=SOLID or 4=HATCH.

Face distinguish mode must be set to 2=COLOR_SURFACE_PROPERTIES for the graPHIGS API to fill the front facing properties only. If the face distinguish mode is 1=NONE, then the graPHIGS API uses the color values to fill both the front facing and back facing portions of the polygon definitions.

This index is used at structure traversal time to render the interiors when the interior color aspect source flag is set to 2=INDIVIDUAL.

This attribute sets the same traversal state as the Set Interior Color Direct (**GPICD**) subroutine. The traversal default for interior color is a color index value of 1.

If the workstation does not support the specified index, then the color index defaults to a value of 1.

**Parameters**

*index* **— specified by user, fullword integer**
Color index (>=0).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**92** COLOR INDEX < ZERO

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPEF** Set Edge Flag

**GPFDMO**
Set Face Distinguish Mode

**GPICD**
Set Interior Color Direct

**GPQPIR**
Inquire Predefined Interior Representation

**GPQXIR**
Inquire Extended Interior Representation

**GPXIR**
Set Extended Interior Representation

**RCP code**

201328899 (X'0C000903')

## GPII - Set Interior Index

**GPII (*index*)**

**Purpose**

Use **GPII** to insert a Set Interior Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Interior Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's interior bundle table. The entry contains attribute settings for interior style, interior style index, and color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent polygon primitives for those attributes which have an aspect source flag set to `1=BUNDLED`.

The traversal default interior index is 1.

If the workstation does not support the specified index, then the interior index defaults to a value of 1.

### Parameters

*index* **— specified by user, fullword integer**
Interior bundle table index (>=1).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**60** BUNDLE INDEX VALUE < ONE

### Related Subroutines

**GPQLW**
Inquire Length of Workstation State Tables

**GPQXIR**
Inquire Extended Interior Representation

**GPXIR**
Set Extended Interior Representation

### RCP code

201328389 (X'0C000705')

## GPIS - Set Interior Style

**GPIS (*style*)**

### Purpose

Use **GPIS** to insert a Set Interior Style structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Interior Style structure element depending on the current edit mode.

This structure element specifies the way the graPHIGS API draws the interior of a polygon at structure traversal time when rendering all subsequent polygon (fill area) primitives. The graPHIGS API uses this style value when the interior style aspect source flag is set to `2=INDIVIDUAL`.

The traversal default for interior style is `1=HOLLOW`.

If the workstation does not support the specified interior style, then the interior style defaults to `1=HOLLOW`.

### Parameters

*style* **— specified by user, fullword integer**
Interior style (1=HOLLOW, 2=SOLID, 3=PATTERN, 4=HATCH, 5=EMPTY).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**310** INTERIOR STYLE VALUE IS INVALID

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPISI** Set Interior Style Index

**GPQIF**
Inquire Interior Facilities

**GPQPIR**
Inquire Predefined Interior Representation

**GPQXIR**
Inquire Extended Interior Representation

**GPXIR**
Set Extended Interior Representation

**RCP code**

201328897 (X'0C000901')

## GPISI - Set Interior Style Index

**GPISI (*index*)**

**Purpose**

Use **GPISI** to insert a Set Interior Style Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Interior Style Index structure element depending on the current edit mode.

This structure element specifies an index into the workstation hatch table if the current interior style is 4=HATCH. If the current interior style is 3=PATTERN, then this structure element specifies an index into the workstation pattern table. The graPHIGS API uses this value at structure traversal time when rendering all subsequent area defining primitives when the interior style index aspect source flag is set to 2=INDIVIDUAL.

If the current interior style is *not* a hatch or pattern interior style, then the graPHIGS API ignores this structure element and the element counter is simply incremented.

The traversal default interior style index value is 1.

If the workstation does not support the specified index, then the interior style index defaults to a value of 1.

**Parameters**

*index* **— specified by user, fullword integer**
Interior style index (>=1).

The default hatch table for workstations is defined as follows:

1. Vertical lines.
2. Horizontal lines.
3. Diagonal lines, lower left to upper right, wide spacing.
4. Diagonal lines, lower left to upper right, medium spacing.
5. Diagonal lines, lower right to upper left, wide spacing.
6. Diagonal lines, lower right to upper left, medium spacing.
7. Raster pattern 1.
8. Raster pattern 2.
9. Raster pattern 3.
10. Raster pattern 4.
11. Raster pattern 5.
12. Raster pattern 6.
13. Raster pattern 7.
14. Raster pattern 8.
15. Horizontal/Vertical cross-hatch, spacing 1.
16. Diagonal cross-hatch, spacing 1.
17. Horizontal/Vertical cross-hatch, spacing 2.
18. Diagonal cross-hatch, spacing 2.
19. Horizontal/Vertical cross-hatch, spacing 3.
20. Diagonal cross-hatch, spacing 3.
21. Horizontal/Vertical cross-hatch, spacing 4.
22. Diagonal cross-hatch, spacing 4.
23. Brick pattern, horizontal.
24. Brick pattern, diagonal.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**84** INTERIOR STYLE INDEX VALUE < ONE

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPHR** Set Hatch Representation

**GPIS** Set Interior Style

**GPPAR**
Set Pattern Representation

**GPQIF**
Inquire Interior Facilities

**GPQPIR**
Inquire Predefined Interior Representation

**GPQXIR**
Inquire Extended Interior Representation

**GPXIR**
Set Extended Interior Representation

**RCP code**

201328898 (X'0C000902')

## GPISM - Set Interior Shading Method

**GPISM (*method*)**

### Purpose

Use **GPISM** to insert a Set Interior Shading Method structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Interior Shading Method structure element, depending on the current edit mode.

The graPHIGS API uses this element to specify the shading of the interior of subsequent area primitives.

If the face distinguish mode (**GPFDMO**) is `1=NONE`, then the graPHIGS API uses this method to shade the interior of both front and back facing portions of area primitives. If the face distinguish mode is `2=COLOR_SURFACE_PROPERTIES`, then the graPHIGS API uses this method only on front facing portions of subsequent area primitives.

The interior shading methods include `1=SHADING_NONE` which is also known as *flat* shading, `2=SHADING_COLOR`, traditionally known as *Gourand* shading, and `3=SHADING_DATA`. See *The graPHIGS Programming Interface: Understanding Concepts* for information on the interactions between lighting, shading, and data mapping.

For compatibility, the Set Interior Shading Method traversal state is also set by the **GPLMO** subroutine. For more information, see *The graPHIGS Programming Interface: Understanding Concepts*.

The traversal default interior shading method is `2=SHADING_COLOR`. If the workstation does not support the specified method, then the graPHIGS API uses `2=SHADING_COLOR`.

### Parameters

*method* **— specified by user, fullword integer**
Interior shading method (`1=SHADING_NONE`, `2=SHADING_COLOR`, `3=SHADING_DATA`).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**512** METHOD NOT SUPPORTED

### Related Subroutines

**GPBISM**
Set Back Interior Shading Method

**GPQAAF**
Inquire Advanced Attribute Facilities

### RCP code

201343511 (X'0C004217')

## GPLLCD - Set Line-on-Line Color Direct

**GPLLCD (*color*)**

### Purpose

Use **GPLLCD** to insert a Set Line-on-Line Color Direct structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Line-on-Line Color Direct structure element, depending upon the current edit mode.

This structure element specifies the direct color values that the graPHIGS API uses when highlighting using the Set Frame Buffer Comparison (**GPFBC**)option `3=WRITE_WHEN_NOT_EQUAL`.

Depending upon the results of the Frame Buffer Comparison, this color overrides any color specified either by the attribute color for the primitive or in the primitive's definition.

This attribute sets the same traversal state as the Set Line-on-Line Color Index (**GPLLCI**) subroutine. The traversal default for the line-on-line color is the content of entry 1 of the rendering color table.

**GPLLCD** is identified as GSE 1011.

> **Note:** Not all GSEs are supported on all workstations. Use the List of Available GSEs (**GPQGSE**) subroutine to determine the GSEs which are supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*color*— **specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

### Related Subroutines

**GPFBC**
Set Frame Buffer Comparison

**GPLLCI**
Set Line-on-Line Color Index

### RCP code

201343523 (X'0C004223')

## GPLLCI - Set Line-on-Line Color Index

**GPLLCI (*index*)**

### Purpose

Use **GPLLCI** to insert a Set Line-on-Line Color Index structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Line-on-Line Color Index structure element, depending upon the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that contains the color that the graPHIGS API uses when highlighting using the Set Frame Buffer Comparison (**GPFBC**) option `3=WRITE_WHEN_NOT_EQUAL`.

Depending upon the results of the Frame Buffer Comparison, this color overrides any color specified either by the attribute color for the primitive or in the primitive's definition.

This attribute sets the same traversal state as the Set Line-on-Line Color Direct (**GPLLCD**) subroutine. The traversal default for the line-on-line color is the content of entry 1 of the rendering color table.

If the specified index is not supported by the workstation then a default color index of *1* is used.

**GPLLCI** is identified as GSE 1012.

> **Note:** Not all GSEs are supported on all workstations. Use the List of Available GSEs (**GPQGSE**) subroutine to determine the GSEs which are supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*index*— **specified by user, fullword integer**
Color index (`>=0`).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**92** COLOR INDEX < ZERO

**Related Subroutines**

**GPFBC**
Set Frame Buffer Comparison

**GPLLCD**
Set Line-on-Line Color Direct

**RCP code**

201343522 (X'0C004222')

## GPLMO - Set Lighting Calculation Mode

**GPLMO (*mode*)**

**Purpose**

Use **GPLMO** to insert a Set Lighting Calculation Mode structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Lighting Calculation Mode structure element depending on the current edit mode.

This structure element specifies a mode which the graPHIGS API uses at structure traversal time to control the lighting process of subsequent area defining output primitives in order to control the quality and

performance of the resulting display. Performance typically decreases as quality increases. This value selects the most coarse granularity at which lighting calculations are to be performed.

The traversal default for lighting calculation mode is 1=NONE.

If the workstation does not support the specified mode, then the lighting calculation mode defaults to 1=NONE.

### Parameters

*mode* **— specified by user, fullword integer**
Lighting calculation mode (1=NONE, 2=PER_AREA, 3=PER_VERTEX).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL).

**257** LIGHTING CALCULATION MODE IS INVALID.

### Related Subroutines

**GPBSPR**
Set Back Surface Properties

**GPDCI**
Set Depth Cue Index

**GPDCR**
Set Depth Cue Representation

**GPLSR**
Set Light Source Representation

**GPLSS**
Set Light Source State

**GPQAAF**
Inquire Advanced Attribute Facilities

**GPSPR**
Set Surface Properties

### RCP code

201343491 (X'0C004203')

## GPLSS - Set Light Source State

**GPLSS (*nact*, *act*, *ndea*, *dea*)**

### Purpose

Use **GPLSS** to insert a Set Light Source State structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Light Source State structure element depending on the current edit mode.

This structure element specifies a list of light source indexes to be added to the current light source state and a list of light source indexes to be deleted from the current light source state. Each index points to an entry in the workstation's light source table. Each table entry contains the set of characteristics of a light

source that the graPHIGS API uses at structure traversal time to calculate lighting effects. The characteristics include light source type, light source color, and some light source dependent parameters.

If a light source index in the activation list already exists in the current light source state, it is ignored. If a light source index in the deactivation list does not exist in the current light source state, it is ignored. If the workstation does not support the light source index, then the graPHIGS API ignores the light source index.

The traversal default for light source state is no light sources (an empty list).

**Parameters**

*nact* **— specified by user, fullword integer**
Number of light sources to be added to the current light source state (>=0).

*act* **— specified by user, array of fullword integers**
Light source indexes to be added to the current light source state (>=1).

*ndea* **— specified by user, fullword integer**
Number of light sources to be deleted from the current light source state (>=0).

*dea* **— specified by user, array of fullword integers**
Light source indexes to be deleted from the current light source state (>=1).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**116** NUMBER OF LIGHT SOURCE INDEXES < ZERO

**254** LIGHT SOURCE INDEX < ONE

**256** ACTIVATE LIST AND DEACTIVATE LIST ARE NOT DISJOINT

**Related Subroutines**

**GPBSPR**
Set Back Surface Properties

**GPLMO**
Set Lighting Calculation Mode

**GPLSR**
Set Light Source Representation

**GPSPR**
Set Surface Properties

**RCP code**

201343490 (X'0C004202')

## GPLT - Set Linetype

**GPLT (*ltype*)**

**Purpose**

Use **GPLT** to insert a Set Linetype structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Linetype structure element depending on the current edit mode.

This structure element specifies an index into a workstation line type table that contains line types that the graPHIGS API uses to render all subsequent output primitives to which this attribute applies. At traversal time, the graPHIGS API uses the line type to render the output primitives when the line type aspect source flag is set to 2=INDIVIDUAL.

The traversal default for line type is 1=SOLID_LINE.

If the workstation does not support the specified line type, then the line type defaults to 1=SOLID_LINE.

### Parameters

*ltype* **— specified by user, fullword integer**
Specifies an index into the workstation's line type table. The table size and specific entries supported are workstation dependent. Use the Inquire Polyline Facilities (**GPQPLF**) subroutine to determine the supported line types on your workstation. The size of the line type table is workstation dependent. The default line type table for supported entries is defined with the following line types: (1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE). Use the Set Linetype Representation (**GPLTR**) subroutine to change any entry except for entry 1.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**63** LINETYPE VALUE < ONE

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPLNR**
Set Linetype Rendering

**GPQPLF**
Inquire Polyline Facilities

**GPQPLR**
Inquire Predefined Polyline Representation

**GPQXLR**
Inquire Extended Polyline Representation

**GPXPLR**
Set Extended Polyline Representation

### RCP code

201328641 (X'0C000801')

## GPLWSC - Set Linewidth Scale Factor

**GPLWSC (*lwidth*)**

### Purpose

Use **GPLWSC** to insert a Set Linewidth Scale Factor structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Linewidth Scale Factor structure element depending on the current edit mode.

This structure element specifies the width of the line as a fraction of the nominal. The device support multiplies this scale factor times the nominal line width on the corresponding device to determine the requested width. The graPHIGS API maps the calculate value to the closest width available on the device. A scale factor of 1.0 generates a nominal size line on any workstation. At structure traversal, the graPHIGS API uses this scale factor when the line width scale factor aspect source flag is set to `2=INDIVIDUAL`.

The traversal default for linewidth scale factor is a linewidth scale factor of 1.0.

**Parameters**

*lwidth* **— specified by user, short floating-point number**
Line width scale factor.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPQPLF**
Inquire Polyline Facilities

**GPQPLR**
Inquire Predefined Polyline Representation

**GPQXLR**
Inquire Extended Polyline Representation

**GPXPLR**
Set Extended Polyline Representation

**RCP code**

201328642 (X'0C000802')

## GPMSSC - Set Marker Size Scale Factor

**GPMSSC (*msize*)**

**Purpose**

Use **GPMSC** to insert a Set Marker Size Scale Factor structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Marker Size Scale Factor structure element depending on the current edit mode.

This element specifies the marker's size as a fraction of the nominal marker size. The device support multiplies this scale factor times the nominal size of markers on the corresponding device to determine the requested size. The graPHIGS API maps the calculated value to the closest size available on the device. A scale factor of 1.0 generates a nominal size marker on any workstation. At structure traversal, the graPHIGS API uses this marker size scale factor when the marker size scale factor aspect source flag is set to `2=INDIVIDUAL`.

The traversal default for marker size scale factor is a marker size scale factor of 1.0.

### Parameters

*msize* **— specified by user, short floating-point number**
Marker size scale factor.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |

### Related Subroutines

| | |
|---|---|
| **GPASF** | Attribute Source Flag Setting |
| **GPQPMF** | Inquire Polymarker Facilities |
| **GPQPMR** | Inquire Predefined Polymarker Representation |
| **GPQXMR** | Inquire Extended Polymarker Representation |
| **GPXPMR** | Set Extended Polymarker Representation |

### RCP code

201328645 (X'0C000805')

## GPMT - Set Marker Type

**GPMT (*mtype*)**

### Purpose

Use **GPMT** to insert a Set Marker Type structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Marker Type structure element depending on the current edit mode.

This structure element specifies an index into a workstation markertype table that contains marker types that the graPHIGS API uses to render all subsequent polymarker primitives. At structure traversal time, the graPHIGS API uses this marker type to render the polymarker primitives when the marker type aspect source flag is set to `2=INDIVIDUAL`.

The traversal default for marker type is `3=ASTERISK`.

If your workstation does not support the specified entry or the entry is outside the allowable range, then the marker type index defaults to `3=ASTERISK`.

### Parameters

*mtype* **— specified by user, fullword integer**
Specifies an index into the marker type table of the workstation. The table size and specific entries supported are workstation dependent. Use the Inquire Polymarker Facilities (**GPQPMF**) subroutine to determine the supported marker types on your workstation. The default marker type table for supported entries is defined with the following marker types: (`1=DOT`, `2=PLUS_SIGN`, `3=ASTERISK`, `4=CIRCLE`, `5=DIAGONAL_CROSS`, `6-n=ASTERISK`). Use the Set Marker Type Representation (**GPMTR**) subroutine to change any entry except entry 3.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |

**69** MARKER TYPE VALUE < ONE

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPQPMF**
Inquire Polymarker Facilities

**GPQPMR**
Inquire Predefined Polymarker Representation

**GPQXMR**
Inquire Extended Polymarker Representation

**GPXPMR**
Set Extended Polymarker Representation

**RCP code**

201328644 (X'0C000804')

## GPPGC - Set Polygon Culling

**GPPGC (*mode*)**

**Purpose**

Use **GPPGC** to insert a Set Polygon Culling structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polygon Culling structure element depending on the current edit mode.

This structure element specifies a mode. This mode specifies whether or not a given area geometry including its boundaries and edges should be visualized when rendering subsequent primitives to which this attribute applies.

The traversal default for polygon culling is `1=NONE`.

If the workstation does not support the specified mode, then polygon culling mode defaults to `1=NONE`.

**Parameters**

*mode* **— specified by user, fullword integer**
Polygon culling mode (`1=NONE, 2=BACK, 3=FRONT`)

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**87** POLYGON CULLING MODE IS INVALID

**Related Subroutines**

**GPQAAF**
Inquire Advanced Attribute Facilities

**RCP code**

201343237 (X'0C004105')

## GPPHEC - Set Polyhedron Edge Culling

**GPPHEC (*mode*)**

### Purpose

Use **GPPHEC** to insert a Set Polyhedron Edge Culling structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polyhedron Edge Culling structure element depending on the current edit mode.

During structure traversal, this structure element sets the current polyhedron edge culling mode entry in the graPHIGS API traversal state list to the value specified by the parameter. The graPHIGS API uses this value to create subsequent polyhedron edge output primitives. This value supplies polyhedron edge culling information to the workstation. This structure element specifies a mode which defines which orientation of the adjacent area geometries causes the line segments to not be visualized.

A polyhedron edge is a line segment which is the shared edge of two area geometries. A polyhedron edge has two normals representing the geometric normals of the two adjacent areas. Based on the direction of each normal relative to the viewer, the graPHIGS API defines each area geometry as front facing or back facing. A culling mode set to `1=NONE` produces no culling (i.e., area geometries are always visualized). A culling mode of `4=BOTH_BACK_OR_BOTH_FRONT` causes the line segment to not be visualized if both adjacent area geometries are either back facing or front facing.

The traversal default for polyhedron edge culling is `1=NONE`.

If the workstation does not support the specified polyhedron edge, then the polyhedron edge mode defaults to `1=NONE`.

### Parameters

*mode* **— specified by user, fullword integer**
Polyhedron edge culling mode (`1=NONE`, `2=BOTH_BACK`, `3=BOTH_FRONT`, `4=BOTH_BACK_BOTH_FRONT`, `5=BACK_AND_FRONT`, `6=LEAST_ONE_BACK`, `7=LEAST_ONE_FRONT`)

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**67** POLYHEDRON EDGE CULLING MODE IS INVALID

### Related Subroutines

**GPQAAF**
Inquire Advanced Attribute Facilities

### RCP code

201343238 (X'0C004106')

## GPPKID - Set Pick Identifier

**GPPKID (*pickid*)**

### Purpose

Use **GPPKID** to insert a Set Pick Identifier structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Pick Identifier structure element depending on the current edit mode.

The pick identifier is associated with all subsequent primitives and is returned in each entry of a pickpath. The returned *pickid* represents the pick identifier that was current when the corresponding structure element was processed.

The traversal default for pick identifier is no pick identifier.

### Parameters

*pickid* **— specified by user, fullword integer**
Pick identifier.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

### Related Subroutines

None

### RCP code

201334787 (X'0C002003')

## GPPLCD - Set Polyline Color Direct

**GPPLCD (*color*)**

### Purpose

Use **GPPLCD** to insert a Set Polyline Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polyline Color Direct structure element depending on the current edit mode.

This structure element specifies the direct color values that the graPHIGS API uses to render subsequent output primitives to which this attribute applies. During structure traversal, the graPHIGS API uses these values to render the output primitives when the polyline color aspect source flag is set to `2=INDIVIDUAL`.

This attribute sets the same traversal state as the Set Polyline Color Index (**GPPLCI**) subroutine. The traversal default for polyline color is the content of entry 1 of the rendering color table.

### Parameters

*color* **— specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPPLCI**
Set Polyline Color Index

**GPQXLR**
Inquire Extended Polyline Representation

**GPXPLR**
Set Extended Polyline Representation

### RCP code

201328663 (X'0C000817')

## GPPLCI - Set Polyline Color Index

**GPPLCI (*index*)**

### Purpose

Use **GPPLCI** to insert a Set Polyline Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polyline Color Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that defines the color values that the graPHIGS API uses to render all output primitives to which this attribute applies. At structure traversal time, the graPHIGS API uses this index to render the output primitives when the polyline color aspect source flag is set to `2=INDIVIDUAL`.

This attribute sets the same traversal state as the Set Polyline Color Direct (**GPPLCD**) subroutine. The traversal default for polyline color is a color index value of 1.

If the workstation does not support the specified polyline color index, then the polyline color index defaults to a value of 1.

### Parameters

*index* **— specified by user, fullword integer**
Color index (>=0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**92** COLOR INDEX < ZERO

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPPLCD**
Set Polyline Color Direct

**GPQPLF**
Inquire Polyline Facilities

**GPQPLR**
Inquire Predefined Polyline Representation

**GPQXLR**
Inquire Extended Polyline Representation

**GPXPLR**
Set Extended Polyline Representation

**RCP code**

201328643 (X'0C000803')

## GPPLET - Set Polyline End Type

**GPPLET (*endtype*)**

**Purpose**

Use **GPPLET** to insert a Set Polyline End Type structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polyline End Type structure element depending on the current edit mode.

Flat ends are the default setting. The graPHIGS API places flat ends at the line endpoints. Round ends are semicircles centered at the line ends. The diameter of a round end is equal to the width of the line. Square ends extend half a line width past the line's endpoints and are parallel to the last line segment.

The traversal default for polyline end type is `1=FLAT`.

If the workstation does not support the specified polyline end type, then the polyline edge type defaults to `1=FLAT`.

**Parameters**

*endtype* **— specified by user, fullword integer**
Polyline end type (`1=FLAT, 2=ROUND, 3=SQUARE`)

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**584** END TYPE VALUE < ONE

**Related Subroutines**

**GPQAAF**
Inquire Advanced Attribute Facilities

**RCP code**

201328658 (X'0C000812')

## GPPLI - Set Polyline Index

**GPPLI (*index*)**

### Purpose

Use **GPPLI** to insert a Set Polyline Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polyline Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's polyline bundle table. The entry contains attribute settings for line type, line width scale factor, and color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent polyline primitives for those attributes that have an aspect source flag set to `1=BUNDLED`.

The traversal default value for polyline index is 1.

If the workstation does not support the specified polyline index, then the polyline index defaults to a value of 1.

### Parameters

*index* **— specified by user, fullword integer**
Polyline bundle table index (>=1)

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**60** BUNDLE INDEX VALUE < ONE

### Related Subroutines

**GPQLW**
Inquire Length of Workstation State Tables

**GPQXLR**
Inquire Extended Polyline Representation

**GPXPLR**
Set Extended Polyline Representation

### RCP code

201328385 (X'0C000701')

## GPPLSM - Set Polyline Shading Method

**GPPLSM (*method*)**

### Purpose

Use **GPPLSM** to insert a Set Polyline Shading Method structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polyline Shading Method structure element depending on the current edit mode.

During structure traversal, the graPHIGS API sets the current polyline shading method in the traversal state list and uses this shading method when rendering subsequent Polyline Set 3 With Data (**GPPLD3**) primitives. If the specified polyline shading method is `1=POLYLINE_SHADING_NONE` and Polyline Set 3 With Data specifies vertex colors in its primitive definition, then the graPHIGS API uses the $i^{th}$ vertex color to color the $i^{th}$ line of the polyline. If the current polyline shading method is `2=POLYLINE_SHADING_COLOR` and Polyline Set 3 With Data specifies vertex colors in its primitive definition, then the graPHIGS API

interpolates the color along each line between the colors specified at the endpoints of the line. If the **GPPLSM** primitive definition does not specify a vertex color, then the graPHIGS API renders the entire primitive using the current polyline color and **GPPLSM** has no effect.

The traversal default for polyline shading method is 1=POLYLINE_SHADING_NONE.

If the specified polyline shading method is not within the allowable range, then the shading method defaults to a value of 1=POLYLINE_SHADING_NONE.

### Parameters

*method* **— specified by user, fullword integer**
Specifies the polyline shading method to be used with Polyline Set 3 With Data primitives (1=POLYLINE_SHADING_NONE, 2=POLYLINE_SHADING_COLOR).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**95** POLYLINE SHADING METHOD IS INVALID

### Related Subroutines

**GPPLD3**
Polyline Set 3 With Data

### RCP code

201343504 (X'0C004210')

## GPPMCD - Set Polymarker Color Direct

**GPPMCD (*color*)**

### Purpose

Use **GPPMCD** to insert a Set Polymarker Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polymarker Color Direct structure element depending on the current edit mode.

This structure element specifies the direct color values that the graPHIGS API uses to render subsequent polymarker primitives. At structure traversal time, the graPHIGS API uses these values to render the polymarker primitives when the polymarker color aspect source flag is set to 2=INDIVIDUAL.

This attribute sets the same traversal state as the Set Polymarker Color Index (**GPPMCI**) subroutine. The traversal default for polymarker color is the content of entry 1 of the rendering color table.

### Parameters

*color* **— specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0)

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPQXMR**
Inquire Extended Polymarker Representation

**GPXPMR**
Set Extended Polymarker Representation

### RCP code

201328664 (X'0C000818')

## GPPMCI - Set Polymarker Color Index

**GPPMCI (*index*)**

### Purpose

Use **GPPMCI** to insert a Set Polymarker Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polymarker Color Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that defines the color values that the graPHIGS API uses to render all polymarker primitives. At structure traversal time, the graPHIGS API uses this index to render the polymarker primitives when the polymarker color aspect source flag is set to `2=INDIVIDUAL`.

This attribute sets the same traversal state as the Set Polymarker Color Direct (**GPPMCD**) subroutine. The traversal default for polymarker color is a color index value of 1.

If the workstation does not support the specified polymarker color, then the polymarker color index defaults to a value of 1.

### Parameters

*index* **— specified by user, fullword integer**
Color index (>=0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**92** COLOR INDEX < ZERO

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPPMCD**
Set Polymarker Color Direct

**GPQPMF**
Inquire Polymarker Facilities

**GPQPMR**
Inquire Predefined Polymarker Representation

**GPQXMR**
Inquire Extended Polymarker Representation

**GPXPMR**
Set Extended Polymarker Representation

**RCP code**

201328646 (X'0C000806')

## GPPMI - Set Polymarker Index

**GPPMI (*index*)**

**Purpose**

Use **GPPMI** to insert a Set Polymarker Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Polymarker Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's polymarker bundle table. The entry contains attribute settings for marker type, marker size scale factor, and color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent polymarker primitives for the attributes that have an aspect source flag set to `1=BUNDLED`.

The traversal default value for polymarker index is 1.

If the workstation does not support the specified polymarker index, then the polymarker index defaults to a value of 1.

**Parameters**

*index* **— specified by user, fullword integer**
Polymarker bundle table index (>=1).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**60** BUNDLE INDEX VALUE < ONE

**Related Subroutines**

**GPQLW**
Inquire Length of Workstation State Tables

**GPQXMR**
Inquire Extended Polymarker Representation

**GPXPMR**
Set Extended Polymarker Representation

**RCP code**

201328386 (X'0C000702')

# GPPSC - Parametric Surface Characteristics

**GPPSC (*type*, *data*)**

### Purpose

Use **GPPSC** to insert a Parametric Surface Characteristics structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Parametric Surface Characteristics structure element depending upon the current edit mode.

During structure traversal, the graPHIGS API uses the *type* and *data* parameters to update the Parametric Surface Characteristic entry in the graPHIGS API traversal state list. The graPHIGS API uses this new state to render all subsequent parametric surface primitives until this traversal state list entry is changed.

The Parametric Surface Characteristic traversal state defines the type of line geometry that is generated in the interior of a parametric surface. When rendering a line geometry, all polyline attributes are applied to the line representation.

When rendering a parametric surface, edges (which are controlled by edge attributes), have priority over line geometries, which have priority over rendering surface interiors (which are controlled by interior attributes).

The traversal default for parametric surface characteristics is `1=NONE`. If a workstation encounters an unsupported *type*, the parametric surface defaults to `1=NONE`.

**GPPSC** is identified as GSE 1008.

> **Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Generalized Structure Elements (**GPQGSE**) subroutine determine the GSEs which are supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

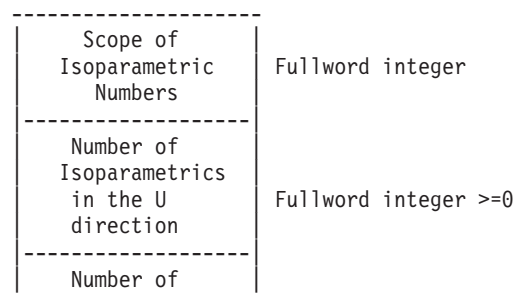
### Parameters

*type* — **specified by user, fullword integer**
Rendering style that is to be applied to the interior of parametric surfaces (`1=NONE`, `2=ISOPARAMETRIC_LINES`).

*data* — **specified by user, array of integers**
Rendering style type dependent data. The required data for each type is listed below. The parameters must be specified in the order shown.

**If *type*=1 ( `NONE`)**
The *data* parameter is ignored.

**If *type*=2 ( `ISOPARAMETRIC_LINES`)**
The *data* parameter requires the following format:



```
  8 |   Isoparametrics   |
    |    in the V        | Fullword integer >=0
    |    direction       |
    ----------------------
```

***Scope of Isoparametric Numbers***
This field defines whether the number of Isoparametrics is to be applied to the surface as a whole or just between knots (`1=Surface, 2=Between Knots`). If the surface is uniform and `2=Between Knots` is specified, `1=Surface` is used instead.

***Number of Isoparametrics in the U Direction***
These curves are drawn parallel to the v-axis of parameter space. For scope of `1=Surface`, this field defines the number of isoparametric curves that are drawn between minimum and maximum parameter limits of the surface. For an untrimmed surface, a value of one would result in an isoparametric curve that is drawn at (vmin+vmax)/2. For a scope of `2=Between Knots`, this field defines the number of isoparametric curves that are to be drawn between knots in addition to isoparametric curves through the knot lines. A value of zero would result in an isoparametric curve being drawn at each knot line within the parameter limits of the surface. A value of one would add a curve placed midway between each pair of knots.

***Number of Isoparametrics in the V Direction***
This parameter is identical to the previous parameter except the isoparametric curves are drawn parallel to the u-axis.

> **Note:** The graPHIGS API only displays isoparametric curves in the active region of a trimmed surface. This means that the trimming curves clip the isoparametric curves.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**102** PARAMETRIC SURFACE CHARACTERISTICS TYPE IS INVALID

**103** PARAMETRIC SURFACE CHARACTERISTICS DATA IS INVALID

**Related Subroutines**

**GPNBC2**
Non-Uniform B-Spline Curve 2

**GPNBC3**
Non-Uniform B-Spline Curve 3

**GPNBS**
Non-Uniform B-Spline Surface

**GPSPH**
Polysphere

**GPTNBS**
Trimmed Non-Uniform B-Spline Surface

**RCP code**

201328907 (X'0C00090B')

## GPRCN - Remove Class Name from Set

**GPRCN (*number*, *names*)**

### Purpose

Use **GPRCN** to insert a Remove Class Name from Set structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Remove Class Name from Set structure element depending on the current edit mode.

The class set traversal state consists of a list of class names. During structure traversal, this structure element removes one or more names from the list but does not completely replace the traversal state as other attributes do.

Class names let an application control the eligibility of a primitive for pickability (detectability), highlighting, and invisibility by associating it with a class set. The effects of adding or removing a class name to the current class set are inherited (i.e., by child structures).

When the graPHIGS API encounters a primitive during structure traversal, it uses the list of class names in the class set to determine the pickability (detectability), highlighting, and invisibility aspects. If the workstation does not support a specified name, then the graPHIGS API ignores the name and the name has no effect on the primitive.

Also use class names are to create inclusion and exclusion filters for the specified workstation. The graPHIGS API uses these filters in conjunction with the class set traversal state to determine if pickability, highlighting, and visibility apply. The filters act independently of each other.

During structure traversal, the graPHIGS API compares the current class set to the current filters. When root structure traversal begins, the current class set is null.

For a complete discussion of class names and filters, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*number* **— specified by user, fullword integer**
Number of class names to remove from the class set (>=0).

*names* **— specified by user, array of fullword integers**
Array of class names to remove from the class set (class names must be >=0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**320** CLASS NAME VALUE IS INVALID

**530** NUMBER OF CLASS NAMES < ZERO

### Related Subroutines

**GPADCN**
Add Class Name to Set

**GPQNCN**
Inquire Number of Available Class Names

### RCP code

201334786 (X'0C002002')

# GPRMO - Set Reflectance Model

**GPRMO (*model*)**

### Purpose

Use **GPRMO** to insert a Set Reflectance Model structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Reflectance Model structure element, depending on the current edit mode.

This element specifies the lighting reflectance model that the graPHIGS API uses when performing lighting calculations on subsequent area primitives. If the face distinguish mode (**GPFDMO**) is `1=NONE`, then the graPHIGS API uses the specified model to calculate the lighting effects on both front and back facing portions of area primitives. If the face distinguish mode is `2=COLOR_SURFACE_PROPERTIES`, then the graPHIGS API uses the specified model to calculate the lighting effects on only front facing portions of area primitives.

The defined reflectance models and their effect are as follows:

**1=REFLECTANCE_NONE**
No reflectance calculation is performed.

**2=AMB** Ambient reflectance effects are computed.

**3=AMB_DIFF**
Ambient and diffuse reflectance effects are computed.

**4=AMB_DIFF_SPEC**
Ambient, diffuse, and specular reflectance effects are computed.

For compatibility, the Set Reflectance Model traversal state is also set by the Set Lighting Calculation Mode (**GPLMO**) subroutine. For more information, see *The graPHIGS Programming Interface: Understanding Concepts*.

The traversal default for the reflectance model is `1=REFLECTANCE_NONE`. If the workstation does not support the specified method, then the graPHIGS API uses `1=REFLECTANCE_NONE`.

### Parameters

*model* **— specified by user, fullword integer**
Reflectance model (`1=REFLECTANCE_NONE, 2=AMB, 3=AMB_DIFF, 4=AMB_DIFF_SPEC`).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**110** REFLECTANCE MODEL IS INVALID

### Related Subroutines

**GPBRMO**
Set Back Reflectance Model

**GPQAAF**
Inquire Advanced Attribute Facilities

### RCP code

201343509 (X'0C004215')

# GPSAC - Set Surface Approximation Criteria

**GPSAC (*criteria*, *ctrlval1*, *ctrlval2*)**

### Purpose

Use **GPSAC** to insert a Set Surface Approximation Criteria structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Surface Approximation Criteria structure element depending on the current edit mode.

This structure element determines how the graPHIGS API tessellates subsequent surfaces during structure traversal. *Tessellation* divides the surfaces into a set of line and/or area geometries for subsequent processing.

Depending on the criteria selected, the graPHIGS API uses only the control values or uses the control values in conjunction with the tessellation vectors in the surface definition to determine how surfaces are tessellated.

The traversal default for surface approximation criteria is `1=WORKSTATION_DEPENDENT` with the *u* control value (*ctrlval1*) and *v* control value (*ctrlval2*) parameter values both set to 1.0.

If the workstation does not support the specified criteria, then the criteria defaults to `1=WORKSTATION_DEPENDENT`, with *ctrlval1* and *ctrlval2* values both set to 1.0.

For information on surface tessellation and on how surface approximation criteria is applied, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*criteria* **— specified by user, fullword integer**
Surface approximation criteria (`1=WORKSTATION_DEPENDENT`, `3=CONSTANT_SUBDIVISION_BETWEEN_KNOTS`, `8=VARIABLE_SUBDIVISION_BETWEEN_KNOTS`).

*ctrlval1* **— specified by user, short floating-point number**
U-control value (>=0).

*ctrlval2* **— specified by user, short floating-point number**
V-control value (>=0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**66** CONTROL VALUE < ZERO

**86** SURFACE APPROXIMATION CRITERIA IS INVALID

### Related Subroutines

**GPQSDF**
Inquire Surface Display Facilities

### RCP code

201343236 (X'0C004104')

## GPSCD - Set Specular Color Direct

**GPSCD (*color*)**

### Purpose

Use **GPSCD** to insert a Set Specular Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Specular Color Direct structure element depending on the current edit mode.

This structure element specifies the color values that the graPHIGS API uses at structure traversal time for the specular highlights produced by lighting calculations on area geometries.

If the face distinguish mode is `1=NONE`, then the graPHIGS API uses the color values on both the front facing and back facing surfaces. If the face distinguish mode is set to `2=COLOR_SURFACE_PROPERTIES`, then the graPHIGS API uses the color values on the front facing portions only.

This attribute sets the same traversal state as the Set Specular Color Index (**GPSCI**) subroutine. The traversal default for specular color is the content of entry 1 of the rendering color table.

### Parameters

*color* **— specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

### Related Subroutines

**GPFDMO**
Set Face Distinguish Mode

**GPSCI**
Set Specular Color Index

### RCP code

201343497 (X'0C004209')

## GPSCI - Set Specular Color Index

**GPSCI (*index*)**

### Purpose

Use **GPSCI** to insert a Set Specular Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Specular Color Index structure element depending on the current edit mode.

This structure element specifies the color values that the graPHIGS API uses at structure traversal time for the specular highlights produced by lighting calculations on area geometries.

If the face distinguish mode is 1=NONE, then the graPHIGS API uses the color values on both the front facing and back facing surfaces. If the face distinguish mode is set to 2=COLOR_SURFACE_PROPERTIES, then the graPHIGS API uses the color values on the front facing portions only.

This attribute sets the same traversal state as the Set Specular Color Direct (**GPSCD**) subroutine. The traversal default for specular color is a color index value of 1.

If the workstation does not support the specified specular color index, then the color index defaults to a value of 1.

**Parameters**

*index* **— specified by user, fullword integer**
Color index (>=0).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**92** COLOR INDEX < ZERO

**Related Subroutines**

**GPFDMO**
Set Face Distinguish Mode

**GPSCD**
Set Specular Color Direct

**RCP code**

201343496 (X'0C004208')

## GPSPR - Set Surface Properties

**GPSPR (*amb*, *diff*, *spec*, *exp*, *trans*)**

**Purpose**

Use **GPSPR** to insert a Set Surface Properties structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Surface Properties structure element depending on the current edit mode.

During structure traversal, this structure element specifies the coefficients and exponents which the graPHIGS API uses in calculating lighting and transparency effects.

If face distinguish mode (**GPFDMO**) is set to 1=NONE, then the graPHIGS API uses these values to calculate the lighting and transparency effects on both the front and back facing portions of a surface. If face distinguish mode is set to 2=COLOR_SURFACE_PROPERTIES, then the graPHIGS API uses these values to calculate the effects on only the front facing portions of a surface.

The traversal default for the coefficients and exponent for the surface properties are as follows:

- *amb* = 1.0 (ambient reflection coefficient)
- *diff* = 1.0 (diffuse reflection coefficient)
- *spec* = 1.0 (specular reflection coefficient)
- *exp* = 0.0 (specular reflection exponent) (i.e., no specular effect)

- *trans* = 0.0 (transparency coefficient) (i.e., opaque)

The transparency mode (**GPXVR**) for the view must be set to a value other than `1=NONE` for the graPHIGS API to use the specified transparency coefficient.

The current transparency coefficient in the graPHIGS API Traversal State List is also set by the transparency coefficient of the Set Transparency Coefficient (**GPTCO**) subroutine.

**Parameters**

*amb* **— specified by user, short floating-point number**
Ambient reflection coefficient (0<= *amb*<=1).

*diff* **— specified by user, short floating-point number**
Diffuse reflection coefficient (0<= *diff*<=1).

*spec* **— specified by user, short floating-point number**
Specular reflection coefficient (0<= *spec*<=1).

*exp* **— specified by user, short floating-point numbers**
Specular reflection exponent (>=0).

*trans* **— specified by user, short floating-point number**
Transparency coefficient (0<= *trans*<=1).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**111** AMBIENT COEFFICIENT IS INVALID

**112** DIFFUSE COEFFICIENT IS INVALID

**113** SPECULAR COEFFICIENT IS INVALID

**114** SPECULAR EXPONENT IS INVALID

**115** TRANSPARENT COEFFICIENT IS INVALID

**Related Subroutines**

**GPBSPR**
Set Back Surface Properties

**GPLMO**
Set Lighting Calculation Mode

**GPLSR**
Set Light Source Representation

**GPLSS**
Set Light Source State

**GPRMO**
Set Reflectance Model

**GPTCO**
Set Transparency Coefficient

**RCP code**

201343493 (X'0C004205')

# GPTCAC - Set Trimming Curve Approximation Criteria

**GPTCAC (*criteria*, *ctrlval1*, *ctrlval2*, *ctrlval3*)**

### Purpose

Use **GPTCAC** to insert a Set Trimming Curve Approximation Criteria structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Trimming Curve Approximation Criteria structure element depending on the current edit mode.

This structure element enables the application to control the tessellation of the trimming curve as well as the surface in the area of the curve when rendering subsequent trimmed surface primitives during structure traversal.

Depending on the criteria selected, the graPHIGS API only uses the control values or uses the control values in conjunction with the tessellation vector in the trimmed surface primitive definition to determine how the trimming curve is tessellated.

The traversal default for trimming curve approximation criteria is `1=WORKSTATION_DEPENDENT` with the control value (*ctrlval1*), the *u* control value (*ctrlval2*), and *v* control value (*ctrlval3*) all set to a value of 1.0.

If the workstation does not support the specified trimming curve approximation criteria, then the default trimming curve criteria is `1=WORKSTATION_DEPENDENT` with *ctrlval1*, *ctrlval2*, and *ctrlval3* all set to a value of 1.0.

For information on trimmed surfaces and how trimming curve approximation criteria is applied, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*criteria* **— specified by user, fullword integer**
Curve approximation criteria (`1=WORKSTATION_DEPENDENT`, `3=CONSTANT_SUBDIVISION_BETWEEN_KNOTS`, `8=VARIABLE_SUBDIVISION_BETWEEN_KNOTS`).

*ctrlval1* **— specified by user, short floating-point number**
Control value (>=0).

*ctrlval2* **— specified by user, short floating-point number**
*u*-Control value (>=0).

*ctrlval3* **— specified by user, short floating-point number**
*v*-Control value (>=0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**65** CURVE APPROXIMATION CRITERIA IS INVALID

**66** CONTROL VALUE < ZERO

### Related Subroutines

**GPQTDF**
Inquire Trimming Curve Display Facilities

### RCP code

201328906 (X'0C00090A')

## GPTCO - Set Transparency Coefficient

**GPTCO (*coeff*)**

### Purpose

Use **GPTCO** to insert a Set Transparency Coefficient structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Transparency Coefficient structure element, depending upon the current edit mode.

During traversal, the value of *coeff* specifies the source transparency coefficient of subsequent primitives. The use of the transparency coefficient depends on the Transparency Processing Mode of the view table entry.

If face distinguish mode (**GPFDMO**) is `1=NONE`, then the graPHIGS API uses this value to calculate the transparency effects on both front and back facing portions of area primitives. If face distinguish mode is `2=COLOR_SURFACE_PROPERTIES`, then graPHIGS API uses this value to calculate the transparency effects on only front facing portions of area primitives.

The current transparency coefficient in the graPHIGS API Traversal State List is also set by the transparency coefficient of the Set Surface Properties (**GPSPR**) subroutine.

The traversal default for the transparency coefficient value is 0.0.

### Parameters

*coeff* **— specified by user, short floating-point number**
Transparency coefficient (0.0<=*coeff*<=1.0).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**115** TRANSPARENT COEFFICIENT IS INVALID

### Related Subroutines

**GPBSPR**
Set Back Surface Properties

**GPBTCO**
Set Back Transparency Coefficient

**GPFDMO**
Set Face Distinguish Mode

**GPSPR**
Set Surface Properties

### RCP code

201343505 (X'0C004211')

## GPTXAL - Set Text Alignment

**GPTXAL (*horiz*, *vert*)**

### Purpose

Use **GPTXAL** to insert a Set Text Alignment structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Text Alignment structure element depending on the current edit mode.

At structure traversal time, the graPHIGS API uses the specified alignment in this structure element to render all subsequent geometric text primitives. This setting affects the manner in which the graPHIGS API positions the geometric text extent rectangle in relation to the text position.

The traversal default for geometric text horizontal and vertical alignment is `1=NORMAL`.

If the workstation does not support the specified text alignment values, then the text alignment defaults to `1=NORMAL` for both horizontal and vertical text alignment.

### Parameters

*horiz* **— specified by user, fullword integer**
Horizontal alignment (`1=NORMAL, 2=LEFT_ALIGN, 3=CENTER, 4=RIGHT_ALIGN`).

*vert* **— specified by user, fullword integer**
Vertical alignment (`1=NORMAL, 2=TOP, 3=CAP, 4=HALF, 5=BASE, 6=BOTTOM`).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**309** TEXT ALIGNMENT COMPONENT IS INVALID

### Related Subroutines

**GPTXPR**
Set Text Precision

### RCP code

201328908 (X'0C00090C')

## GPTXCD - Set Text Color Direct

**GPTXCD (*color*)**

### Purpose

Use **GPTXCD** to insert a Set Text Color Direct structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Text Color Direct structure element depending on the current edit mode.

This structure element specifies the direct color values that the graPHIGS API uses to render the subsequent text primitives. At structure traversal time, the graPHIGS API uses these values to render all subsequent annotation and geometric text primitives when the text color aspect source flag is set to `2=INDIVIDUAL`.

This attribute sets the same traversal state as the Set Text Color Index (**GPTXCI**) subroutine. The traversal default is the content of entry 1 of the rendering color table.

### Parameters

*color* **— specified by user, 3 short floating-point numbers**
Three color components of the current direct color model in the graPHIGS API state list (0.0<=component<=1.0).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPQXTR**
Inquire Extended Text Representation

**GPTXCI**
Set Text Color Index

**GPXTXR**
Set Extended Text Representation

**RCP code**

201328665 (X'0C000819')

## GPTXCI - Set Text Color Index

**GPTXCI (*index*)**

**Purpose**

Use **GPTXCI** to insert a Set Text Color Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Text Color Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's rendering color table that defines the color that the graPHIGS API uses to render all subsequent annotation and geometric text primitives. At structure traversal time, the graPHIGS API uses this to render the text primitives when the text color aspect source flag is set to `2=INDIVIDUAL`.

This attribute sets the same traversal state as the Set Text Color Direct (**GPTXCD**) subroutine. The traversal default for text color is a color index value of 1.

If the workstation does not support the specified text color index, then the color index defaults to a value of 1.

**Parameters**

*index* **— specified by user, fullword integer**
Color index (>=0).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**92** COLOR INDEX < ZERO

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPQPTR**
Inquire Predefined Text Representation

**GPQXTR**
Inquire Extended Text Representation

**GPTXCD**
Set Text Color Direct

**GPXTXR**
Set Extended Text Representation

### RCP code

201328651 (X'0C00080B')

## GPTXFO - Set Text Font

**GPTXFO (*font*)**

### Purpose

Use **GPTXFO** to insert a Set Text Font structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Text Font structure element depending on the current edit mode.

This structure element specifies a font identifier that the graPHIGS API uses to render all subsequent annotation and geometric text primitives. At structure traversal time, the graPHIGS API uses this identifier when the text font aspect source flag is set to `2=INDIVIDUAL`.

The traversal default for annotation and geometric text font is font 1.

### Parameters

*font* **— specified by user, fullword integer**
Specifies the text font to be used (>=1).

If a value is encountered that is unavailable (unsupported or inactive), font 1 of that character set is substituted. If that font is unavailable, font 1 of the primary character set is substituted at the same precision.

For a complete list of supported fonts, see *The graPHIGS Programming Interface: Technical Reference*. For specific workstation support, use Inquiry programming subroutines (Chapter 16. "Inquire Subroutines" or see *The graPHIGS Programming Interface: Technical Reference*).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**75** TEXT FONT VALUE IS INVALID

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPQPTR**
Inquire Predefined Text Representation

**GPQXTR**
Inquire Extended Text Representation

**GPTXCS**
Set Text Character Set

**GPXTXR**
Set Extended Text Representation

**RCP code**

201328647 (X'0C000807')

## GPTXI - Set Text Index

**GPTXI (*index*)**

**Purpose**

Use **GPTXI** to insert a Set Text Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Text Index structure element depending on the current edit mode.

This structure element specifies an entry in the workstation's text bundle table. The entry contains attribute settings for text font, text precision, character expansion factor, character spacing, and color. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent annotation and geometric text primitives for those attributes that have an aspect source flag set to `1=BUNDLED`.

The traversal default text index value is 1.

If the workstation does not support the specified text index, then the text index defaults to a value of 1.

**Parameters**

*index* **— specified by user, fullword integer**
Text bundle table index (>=1).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**60** BUNDLE INDEX VALUE < ONE

**Related Subroutines**

**GPQLW**
Inquire Length of Workstation State Tables

**GPQPTR**
Inquire Predefined Text Representation

**GPQXTR**
Inquire Extended Text Representation

**GPQXTX**
Inquire Extended Text Facilities

**GPXTXR**
Set Extended Text Representation

**RCP code**

201328387 (X'0C000703')

# GPTXPR - Set Text Precision

**GPTXPR (*prec*)**

## Purpose

Use **GPTXPR** to insert a Set Text Precision structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Text Precision structure element depending on the current edit mode.

The text precision specifies which attributes apply to annotation and geometric text primitives and the manner in which the graPHIGS API uses them. At structure traversal time, the graPHIGS API uses this precision when the text precision aspect source flag is set to 2=INDIVIDUAL.

The traversal default for text precision is 1=STRING_PREC.

If the workstation does not support the specified text precision, then the graPHIGS API uses the highest available precision in the font instead.

## Parameters

*prec* — **specified by user, fullword integer**
Text precision (1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC).

The following figure describes the attributes and precision for geometric text:

Geometric Text Attributes

| Precision | TEXT FONT | CHARACTER EXPANSION FACTOR | CHARACTER SPACING | COLOR* | CHARACTER HEIGHT | CHARACTER UP VECTOR | TEXT PATH | TEXT ALIGNMENT | CHARACTER UP AND BASE VECTORS | CHARACTER POSITIONING MODE |
|---|---|---|---|---|---|---|---|---|---|---|
| STRING | Y ** | N | N | Y | Y | N | N | N | N | Y *** |
| CHAR | Y ** | Y | Y | Y | Y | Y | Y | Y | Y | Y *** |
| STROKE | Y ** | Y | Y | Y | Y | Y | Y | Y | Y | Y *** |

* Color refers to both Set Text Color Direct and Set Text Color Index.

**Note:** The following keywords are used above to designate which attributes will be processed for a particular precision:

**Y** The attribute is applied for this precision.

**N** The attribute is not applied for this precision.

****** The requested font will be applied if it is available on the requested workstation and if it was activated using the Activate Font (**GPACFO**) subroutine. Otherwise, the workstation will default to an alternate font.

******* The character positioning mode will be applied if the font has the requested character information.

The following figure describes the attributes and precision for annotation text:

Annotation Text Attributes

| Precision | TEXT FONT | CHARACTER EXPANSION FACTOR | CHARACTER SPACING | COLOR* | ANNOTATION HEIGHT SCALE FACTOR | ANNOTATION UP VECTOR | ANNOTATION PATH | ANNOTATION ALIGNMENT | ANNOTATION HEIGHT | CHARACTER POSITIONING MODE |
|---|---|---|---|---|---|---|---|---|---|---|
| STRING | Y 4 | N | N | Y | Y 1 | N | N | N | Y 1 | Y 5 |
| CHAR | Y 4 | Y 1 | Y 2 | Y | Y 1 | Y 2 | Y 2 | Y 2 | Y 1 | Y 5 |
| STROKE | Y 4 | Y 3 | Y 3 | Y | Y 3 | Y 3 | Y 3 | Y 3 | Y 3 | Y 5 |

* Color refers to both Set Text Color Direct and Set Text Color Index.

**Note:** The following keywords are used above to designate which attributes will be processed for a particular precision:

**Y** The attribute is applied for this precision.

**N** The attribute is not applied for this precision.

The following numbers are used above to describe how precisely an attribute will be applied.

**1** The attribute is applied a closely as possible for the entire text string.

**2** Whether these attributes are applied is workstation dependent. See *The graPHIGS Programming Interface: Technical Reference* for more information.

**3** The attribute is applied on a stroke-by-stroke basis, that is, exactly.

**4** The requested font will be applied if the font is available in the requested workstation; otherwise, the workstation will default to an alternate font.

**5** The character positioning mode will be applied if the font has the requested character information.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**305** TEXT PRECISION VALUE IS INVALID

### Related Subroutines

**GPAAL**
Set Annotation Alignment

**GPAPT**
Set Annotation Path

**GPASF**
Attribute Source Flag Setting

**GPAUP**
Set Annotation Up Vector

**GPCHSP**
Set Character Spacing

**GPCHUB**
Set Character Up and Base Vectors

**GPCHUP**
Set Character Up Vector

**GPQGFC**
Inquire Geometric Font Characteristics

**GPQPTR**
Inquire Predefined Text Representation

**GPQXAF**
Inquire Extended Annotation Font Characteristics

**GPQXTR**
Inquire Extended Text Representation

**GPTXAL**
Set Text Alignment

**GPTXPT**
Set Text Path

**GPXTXR**
Set Extended Text Representation

### RCP code

201328648 (X'0C000808')

## GPTXPT - Set Text Path

**GPTXPT (*path*)**

### Purpose

Use **GPTXPT** to insert a Set Text Path structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Text Path structure element depending on the current edit mode.

This structure element specifies the writing direction of the text string relative to the Character Up Vector. At structure traversal time, the graPHIGS API uses this path value to render all subsequent geometric text primitives.

The traversal default for text path is `1=RIGHT`.

If the workstation does not support the specified text path value, then the text path defaults to `1=RIGHT`.

### Parameters

*path* **— specified by user, fullword integer**
Specifies text path (`1=RIGHT`, `2=LEFT`, `3=UP`, `4=DOWN`).

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**306** TEXT PATH VALUE IS INVALID

### Related Subroutines

**GPTXPR**
Set Text Precision

### RCP code

201328654 (X'0C00080E')

## GPVWI - Set View Index

**GPVWI (*index*)**

### Purpose

Use **GPVWI** to insert a Set View Index structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set View Index structure element depending on the current edit mode.

During structure traversal, this structure element specifies an entry in the workstation view table. The entry contains attribute settings for the view orientation matrix, view mapping matrix, viewport boundaries, and viewport clipping indicators. At structure traversal time, the graPHIGS API uses these attribute settings to render all subsequent output primitives.

The traversal default for the view index is entry 0 of the workstations's view table.

### Parameters

*index* **— specified by user, fullword integer**
An index which specifies an entry into the workstation's view table.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**59** VIEW INDEX VALUE < ZERO

### Related Subroutines

**GPXVR**
Set Extended View Representation

### RCP code

201329154 (X'0C000A02')

## GPZBM - Set Z-Buffer Protect Mask

**GPZBM (*mask*)**

### Purpose

Use **GPZBM** to insert a Set Z-Buffer Protect Mask structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Z-Buffer Protect Mask structure element depending on the current edit mode.

During structure traversal, this structure elements sets the Z-Buffer Protect Mask entry of the graPHIGS API traversal state to the value specified by the parameter.

The mask value specifies a write-protect mask for the z-buffer to inhibit writing into the z-buffer during the rendering of output primitives. If a z-buffer is not present on the workstation, then this attribute is ignored.

The mask is a 32-bit integer. The mask supports two values:

- All bits 0 - This value allows updates to the z-buffer.
- All bits 1 - This value prohibits updates to the z-buffer.

If the *mask* value is not all zeros, then it is processed as if a *mask* value of all ones had been specified.

The traversal default for the z-buffer protect mask is a value of 0 for all 32 bits.

**GPZBM** is identified as GSE 1009.

> **Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Generalized Structure Elements (**GPQGSE**) subroutine to determine the GSEs which are supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*mask* **— specified by user, fullword integer**
z-buffer protect mask. A mask of 0 for all 32 bits allows the application to update the z-buffer. A mask of 1 for all 32 bits prohibits updates to the z-buffer.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**Related Subroutines**

None

**RCP code**

201343243 (X'0C00410B')

# Chapter 5. Miscellaneous Structure Elements

This section describes subroutines which generate structure elements which are *not* related to primitives or primitive attributes. The subroutines in this section generate structure execution and return elements and application data elements used to store application specific information.

## GPCEXS - Conditional Execute Structure

**GPCEXS (*mask*, *crit*, *mode*, *strid*)**

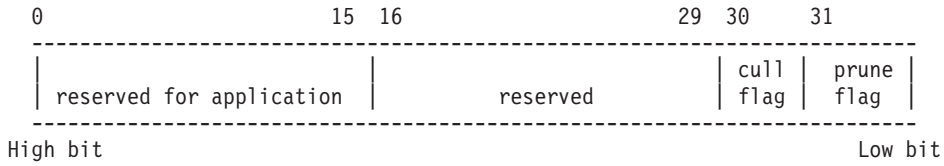
### Purpose

Use **GPCEXS** to insert a Conditional Execute Structure structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Conditional Execute Structure structure element depending on the current edit mode.

If the specified structure identifier does not exist, then the graPHIGS API creates a new empty structure.

If the current condition flags selected by the *mask* parameter satisfy the specified criteria, then the graPHIGS API executes the specified structure according to the specified execution mode. Otherwise, no action is taken, and the element count is incremented.

The condition flag is a 32-bit bit string. Each bit is defined as follows:




**GPCEXS** is identified as GSE 1004.

> **Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Generalized Structure Elements (**GPQGSE**) inquiry subroutine to determine the GSEs which are supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*mask* **— specified by user, fullword integer**
32-bit mask specifying flags to be tested in the current condition flag.

*crit* **— specified by user, fullword integer**
Criteria to be satisfied when the set of flags in the specified mask is tested against the current condition flag (`1=ALL_ONES`, `2=ALL_ZEROS`, `3=NOT_ALL_ONES`, `4=NOT_ALL_ZEROS`).

*mode* **— specified by user, fullword integer**
Execution mode of the child structure (`1=NORMAL`).

*strid* **— specified by user, fullword integer**
Structure identifier.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |

| | |
|---|---|
| **123** | CONDITION CRITERIA IS INVALID |
| **124** | EXECUTE MODE IS INVALID |
| **125** | ATTEMPTING TO EXECUTE THE OPEN STRUCTURE |

### Related Subroutines

| | |
|---|---|
| **GPCOND** | Set Condition |
| **GPCRET** | Conditional Return |
| **GPTEX2** | Test Extent 2 |
| **GPTEX3** | Test Extent 3 |

### RCP code

201331716 (X'0C001404')

## GPCOND - Set Condition

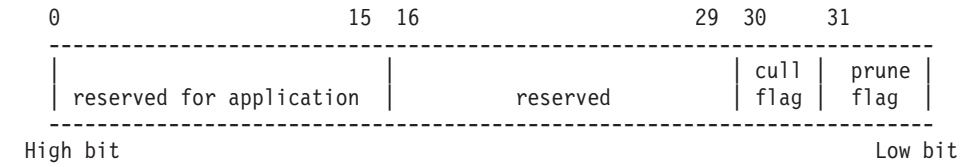
**GPCOND (*on*, *off*)**

### Purpose

Use **GPCOND** to insert a Set Condition structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Set Condition structure element depending on the current edit mode.

During structure traversal, this structure element modifies the current condition flag as specified by the parameter. The graPHIGS API uses the condition flag when processing subsequent conditional execute structure and conditional return elements. Flags specified in the on-flag (*on*) parameter are turned on and flags specified in the off-flag (*off* parameter are turned off in the condition flag. Mathematically, the current condition flag becomes ([current-flag] or [on-flag]) and (NOT[off-flag]).

The condition flag is a 32-bit bit string. Each bit is defined as follows:




**GPCOND** is identified as GSE 1003.

> **Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Generalized Structure Elements (**GPQGSE**) inquiry subroutine to determine the GSEs supported by an open workstation. See also the workstation descriptors described in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*on* **— specified by user, fullword integer**
On-flag indicates the bits in the current condition flag to be set to one.

*off* **— specified by user, fullword integer**
Off-flag indicates the bits in the current condition flag to be set to zero.

### Error Codes

| | |
|---|---|
| 5 | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |

### Related Subroutines

| | |
|---|---|
| **GPCEXS** | Conditional Execute Structure |
| **GPCOND** | Set Condition |
| **GPCRET** | Conditional Return |
| **GPTEX2** | Test Extent 2 |
| **GPTEX3** | Test Extent 3 |

### RCP code

201331715 (X'0C001403')

## GPCRET - Conditional Return

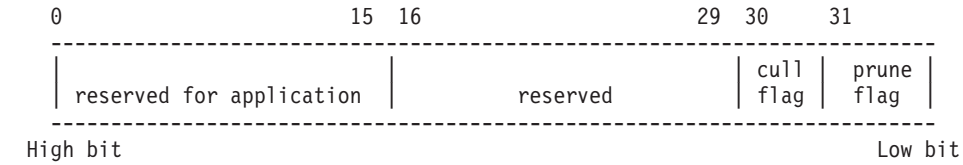
**GPCRET (*mask*, *crit*)**

### Purpose

Use **GPCRET** to insert a Conditional Return structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Conditional Return structure element depending on the current edit mode.

If the current condition flags selected by the *mask* parameter satisfy the specified criteria, then the traversal of the current structure is terminated and the traversal of the parent structure is resumed. Otherwise, no action is taken, and the element count is incremented.

The condition flag is a 32-bit bit string. Each bit is defined as follows:




**GPCRET** is identified as GSE 1005.

> **Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Generalized Structure Elements (**GPQGSE**) inquiry subroutine to determine the GSEs supported by an open workstation. See also the workstation descriptors described in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*mask* **— specified by user, fullword integer**
Mask specifying flags to be tested in the current condition flag.

*crit* **— specified by user, fullword integer**
Criteria to be satisfied when the set of flags in the specified mask is tested against the current condition flag (`1=ALL_ONES, 2=ALL_ZEROS, 3=NOT_ALL_ONES, 4=NOT_ALL_ZEROS`).

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **123** | CONDITION CRITERIA IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPCOND** | Set Condition |
| **GPCEXS** | Conditional Execute Structure |
| **GPTEX2** | Test Extent 2 |
| **GPTEX3** | Test Extent 3 |

### RCP code

201331717 (X'0C001405')

## GPEXST - Execute Structure

**GPEXST (*strid*)**

### Purpose

Use **GPEXST** to insert an Execute Structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Execute Structure element depending on the current edit mode. If the specified structure does not exist, then the graPHIGS API creates a new empty structure.

This element is one of two different execute structure-type elements that can exist in a structure. The other execute structure-type element is the Conditional Execute Structure (**GPCEXS**) element.

Traversal of the structure in which the Execute Structure element exists causes invocation of the target structure as soon as the Execute Structure element is encountered. Although the graPHIGS API does not allow recursive structure networks, no error is generated by the creation of such a network. The graPHIGS API only generates an error when an attempt is made to execute the open structure. The behavior of the graPHIGS API when traversing a recursive structure network is undefined.

### Parameters

*strid* **— specified by user, fullword integer**
Structure identifier invoked at traversal time.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **125** | ATTEMPTING TO EXECUTE THE OPEN STRUCTURE |

### Related Subroutines

**GPCEXS**
Conditional Execute Structure

### RCP code

201331713 (X'0C001401')

## GPINAD - Insert Application Data

**GPINAD (*length*, *data*)**

### Purpose

Use **GPINAD** to insert an application data structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with an Insert Application Data structure element depending on the current edit mode.

This subroutine allows the insertion of application specific data into a structure element. This data is ignored during structure traversal.

### Parameters

*length* **— specified by user, fullword integer**
Length of application data in bytes(>=0).

*data* **— specified by user, variable length data**
Application data to be placed in the element.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

### Related Subroutines

### RCP code

201328138 (X'0C00060A')

## GPINLB - Insert Label

**GPINLB (*label*)**

### Purpose

Use **GPINLB** to insert a label element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a label element depending on the current edit mode.

This structure element defines a label that the application uses to reference and modify structure elements.

### Parameters

*label* **— specified by user, fullword integer**
Identifier of label element.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

### Related Subroutines

**GPDELB**
Delete Element Between Labels

**GPDLEG**
Delete Element Group

**GPEPLG**
Generalized Set Element Pointer at Label

**RCP code**

201328137 (X'0C000609')

## GPNULL - Null Data

**GPNULL**

**Purpose**

Use **GPNULL** to insert a Null Data structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Null Data structure element depending on the current edit mode.

This structure element occupies an element position, but is ignored during structure traversal. Use **GPNULL** in an application as a minimal size placeholder structure element.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**Related Subroutines**

**GPNLER**
Nullify Element Range

**RCP code**

201328165 (X'0C000625')

## GPTEX2 - Test Extent 2

**GPTEX2 (*point1*, *point2*, *index*)**

**Purpose**

Use **GPTEX2** to insert a Test Extent 2 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Test Extent 2 structure element depending on the current edit mode.

The Test Extent 2 structure element is a short hand form of the Test Extent 3 structure element. During traversal the two endpoints of this element are expanded to 3D by supplying zero *z* values and then are treated the same as Test Extent 3. (See the Test Extent 3 [**GPTEX3**] subroutine (page GPTEX3 - Test Extent 3) for further details).

**GPTEX2** is identified as GSE 1007.

**Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Generalized Structure Elements (**GPQGSE**) inquiry subroutine to determine the GSEs supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*point1* **— specified by user, 2 short floating-point numbers (MC)**
*x*, *y* coordinates of one end point of the extent box's diagonal.

*point2* **— specified by user, 2 short floating-point numbers (MC)**
*x*, *y* coordinates of another end point of the extent box's diagonal.

*index* **— specified by user, fullword integer**
Cull size index specifying an entry in the workstation's cull size table (>=1).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**278** CULL SIZE INDEX < ONE

**Related Subroutines**

**GPCEXS**
Conditional Execute Structure

**GPCOND**
Set Condition

**GPCRET**
Conditional Return

**GPCSR**
Set Cull Size Representation

**GPQCSR**
Inquire Cull Size Representation

**GPTEX3**
Test Extent 3

**RCP code**

201331719 (X'0C001407')

## GPTEX3 - Test Extent 3

**GPTEX3 (*point1*, *point2*, *index*)**

**Purpose**

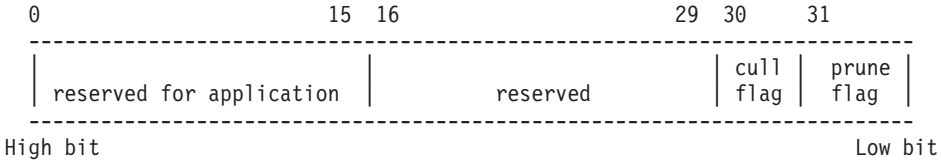
Use **GPTEX3** to insert a Test Extent 3 structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Test Extent 3 structure element depending on the current edit mode.

This structure element modifies the cull flag and prune flag within the current condition flags. These flags are used when processing subsequent conditional execute structure and conditional return elements.

During structure traversal, this element sets the prune flag (31st bit) and cull flag (30th bit) of the current condition flags in the graPHIGS traversal state list as follows:

1. Transform the specified extent box by the current modelling and viewing transformation but applying only *z* clip of the viewing transformation. The window clipping of the viewing and workstation transformation is not applied.
2. Determine the smallest box whose edges are parallel to the NPC axes and which surrounds the remaining extent box after *z* clipping.
3. If the box is completely outside the viewport of the view or completely outside of the workstation window, set the 31st bit (prune flag) to a value of one; otherwise set it to zero.
4. Transform the box by the workstation transformation and map it onto two-dimensional-DC using a parallel projection.
5. If the diagonal of the mapped rectangle is shorter than the cull size threshold specified by the cull size index, set the 30th bit (cull flag) to a value of one; otherwise set it to zero.

The condition flag is a 32-bit bit string. Each bit is defined as follows:




If the workstation does not support the specified cull size index or the specified index is outside the allowable range, then the cull size index defaults to a value of 1.

**GPTEX3** is identified as GSE 1006.

> **Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Generalized Structure Elements (**GPQGSE**) inquiry subroutine to determine the GSEs supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*point1* **— specified by user, 3 short floating-point numbers (MC)**
*x*, *y*, and *z* coordinates of one end point of the extent box's diagonal.

*point2* **— specified by user, 3 short floating-point numbers (MC)**
*x*, *y*, and *z* coordinates of the other end point of the extent box's diagonal.

*index* **— specified by user, fullword integer**
Cull size index specifying an entry of the cull size table (>=1).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**278** CULL SIZE INDEX < ONE

**Related Subroutines**

**GPCEXS**
Conditional Execute Structure

**GPCOND**
Set Condition

**GPCRET**
Conditional Return

**GPCSR**
Set Cull Size Representation

**GPQCSR**
Inquire Cull Size Representation

**GPTEX2**
Test Extent 2

**RCP code**

201331718 (X'0C001406')

## GPWDO - Workstation-Dependent Output

**GPWDO (*length*, *data*)**

### Purpose

Use **GPWDO** to insert a Workstation-Dependent Output structure element into the open structure following the element pointer or to replace the element pointed at by the element pointer with a Workstation-Dependent Output structure element depending on the current edit mode.

During structure traversal, the workstation performing the traversal outputs the data directly. The graPHIGS API does not check the validity of the data. It is the application's responsibility to ensure that the data is valid (e.g., proper length(s), identifiers, padding, etc.). When character encoding, floating-point, and/or byte ordering differences exist between the target workstation and the application environment, the application must ensure the validity of this data also. For information on how the application can ensure the validity of this data, see the Convert Data (**GPCVD**) subroutine.

Currently, this subroutine is only supported by CGM workstations. This subroutine is ignored on all other workstations. For more information on CGM data, see "CGM Workstation" in *The graPHIGS Programming Interface: Technical Reference*.

**GPWDO** is identified as GSE 1010.

> **Note:** Not all GSEs are supported on all workstations. Use the Inquire List of Generalized Structure Elements (**GPQGSE**) inquiry subroutine to determine the GSEs supported by an open workstation. See also the workstation descriptors in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*length* **— specified by user, fullword integer**
Length of output data in bytes (>=0).

*data* **— specified by user, variable length data**
Data that the application sends as output to the workstation.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**2050** INSUFFICIENT DATA LEN *n1* FOR CGM WDO

**2051** DATA LEN *n1* > 32771 FOR CGM WDO

**2052** DATA LEN *n1* <> ENCODED LEN *n2* + HDRSZ *n3* IN CGM WDO - USING ENCODED LEN

### Related Subroutines

**GPCVD**
Convert Data

**GPQNCE**
Inquire Nucleus Environment

**RCP code**

201331722 (X'0C00140A')

# Chapter 6. Structure Operations

The subroutines included in this section let your application program manipulate structures and their contents. Operations performed by these subroutines include:

- creating-deleting a structure
- creating structure hierarchies
- opening a structure for modification
- editing structure content

After opening a structure, the element pointer normally points to the last element in the structure. In order to modify a structure, the element pointer can be repositioned to a desired element. If the edit mode is set for insertion, then an element is inserted in the open structure following the element pointer. Otherwise, if the edit mode is set for replacement, then an element replaces the element at the current element pointer.

When editing structure content, the structure store containing the structures must first be selected by the application by using the Select Structure Store (**GPSSS**) subroutine.

In order to have structure data displayed on the workstation, the structure store containing the structures must first be associated to the specified workstation by the application by using the Associate Structure Store with Workstation (**GPASSW**) subroutine.

## GPASSW - Associate Structure Store with Workstation

**GPASSW (*wsid*, *ssid*)**

### Purpose

Use **GPASSW** to associate a structure store to all views on a workstation. This association gives the workstations the ability to associate structures from the specified structure store to the workstation or to views in the workstation (a necessary step to having structures displayed). If the workstation currently has a structure store associated to it, then the graPHIGS API disassociates the structure store from the workstation and associates the specified structure store.

Disassociating a structure store from a workstation is similar to invoking the Disassociate All Roots from Workstation (**GPDARW**) subroutine. In addition, **GPASSW** does not allow the graPHIGS API to associate any of the structures in the structure store to a workstation or view on the workstation until that structure store is once again associated to the workstation.

If a structure store is selected when the Open Workstation (**GPOPWS**) subroutine is invoked, then the subroutine automatically associates the selected structure store to the workstation. Use **GPASSW** at a later time to associate a different structure store to the workstation. If your application does not select a structure store when invoking the Open Workstation (**GPOPWS**) subroutine, or your application used the Create Workstation (**GPCRWS**) subroutine to open a workstation, then your application needs to explicitly associate a structure store to the workstation before the graPHIGS API can add the structures to views and then display them.

A structure store has a limit on the number of workstations to which it can associate. This limit applies to all application processes attached to the structure store resource. To determine the limit, use the Inquire List of Available Workstation Types on Nucleus (**GPQWTN**) subroutine. The graPHIGS API issues error message 226 if this limit is exceeded.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*ssid* — **specified by user, fullword integer**
Structure store identifier.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **215** | SPECIFIED RESOURCES DO NOT EXIST ON THE SAME NUCLEUS |
| **222** | SPECIFIED STRUCTURE STORE DOES NOT EXIST |
| **226** | MAXIMUM NUMBER OF SIMULTANEOUS ASSOCIATED WORKSTATIONS EXCEEDED |

### Related Subroutines

| | |
|---|---|
| **GPARV** | Associate Root with View |
| **GPARW** | Associate Root with Workstation |
| **GPDARW** | Disassociate All Roots from Workstation |
| **GPOPWS** | Open Workstation |
| **GPSSS** | Select Structure Store |

### RCP code

201333506 (X'0C001B02')

## GPCCM - Set Convexity Checking Mode

**GPCCM (*mode*)**

### Purpose

Use **GPCCM** to set the current convexity checking mode in the graPHIGS API state list.

Your application can set this mode to `1=OFF` or `2=ON`. If the convexity checking mode is set to `2=ON`, then the graPHIGS API determines if subsequent primitives with indeterminate convexity are convex for the following primitives:

| | |
|---|---|
| **GPPG2** | Polygon 2 |
| **GPPG3** | Polygon 3 |
| **GPPGD2** | Polygon 2 With Data |
| **GPPGD3** | Polygon 3 With Data |

The graPHIGS API then stores the convexity determination as part of the structure element, which can enhance traversal performance.

The default convexity checking mode is `1=OFF`.

### Parameters

*mode* — **specified by user, fullword integer**
Convexity checking mode (`1=OFF`, `2=ON`).

### Error Codes

| | |
|---|---|
| **98** | CONVEXITY CHECKING MODE IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPPG2** | Polygon 2 |
| **GPPG3** | Polygon 3 |
| **GPPGD2** | Polygon 2 With Data |
| **GPPGD3** | Polygon 3 With Data |

### RCP code

201330966 (X'0C001116')

## GPCEDT - Conditional Editing

**GPCEDT (*flag*)**

### Purpose

Use **GPCEDT** to begin and end conditional structure editing in the current open structure.

Once conditional editing begins, all affected structure operation subroutine calls (i.e., all structure store subroutine calls except for inquiry subroutines and those subroutines which do not require an open structure) are performed as usual until either the conditional editing is ended or your application closes the structure or until an error occurs (except for errors caused by inserting or replacing structure elements). When an error occurs while conditional editing is active, the graPHIGS API ignores all subsequent editing operations until either the conditional editing is ended or your application closes the structure store.

Structure operations that temporarily close an open structure, such as the Delete Structure (**GPDLST**) subroutine, do not affect the conditional editing mode.

### Parameters

*flag* **— specified by user, fullword integer**
Conditional edit flag (`1=START, 2=END`).

### Error Codes

| | |
|---|---|
| **4** | FUNCTION REQUIRES STATE STOP |
| **137** | VALUE OF CONDITIONAL EDIT FLAG IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPCLST** | Close Structure |
| **GPCPER** | Copy Element Range |
| **GPCPST** | Copy Structure |
| **GPDELB** | Delete Element Between Labels |
| **GPDLE** | Delete Element |
| **GPDLER** | Delete Element Range |
| **GPEP** | Set Element Pointer at Label |
| **GPEPCD** | Locate Element Pointer at Element Code |

| | |
|---|---|
| **GPEPLG** | Generalized Set Element Pointer at Label |
| **GPEPPG** | Generalized Set Element Pointer at Pick Identifier |
| **GPMVER** | Move Element Range |
| **GPOEP** | Offset Element Pointer |
| **GPOPST** | Open Structure |
| **GPSSS** | Select Structure Store |
| **All Structure Elements** | Chapter 3, Attribute Structure Elements<br>Chapter 5, Structure Operations<br>Chapter 15, Miscellaneous Subroutines |

### RCP code

201331462 (X'0C001306')

## GPCLST - Close Structure

**GPCLST**

### Purpose

Use **GPCLST** to close a structure in the currently selected structure store. The current structure state is set to Structure Closed (STCL).

If your application started conditional editing, then the graPHIGS API stops conditional editing.

Once closed, your application cannot insert or replace structure elements in the structure until the structure is opened. Also, the structure store containing the structure being closed becomes available to be updated by other application processes attached to the structure store.

### Parameters

None

### Error Codes

| | |
|---|---|
| **4** | FUNCTION REQUIRES STATE STOP |

### Related Subroutines

| | |
|---|---|
| **GPCEDT** | Conditional Editing |
| **GPOPST** | Open Structure |

### RCP code

201331458 (X'0C001302')

## GPCPER - Copy Element Range

**GPCPER (*strid*, *elem1*, *elem2*)**

### Purpose

Use **GPCPER** to copy a range of elements from a specified structure and to insert the range of elements into the open structure following the current element pointer. The element pointer moves to the last element copied into the open structure.

- If the value for *elem1* is greater than the value for *elem2*, then the values are swapped. (The element range always starts at the lowest value and ends at the highest value.)
- If both values point beyond the last element of the specified structure or if both values are less than one, then the graPHIGS API does not perform the copy.
- If one of the values is less than one, then the value defaults to a value of one.
- If one of the values is greater than the number of elements in the specified structure, then the element number of the last element in that structure is used instead.

The value of the edit mode set by **GPEDMO** does not affect the functionality of this subroutine.

If the elements to be copied contain an execute structure-type element that references the open structure, then the graPHIGS API issues an error message and does not perform the copy.

### Parameters

*strid* **— specified by user, fullword integer**
Identifier of the structure which has elements that are copied.

*elem1* **— specified by user, fullword integer**
Element number of the first element to copy.

*elem2* **— specified by user, fullword integer**
Element number of the last element to copy.

### Error Codes

| | |
|---|---|
| **4** | FUNCTION REQUIRES STATE STOP |
| **125** | ATTEMPTING TO EXECUTE THE OPEN STRUCTURE |

### Related Subroutines

| | |
|---|---|
| **GPCEXS** | Conditional Execute Structure |
| **GPCPST** | Copy Structure |
| **GPEXST** | Execute Structure |

### RCP code

201338626 (X'0C002F02')

## GPCPST - Copy Structure

**GPCPST (*strid*)**

### Purpose

Use **GPCPST** to copy the elements from the specified structure and to insert the elements into the open structure following the current element pointer. The element pointer moves to the last element copied into the open structure.

The value of the edit mode set by **GPEDMO** does not affect the functionality of this subroutine.

If the structure you copy references the open structure (by the use of an execute structure-type element), then the graPHIGS API issues an error message and does not perform the copy.

### Parameters

*strid* **— specified by user, fullword integer**
Identifier of the structure whose contents are copied.

### Error Codes

| | |
|---|---|
| **4** | FUNCTION REQUIRES STATE STOP |
| **125** | ATTEMPTING TO EXECUTE THE OPEN STRUCTURE |

### Related Subroutines

| | |
|---|---|
| **GPCEXS** | Conditional Execute Structure |
| **GPEXST** | Execute Structure |

### RCP code

201338625 (X'0C002F01')

## GPCSI - Change Structure Identifier

**GPCSI (*ostrid*, *rstrid*)**

### Purpose

Use **GPCSI** to change the identifier of a structure (called the *original structure*) to a specified structure identifier (called the *resulting structure*). This subroutine does not affect execute structure-type elements that reference the original structure.

If the identifier of the original structure is the same as the identifier of the resulting structure and the structure exists, then no action occurs. If the structure does not exist, however, then the graPHIGS API creates an empty structure with the identifier of the resulting structure.

If the original structure does not exist, then the graPHIGS API empties the resulting structure. If the original structure does exist, then the contents of the original structure replace the contents of the resulting structure and the graPHIGS API empties the original structure. If the original structure references the resulting structure, then the graPHIGS API issues an error message and does not perform the change.

At the completion of this subroutine call, the system deletes the original structure *unless* the original structure is the open structure, is referenced by any other structure or is associated to a view or workstation.

If the *original* structure is the open structure, then the graPHIGS API sets the current element pointer to zero. If the *resulting* structure is the open structure, then the graPHIGS API sets the current element pointer to point to the last element in the structure.

### Parameters

*ostrid* **— specified by user, fullword integer**
Original structure identifier.

*rstrid* **— specified by user, fullword integer**
Resulting structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |
| **129** | ATTEMPTING TO HAVE THE RESULTING STRUCTURE EXECUTE ITSELF |

### Related Subroutines

| | |
|---|---|
| **GPARV** | Associate Root with View |
| **GPARW** | Associate Root with Workstation |
| **GPCEXS** | Conditional Execute Structure |
| **GPEST** | Empty Structure |
| **GPEXST** | Execute Structure |

### RCP code

201347073 (X'0C005001')

## GPCSIR - Change Structure Identifier and References

**GPCSIR (*ostrid*, *rstrid*)**

### Purpose

Use **GPCSIR** to change all execute structure-type elements which reference a structure (called the *original structure*) with elements which reference a specified structure (called the *resulting structure*) and then change the identifier of the original structure with that of the resulting structure. The effect of this subroutine is as though the application called the Change Structure References (**GPCSRS**) subroutine followed by a call to the Change Structure Identifier (**GPCSI**) subroutine.

If the original structure references the resulting structure, then the graPHIGS API issues an error and does not perform the change.

### Parameters

*ostrid* **— specified by user, fullword integer**
Original structure identifier.

*rstrid* **— specified by user, fullword integer**
Resulting structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |
| **129** | ATTEMPTING TO HAVE THE RESULTING STRUCTURE EXECUTE ITSELF |

### Related Subroutines

| | |
|---|---|
| **GPCSI** | Change Structure Identifier |
| **GPCSRS** | Change Structure References |

### RCP code

201347075 (X'0C005003')

## GPCSRS - Change Structure References

**GPCSRS (*ostrid*, *rstrid*)**

### Purpose

Use **GPCSRS** to change all execute structure-type elements which reference a structure (called the *original structure*) with elements which reference a specified structure (called the *resulting structure*). This subroutine does not affect any references to the resulting structure that existed before the call.

If the identifier of the original structure and the identifier of the resulting structure are identical, then no action occurs. If the resulting structure references the original structure, then the graPHIGS API issues an error message and does not perform the change.

If there are references to the original structure and the resulting structure does not exist, then the graPHIGS API creates an empty structure with the identifier of the resulting structure. If the original structure does not exist or if there are no references to the original structure, then no action occurs.

If the original structure and resulting structure are both associated to a workstation (or a view), then the original structure is disassociated and the resulting structure remains associated to the workstation (or the view). If the original structure is associated to a workstation (or view) but the resulting structure is not associated, then the original structure is disassociated and the resulting structure is associated to the workstation, or in the case of a view, the original structure is disassociated and the resulting structure is associated to the view with the same priority that the original structure had.

### Parameters

*ostrid* **— specified by user, fullword integer**
Original structure identifier.

*rstrid* **— specified by user, fullword integer**
Resulting structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |
| **129** | ATTEMPTING TO HAVE THE RESULTING STRUCTURE EXECUTE ITSELF |

### Related Subroutines

| | |
|---|---|
| **GPARV** | Associate Root with View |
| **GPARW** | Associate Root with Workstation |
| **GPCEXS** | Conditional Execute Structure |
| **GPDRV** | Disassociate Root from View |
| **GPDRW** | Disassociate Root from Workstation |
| **GPEXST** | Execute Structure |

### RCP code

201347074 (X'0C005002')

## GPDAST - Delete All Structures

**GPDAST**

### Purpose

Use **GPDAST** to delete all existing structures from the currently selected structure store.

This subroutine is equivalent to invoking the Delete Structure(**GPDLST**) subroutine for each structure in the selected structure store.

### Parameters

None

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |

### Related Subroutines

| | |
|---|---|
| **GPDLST** | Delete Structure |

### RCP code

201332993 (X'0C001901')

## GPDCM - Set Direct Color Model

**GPDCM (*model*)**

### Purpose

Use**GPDCM** to set the current direct color model in the graPHIGS API state list. This color model is used for interpreting color values specified in all subsequent structure element subroutines which have direct color parameters.

### Parameters

*model* **— specified by user, fullword integer**
Color model(`1=RGB, 2=HSV, 3=CMY, 4=CIELUV`).

### Error Codes

| | |
|---|---|
| **318** | COLOR MODEL INVALID |

### Related Subroutines

| | |
|---|---|
| **GPQDCM** | Inquire Direct Color Model |

### RCP code

201343502(X'0C00420E')

## GPDELB - Delete Element Between Labels

**GPDELB (*label1*, *label2*)**

### Purpose

Use**GPDELB** to delete all elements from the open structure between, but not including, the two specified labels. After deletion, the element pointer moves to the element specified by the label identifier 1(*label1*).

Starting from the current position of the element pointer, the system searches for label identifier 1. When the end of the structure is reached, the search continues at element number 1.Beginning with the original position of the element pointer, the system then searches for label identifier 2.When the end of the structure is reached, the system continues the search at element number 1. If the system cannot locate either label, then the graPHIGS API issues an error.

### Parameters

*label1* **— specified by user, fullword integer**
Label identifier 1.

*label2* **— specified by user, fullword integer**
Label identifier 2.

### Error Codes

| | |
|---|---|
| **4** | FUNCTION REQUIRES STATE STOP |
| **130** | LABEL IDENTIFIER CANNOT BE FOUND IN THE OPEN STRUCTURE |

### Related Subroutines

| | |
|---|---|
| **GPINLB** | Insert Label |

### RCP code

201338370 (X'0C002E02')

## GPDLE - Delete Element

**GPDLE**

### Purpose

Use **GPDLE** to delete the element indicated by the element pointer from the open structure. The element pointer moves to the element immediately preceding the deleted element.

### Parameters

None.

### Error Codes

| | |
|---|---|
| **4** | FUNCTION REQUIRES STATE STOP |

### Related Subroutines

None

### RCP code

201332481 (X'0C001701')

## GPDLEG - Delete Element Group

**GPDLEG (*label1*, *label2*, *option*)**

### Purpose

Use **GPDLEG** to delete all elements from the open structure between the two specified labels. There are two methods that you can use to search for the labels: the graPHIGS API method and the ISO PHIGS method. Specify the method using the search and deletion option field.

**graPHIGS API method**: If the *option* parameter is `1=START_AT_CURRENT_ELEMENT_NEITHER_DELETED`, `2=START_AT_CURRENT_ELEMENT_BOTH_DELETED`, `3=START_AT_CURRENT_ELEMENT_FIRST_DELETED`, `4=START_AT_CURRENT_ELEMENT_SECOND_DELETED`, then starting from the current position of the element pointer, the graPHIGS API searches for label identifier 1. The system then searches for label identifier 2, starting from the position of the label now found.

**ISO PHIGS method**: If the *option* parameter is `5=START_AT_NEXT_ELEMENT_NEITHER_DELETED`, `6=START_AT_NEXT_ELEMENT_BOTH_DELETED`, `7=START_AT_NEXT_ELEMENT_FIRST_DELETED`, `8=START_AT_NEXT_ELEMENT_SECOND_DELETED`, then starting from the element immediately *after* the current position of the element pointer, the graPHIGS API searches for label identifier 1. The system then searches for label identifier 2, starting from the label element immediately *after* the label now found.

Regardless of the method used to find the labels, if the system reaches the end of the structure before both labels are found, the graPHIGS API issues an error and the current element pointer remains at the original position before this subroutine was called.

For the deletion process, the application can control whether or not the specified label elements are also deleted. When the *option* parameter is `1=START_AT_CURRENT_ELEMENT_NEITHER_DELETED` (when using the graPHIGS API method) or `5=START_AT_NEXT_ELEMENT_NEITHER_DELETED` (when using the ISO PHIGS method) neither of the two labels are deleted.

When the *option* parameter is `2=START_AT_CURRENT_ELEMENT_BOTH_DELETED` (graPHIGS API) or `6=START_AT_NEXT_ELEMENT_BOTH_DELETED` (ISO PHIGS) then both of the labels are deleted.

When *option* is `3=START_AT_CURRENT_ELEMENT_FIRST_DELETED` (graPHIGS API) or `7=START_AT_NEXT_ELEMENT_FIRST_DELETED` (ISO PHIGS), only the first label is deleted.

When *option* `4=START_AT_CURRENT_ELEMENT_SECOND_DELETED` (graPHIGS API) or `8=START_AT_NEXT_ELEMENT_SECOND_DELETED` (ISO PHIGS), only the second label is deleted.

After deletion, the element pointer is positioned at the element immediately preceding the deleted elements.

### Parameters

*label1* — **specified by user, fullword integer**
Label identifier 1.

*label2* — **specified by user, fullword integer**
Label identifier 2.

*option* — **specified by user, fullword integer**
Label deletion option (1=START_AT_CURRENT_ELEMENT_NEITHER_DELETED, 2=START_AT_CURRENT_ELEMENT_BOTH_DELETED, 3=START_AT_CURRENT_ELEMENT_FIRST_DELETED, 4=START_AT_CURRENT_ELEMENT_SECOND_DELETED, 5=START_AT_NEXT_ELEMENT_NEITHER_DELETED, 6=START_AT_NEXT_ELEMENT_BOTH_DELETED, 7=START_AT_NEXT_ELEMENT_FIRST_DELETED, 8=START_AT_NEXT_ELEMENT_SECOND_DELETED).

### Error Codes

| | |
|---|---|
| **4** | FUNCTION REQUIRES STATE STOP |
| **130** | LABEL IDENTIFIER CANNOT BE FOUND IN THE OPEN STRUCTURE |
| **133** | LABEL DELETE OPTION IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPINLB** | Insert Label |

### RCP code

201338371 (X'0C002E03')

## GPDLER - Delete Element Range

**GPDLER (*elem1*, *elem2*)**

### Purpose

Use **GPDLER** to delete all structure elements between and including the elements indicated by the specified element numbers.

After deletion, the element pointer moves to the element immediately preceding any deleted elements. If both values point beyond the last element of the structure or both values are less than zero, then the graPHIGS API does not delete any elements and the element pointer remains the same. If one of the values is less than zero, then the element pointer defaults to a value of zero. If one of the values is greater than the number of elements in the open structure, then the graPHIGS API uses the element number of the last element in the open structure.

### Parameters

*elem1* — **specified by user, fullword integer**
First element number.

*elem2* — **specified by user, fullword integer**
Second element number.

### Error Codes

| | |
|---|---|
| **4** | FUNCTION REQUIRES STATE STOP |

### Related Subroutines

None

### RCP code

201338369 (X'0C002E01')

## GPDLNC - Delete Structure Network Conditionally

**GPDLNC (*strid*)**

### Purpose

Use **GPDLNC** to delete the specified structure (considered to be the root of the network) and to conditionally delete all structures in the network (which consists of those structures referenced, either directly or indirectly, by the root structure). Those structures in the network that are referenced, either directly or indirectly, by structures outside the network (excluding those structures whose reference is through the root structure) are not deleted.

All execute structure-type elements referencing any of the deleted structures are also deleted.

### Parameters

*strid* **— specified by user, fullword integer**
Structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |

### Related Subroutines

| | |
|---|---|
| **GPCEXS** | Conditional Execute Structure |
| **GPDLST** | Delete Structure |
| **GPEXST** | Execute Structure |

### RCP code

201331970(X'0C001502')

## GPDLNT - Delete Structure Network

**GPDLNT (*strid*)**

### Purpose

Use **GPDLNT** to delete the specified structure and all structures referenced, either directly or indirectly, by the given structure.

All execute structure-type elements referencing any of the deleted structures are also deleted.

### Parameters

*strid* — **specified by user, fullword integer**
Structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |

### Related Subroutines

| | |
|---|---|
| **GPCEXS** | Conditional Execute Structure |
| **GPDLST** | Delete Structure |
| **GPEXST** | Execute Structure |

### RCP code

201331969(X'0C001501')

## GPDLST - Delete Structure

**GPDLST(*strid*)**

### Purpose

Use **GPDLST** to delete the specified structure, its identifier, and its contents. This subroutine removes all references to the deleted structure (execute structure-type elements)from all existing structures. The deleted structure is also disassociated from all workstations.

If it is necessary to empty the contents of a specified structure and at the same time maintain its association with all workstations, use the Empty Structure (**GPEST**) subroutine. References to this now emptied structure remain intact.

### Parameters

*strid* — **specified by user, fullword integer**
Structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |

### Related Subroutines

| | |
|---|---|
| **GPCEXS** | Conditional Execute Structure |
| **GPDRW** | Disassociate Root from Workstation |
| **GPEST** | Empty Structure |
| **GPEXST** | Execute Structure |

### RCP code

201332737(X'0C001801')

## GPEDMO - Set Edit Mode

**GPEDMO (*mode*)**

### Purpose

Use **GPEDMO** to set the current edit mode entry of the graPHIGS API state list. If you do not use this subroutine, then the edit mode default to `1=INSERT_MODE`.

When moving an element into an open structure, the edit mode determines how the element is placed into the structure:

1. When in `2=REPLACE_MODE`, the graPHIGS API deletes the element at the element pointer and the incoming element takes its place. The element pointer does not move (so if you place another element into the structure, then it replaces the last element that was just placed into the structure). If the element pointer is set to element 0, then the graPHIGS API inserts the incoming element into the structure and sets the element element pointer to element 1.
2. When in `1=INSERT_MODE`, the graPHIGS API inserts the incoming element into the open structure after the element at the element pointer. The element pointer is then incremented by 1 to point to the new element.

### Parameters

*mode* **— specified by user, fullword integer**
Edit mode (`1=INSERT_MODE, 2=REPLACE_MODE`).

### Error Codes

**121** EDIT MODE IS INVALID

### Related Subroutines

None

### RCP code

201331460 (X'0C001304')

## GPEP - Set Element Pointer

**GPEP (*elem*)**

### Purpose

Use **GPEP** to set the element pointer to the specified element number.

If the value is less than zero, then the graPHIGS API sets the pointer to zero. If the value is greater than the number of elements in the open structure, then the graPHIGS API sets the pointer to the last element.

### Parameters

*elem* **— specified by user, fullword integer**
Element pointer value.

**Error Codes**

**4** FUNCTION REQUIRES STATE STOP

**Related Subroutines**

**GPQEP**
Inquire Element Pointer

**RCP code**

201332225 (X'0C001601')

## GPEPCD - Locate Element Pointer at Element Code

**GPEPCD (*code*)**

**Purpose**

Use **GPEPCD** to set the element pointer to the next occurrence of the structure element with the specified code (i.e., the structure element identifier). If no element with the specified element code is found between the current element pointer and the end of the open structure, then the graPHIGS API issues an error and the element pointer remains the same.

Valid element codes are found in the *The graPHIGS Programming Interface: Technical Reference*.

**Note:** Element codes used in Version 1 are not supported by this subroutine.

**Parameters**

*code* **— specified by user, fullword integer**
Element code. For a list of element codes used by the graPHIGS API, see *The graPHIGS Programming Interface: Technical Reference*.

**Error Codes**

**4** FUNCTION REQUIRES STATE STOP

**132** ELEMENT CODE DOES NOT EXIST BEFORE END OF STRUCTURE

**Related Subroutines**

**GPQE** Inquire Element Content

**GPQEHD**
Inquire List of Element Headers

**RCP code**

201332229 (X'0C001605')

## GPEPLG - Generalized Set Element Pointer at Label

**GPEPLG (*label*, *flag*)**

### Purpose

Use **GPEPLG** to set the element pointer to the specified pick identifier element within the open structure.

Starting at the position following the current element pointer, the graPHIGS API searches for the first occurrence of the specified label. If the end of the structure is reached, then the result depends on the value of the label search flag (*flag*) parameter:

- If the value of the flag is `2=WRAP`, then the search continues starting at element 1 until the current element pointer is encountered. If the specified *label* is not found, the graPHIGS API issues an error and no action is performed.
- If the value of the flag is `1=NOWRAP`, then the graPHIGS API issues an error and no action is performed.

### Parameters

*label* **— specified by user, fullword integer**
Label identifier.

*flag* **— specified by user, fullword integer**
Label search flag (`1=NOWRAP`, `2=WRAP`).

### Error Codes

**4** FUNCTION REQUIRES STATE STOP

**130** LABEL IDENTIFIER CANNOT BE FOUND IN THE OPEN STRUCTURE

**136** VALUE OF SEARCH FLAG IS INVALID

### Related Subroutines

**GPINLB**
Insert Label

### RCP code

201332230 (X'0C001606')

## GPEPPG - Generalized Set Element Pointer at Pick Identifier

**GPEPPG (*pickid*, *flag*)**

### Purpose

Use **GPEPPG** to set the element pointer to the specified pick identifier element within the open structure.

Starting at the position following the current element pointer, the graPHIGS API searches for the first occurrence of the specified label. If the end of the structure is reached, then the result depends on the value of the pick identifier search flag (*flag*) parameter:

- If the value of the flag is `2=WRAP`, then the search continues starting at element 1 until the current element pointer is encountered. If the specified *pickid* is not found, then the graPHIGS API issues an error and no action is performed.
- If the value of the flag is `1=NOWRAP`, then the graPHIGS API issues an error and no action is performed.

### Parameters

*pickid* **— specified by user, fullword integer**
Pick identifier.

*flag* — **specified by user, fullword integer**
Pick identifier search flag (1=NOWRAP, 2=WRAP)

**Error Codes**

**4** FUNCTION REQUIRES STATE STOP

**136** VALUE OF SEARCH FLAG IS INVALID

**566** PICK IDENTIFIER DOES NOT EXIST IN THE OPEN STRUCTURE

**Related Subroutines**

**GPPKID**
Set Pick Identifier

**RCP code**

201332231 (X'0C001607')

## GPEST - Empty Structure

**GPEST (*strid*)**

**Purpose**

Use **GPEST** to empty the contents of the specified structure.

References to this now empty structure remain intact. If the specified structure is the open structure, then the graPHIGS API sets the element pointer to zero. If the specified structure does not exist, then the graPHIGS API creates a new empty structure.

**Parameters**

*strid* — **specified by user, fullword integer**
Identifier of the structure whose contents are emptied.

**Error Codes**

**12** FUNCTION REQUIRES STATE SSSL

**Related Subroutines**

None

**RCP code**

201333249 (X'0C001A01')

## GPMVER - Move Element Range

**GPMVER (*elem1*, *elem2*)**

**Purpose**

Use **GPMVER** to move a range of elements from an open structure back into the same open structure but inserted following the current element pointer. The element pointer moves to the last element moved into the open structure.

- If the value for *elem1* is greater than the value for *elem2*, then the values are swapped. (The element range always starts at the lowest value and ends at the highest value.)
- If both values point beyond the last element of the specified structure or if both values are less than one, then the graPHIGS API does not perform the move.
- If one of the values is less than one, then the element number defaults to a value of one.
- If one of the values is greater than the number of elements in the specified structure, then the element number of the last element in that structure is used instead.
- If the element range includes the element pointed to by the current element pointer, then the graPHIGS API does not perform the move.

The value of the edit mode set by **GPEDMO** does not affect the functionality of this subroutine.

**Parameters**

*elem1* **— specified by user, fullword integer**
Element number of the first element to move.

*elem2* **— specified by user, fullword integer**
Element number of the last element to move.

**Error Codes**

**4** FUNCTION REQUIRES STATE STOP

**Related Subroutines**

None

**RCP code**

201338627 (X'0C002F03')

## GPNLER - Nullify Element Range

**GPNLER** (*re1*, *re2*)

**Purpose**

Use **GPNLER** to replace all elements between the specified structure elements in the open structure with Null Data structure elements. This subroutine requires that the structure state is Structure Open (STOP).

Two element search parameters, *re1* and *re2*, specify the range of structure elements to be processed. Each of these parameters is an array of three fullword integers. The first integer specifies the method to use in the element search and the second specifies the target value of the element search. The third integer specifies an option value.

If both of the specified elements point beyond the last element in the structure, then no elements are replaced and the element pointer remains unchanged.

If one of the specified elements points beyond the last element in the structure, then the graPHIGS API uses the last last element in the open structure instead.

If both of the element numbers are less than one, then the graPHIGS API does not replace any elements and the element pointer remains unchanged.

If one of the element numbers is less than one, then the element number value defaults to a value of one.

Elements between the lower element number and the higher element number are replaced with Null Data (**GPNULL**) structure elements. If the element range consists of only one element and the options on parameters *re1* and *re2* are conflicting, then the graPHIGS API applies to the `2=EXCLUDE` option value.

After replacement of elements, the element pointer moves to the last element replaced.

**Parameters**

*re1* **— specified by user, 3 fullword integers**
1st range element search parameter in the following format:

```
  -----------------
0 | search method | fullword integer
  |---------------|
4 | target value  | fullword integer
  |---------------|
8 | option value  | fullword integer
  -----------------

 Search method supported:
     Method 1 = ELEMNUM
 Target value:
     For method 1, the target value is a structure element
     number.
 Option values supported:
     1 = INCLUDE
         Nullifies the resulting structure element derived
         from the range element search parameter.
     2 = EXCLUDE
         Does not nullify the resulting structure element
         derived from the range element search parameter.
```

*re2* **— specified by user, 3 fullword integers**
2nd range element search parameter. Format the same as *re1* above.

**Error Codes**

**4** FUNCTION REQUIRES STATE STOP

**139** SEARCH METHOD IS INVALID

**143** OPTION VALUE IS INVALID

**607** NUCLEUS IS DOWN LEVEL. VERSION @A1, RELEASE @A2.@A3 IS REQUIRED

**Related Subroutines**

**GPCEDT**
Conditional Editing

**GPNULL**
Null Data

**RCP code**

201338372 (X'0C002E04')

## GPOEP - Offset Element Pointer

**GPOEP (*offset*)**

### Purpose

Use **GPOEP** to move the element pointer to a new element relative to the current element pointer.

This subroutine adds the pointer offset value to the element pointer. The value can either be positive, which moves the element pointer forward, or negative, which moves the element pointer backward. If the resultant value is less than zero, then the element pointer defaults to a value of zero. If the resultant value is greater than the number of elements in the open structure, then the graPHIGS API sets the pointer to point to the last element.

### Parameters

*offset* — **specified by user, fullword integer**
Element pointer offset value:

**Positive offset value**
Moves element pointer forward.

**Negative offset value**
Moves element pointer backward.

### Error Codes

**4** FUNCTION REQUIRES STATE STOP

### Related Subroutines

**GPQEP**
Inquire Element Pointer

### RCP code

201332226 (X'0C001602')

## GPOPST - Open Structure

**GPOPST (*strid*)**

### Purpose

Use **GPOPST** to open a structure. The structure state is set to Structure Open (STOP).

When the application opens a structure, the graPHIGS API sets the elment pointer to the last element of the currently specified structure. If the specified structure does not exist in the currently selected structure store, then the graPHIGS API creates an empty structure and sets the element pointer to zero.

The currently selected structure store is locked to structure manipulation requests generated by other application processes, and the graPHIGS API considers the application process that opened the structure as the owning application process. The graPHIGS API postpones the requests from the other application processes until the open structure is closed (at which time the owning application process relinquishes its

lock on the structure store and the structure store becomes available for structure requests by other application processes). This ownership of the structure store ensures that other application processes cannot interfere with the structure requests of the owning application process until it closes the structure.

**Parameters**

*strid* — **specified by user, fullword integer**
Structure identifier.

**Error Codes**

**11** FUNCTION REQUIRES STATE STCL

**Related Subroutines**

**GPCLST**
Close Structure

**RCP code**

201331457 (X'0C001301')

## GPSSS - Select Structure Store

**GPSSS (*ssid*)**

**Purpose**

Use **GPSSS** to select a structure store. A structure store must be attached using the Create Structure Store (**GPCRSS**) subroutine or the Attach Resource (**GPATR**) subroutine before your application can call this structure element. The selected structure store becomes the target of all subsequent structure manipulation/editing subroutines until your application issues another call to **GPSSS**.

If a structure is currently open when a structure store is selected, then the graPHIGS API closes the structure. The specified structure store becomes the selected structure store and the structure state is set to Structure Closed (STCL).

**Note:** Creating a structure store or attaching a structure store does *not* automatically select the structure store for editing. If you did not suppress default connection processing during Open graPHIGS (**GPOPPH**), then this subroutine is automatically invoked as part of the **GPOPPH** processing. See **GPOPPH** for more information.

**Parameters**

*ssid* — **specified by user, fullword integer**
Structure store identifier.

**Error Codes**

**13** FUNCTION REQUIRES STATE SSOP

**222** SPECIFIED STRUCTURE STORE DOES NOT EXIST

**Related Subroutines**

**GPASSW**
Associate Structure Store with Workstation

**GPATR**
Attach Resource

**GPCRSS**
Create Structure Store

**GPOPPH**
Open graPHIGS

**RCP code**

201331459 (X'0C001303')

## GPSSTH - Set Structure Store Threshold

**GPSSTH (*ssid*, *threshold*)**

### Purpose

Use **GPSSTH** to specify a storage threshold for a structure store. This threshold represents the size (in bytes) of the structure store.

When the amount of storage allocated to the specific structure store exceeds the specified storage threshold, the graPHIGS API generates a Threshold Exceeded event and sends it to all application processes attached to the structure store. If the size of the structure store already exceeds the threshold value at the time the threshold value is set, then the graPHIGS API generates the event immediately. Once the event is generated, no further events are generated until your application calls **GPSSTH** again (by any application process attached to the structure store).

Each structure store has only one threshold value that your application can set. If your application sets a structure store threshold value, then that value overrides any other threshold that may have been set. For information on Threshold Events, see *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*ssid* **— specified by user, fullword integer**
Structure store identifier.

*threshold* **— specified by user, fullword integer**
Structure store threshold value in bytes (>=0).

### Error Codes

**222** SPECIFIED STRUCTURE STORE DOES NOT EXIST

**225** STRUCTURE STORE THRESHOLD SIZE < ZERO

### Related Subroutines

**GPAWEV**
Await Event

**GPEHND**
Define Error Handling Subroutine

**GPEVHN**
Define Event Handling Subroutine

**GPQNCS**
Inquire Available Nucleus Storage Size

**RCP code**

201331461 (X'0C001305')

# GPTAST - Transfer All Structures

**GPTAST (*ssid*, *flag*)**

### Purpose

Use **GPTAST** to copy all structures from a specified structure store to the currently selected structure store. Both the source and target structure stores can reside on the same or different nuclei. If the specified structure store is the currently selected structure store, then no action is taken.

If any structure within the specified structure store already exists in the currently selected structure store, then the graPHIGS API resolves the conflict according to the conflict resolution flag. If there is a conflict and the resolution flag is set to `1=MAINTAIN`, then the graPHIGS API only transfers structures that do *not* conflict and there is no change to any existing structure. When the conflict resolution flag is set to `2=ABANDON`, the graPHIGS API does not transfer any structures and issues an error. When the flag is set to `3=UPDATE`, the graPHIGS API transfers every structure. The conflicting structure replaces the new one without changing any associations to a workstation or view.

> **Note:** The graPHIGS API empties the structure in the target structure store *before* it copies the source structure. If an error occurs (such as the copying of a structure cannot be completed), then the original structure cannot be reconstructed.

When you replace a structure that is an open structure, the structure remains open and is emptied, but the graPHIGS API does not remove references to the structure. The result is as though you had issued the following:

1. **GPCLST** - Close Structure
2. **GPEST** - Empty Structure
3. **GPTST** - Transfer Structures
4. **GPOPST** - Open Structure

If your application transfers an execute structure-type element that refers to a non-existing structure, then the graPHIGS API creates an empty structure.

### Parameters

*ssid* **— specified by user, fullword integer**
Structure store identifier.

*flag* **— specified by user, fullword integer**
Conflict resolution flag (`1=MAINTAIN, 2=ABANDON, 3=UPDATE`).

### Error Codes

**12** FUNCTION REQUIRES STATE SSSL

**127** CONFLICT RESOLUTION FLAG IS INVALID

**128** STRUCTURE CONFLICT OCCURS WHEN RESOLUTION FLAG IS ABANDON

**222** SPECIFIED STRUCTURE STORE DOES NOT EXIST

**594** DATA EXCEEDS CONNECTION BUFFER SIZE

**614** UNKNOWN ELEMENT FOUND IN STRUCTURE *n1*

**Related Subroutines**

**GPTST**
Transfer Structures

**RCP code**

201348611 (X'0C005603')

# GPTST - Transfer Structures

**GPTST (*ssid*, *flag*, *number*, *lstrid*)**

**Purpose**

Use **GPTST** to copy one or more structures from a specified structure store to the currently selected structure store. Both the source and target structure stores can reside on the same or different nuclei. If the specified structure store is the currently selected structure store, then no action is taken.

If any of the specified structures do not exist in the specified structure store and the specified structure identifier does not exist in the current structure store, then the graPHIGS API issues a warning and creates an empty structure. If any of the specified structures do not exist in the specified structure store and the specified structure identifier does exist in the current structure store and the conflict resolution flag is set to `3=UPDATE`, then the graPHIGS API issues a warning and empties the structure.

If one of the specified structures already exists in the currently selected structure store, then the graPHIGS API resolves the conflict according to the conflict resolution flag. If there is a conflict and the resolution flag is set to `1=MAINTAIN`, the graPHIGS API only transfers structures that do *not* conflict and there is no change to any existing structure. When the conflict resolution flag is set to `2=ABANDON`, the graPHIGS API does not transfer any structures and issues an error. When the flag is set to `3=UPDATE`, the graPHIGS API transfers every structure. The conflicting structure replaces the new one without changing any associations to a workstation or view.

> **Note:** The graPHIGS API empties the structure in the target structure store *before* it copies the source structure. If an error occurs (such as the copying of a structure cannot be completed), then the original structure cannot be reconstructed.

When you replace a structure that is an open structure, the structure remains open and is emptied, but the graPHIGS API does not remove references to the structure. The result is as though you had issued the following:

1. **GPCLST** - Close Structure
2. **GPEST** - Empty Structure
3. **GPTST** - Transfer Structures
4. **GPOPST** - Open Structure

If your application transfers an execute structure-type element that refers to a non-existing structure, then the graPHIGS API creates an empty structure.

**Parameters**

*ssid* **— specified by user, fullword integer**
Structure store identifier.

*flag* **— specified by user, fullword integer**
Conflict resolution flag (`1=MAINTAIN`, `2=ABANDON`, `3=UPDATE`).

*number* **— specified by user, fullword integer**
Number of structure identifier entries in the list (>=1).

*lstrid* **— specified by user, array of fullword integers**
A list of structure identifiers.

**Error Codes**

**12** FUNCTION REQUIRES STATE SSSL

**120** WARNING, ONE OR MORE STRUCTURES DO NOT EXIST

**127** CONFLICT RESOLUTION FLAG IS INVALID

**128** STRUCTURE CONFLICT OCCURS WHEN RESOLUTION FLAG IS ABANDON

**134** NUMBER OF ENTRIES IN LIST < ONE

**222** SPECIFIED STRUCTURE STORE DOES NOT EXIST

**594** DATA EXCEEDS CONNECTION BUFFER SIZE

**614** UNKNOWN ELEMENT FOUND IN STRUCTURE *n1*

**Related Subroutines**

**GPTAST**
Transfer All Structures

**RCP code**

201348609 (X'0C005601')

## GPTXCS - Set Text Character Set

**GPTXCS (*csid*)**

**Purpose**

Use **GPTXCS** to set the current text character set. This subroutine only sets a shell state and gets bound to a text primitive when the graPHIGS API creates it. The graPHIGS API uses this value to interpret character code points in subsequent text output primitives.

**Parameters**

*csid* **— specified by user, fullword integer**
Character set identifier.

See the Appendix A. "Character Set and Font Identifiers."

**Error Codes**

**542** CHARACTER SET IDENTIFIER IS INVALID

**Related Subroutines**

None

**RCP code**

201328657 (X'0C000811')

# Chapter 7. Archive Subroutines

The archive file subroutines allow your application to store structure data to a file on a disk when using the graPHIGS API. The same application or other applications can then retrieve this data at a later time. Since the graPHIGS API nucleus directly reads in archived structure data, performance can be improved (by reducing the overhead of data transmission from the shell to the nucleus). In addition, archive files save on the amount of memory needed by the application (storage that would have been used to hold the structure element data).

With the archive facility, your application can store structure data to, retrieve structure data from, or delete structure data from an archive file.

The Open Archive File (**GPOPAR**) subroutine gives you access to archive files. More than one archive file can be opened simultaneously. The Close Archive File (**GPCLAR**) subroutine terminates access to archive files.

With the appropriate data translations, any graPHIGS API system that supports archive files could use the archive file.

## GPARAS - Archive All Structures

**GPARAS (*arid*)**

### Purpose

Use **GPARAS** to store all structures from the currently selected structure store into the specified open archive file.

If any of the specified structures in the structure store already exists in the archive file, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution (**GPCNRS**) subroutine.

If an error occurs during the archival process, then the graPHIGS API issues a message and terminates the archival process. Any structures that were successfully transferred to the archive file are archived in their entirety.

> **Note:** The graPHIGS API empties the structure in the archive file before it copies the source structure. If an error occurs (such as the copying of a structure cannot be completed), then the original structure cannot be reconstructed.

Currently, the archive file resource and structure store *must* reside on the same nucleus. If the two resources are on different nuclei, then the graPHIGS API issues error 1109.

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

### Error Codes

| | |
|---|---|
| **7** | FUNCTION REQUIRES STATE AROP |
| **12** | FUNCTION REQUIRES STATE SSSL |
| **128** | STRUCTURE CONFLICT OCCURS WHEN RESOLUTION FLAG IS ABANDON |

| | |
|---|---|
| **220** | SPECIFIED ARCHIVE FILE DOES NOT EXIST |
| **594** | DATA EXCEEDS CONNECTION BUFFER SIZE |
| **614** | UNKNOWN ELEMENT FOUND IN STRUCTURE *n1* |
| **1109** | FUNCTION NOT SUPPORTED |
| **1113** | FILE IS READ ONLY |

### Related Subroutines

| | |
|---|---|
| **GPARSN** | Archive Structure Networks |
| **GPARST** | Archive Structures |
| **GPCNRS** | Set Conflict Resolution |

### RCP code

201348614 (X'0C005606')

## GPARSN - Archive Structure Networks

**GPARSN (*arid*, *number*, *lstrid*)**

### Purpose

Use **GPARSN** to store one or more structure networks from the currently selected structure store into the specified open archive file.

If any of the specified root structures do not exist in the currently selected structure store, then the graPHIGS API issues a warning and no action is taken for the non-existing root structures. If any of the specified structures in the structure networks already exists in the archive file, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution (**GPCNRS**) subroutine.

If an error occurs during the archival process, then the graPHIGS API issues a message and terminates the archival process. Any structures that were successfully transferred to the archive file are archived in their entirety.

> **Note:** The graPHIGS API empties the structure in the archive file before it copies the source structure. If an error occurs (such as the copying of a structure cannot be completed), then the original structure cannot be reconstructed.

Currently, the archive file resource and structure store *must* reside on the same nucleus. If the two resources are on different nuclei, then the graPHIGS API issues error 1109.

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

*number* **— specified by user, fullword integer**
Number of root structure identifier entries in the list (>=1).

*lstrid* **— specified by user, array of fullword integers**
List of root structure identifiers.

### Error Codes

| | |
|---|---|
| **7** | FUNCTION REQUIRES STATE AROP |
| **12** | FUNCTION REQUIRES STATE SSSL |
| **120** | WARNING, ONE OR MORE STRUCTURES DO NOT EXIST |
| **128** | STRUCTURE CONFLICT OCCURS WHEN RESOLUTION FLAG IS ABANDON |
| **134** | NUMBER OF ENTRIES IN LIST < ONE |
| **220** | SPECIFIED ARCHIVE FILE DOES NOT EXIST |
| **594** | DATA EXCEEDS CONNECTION BUFFER SIZE |
| **614** | UNKNOWN ELEMENT FOUND IN STRUCTURE *n1* |
| **1109** | FUNCTION NOT SUPPORTED |
| **1113** | FILE IS READ ONLY |

### Related Subroutines

| | |
|---|---|
| **GPARAS** | Archive All Structures |
| **GPARST** | Archive Structures |
| **GPCNRS** | Set Conflict Resolution |

### RCP code

201348613 (X'0C005605')

## GPARST - Archive Structures

**GPARST (*arid*, *number*, *lstrid*)**

### Purpose

Use **GPARST** to store one or more structures from the currently selected structure store into the specified open archive file.

If any of the specified structures do not exist in the currently selected structure store, then the graPHIGS API issues a warning and no action is taken for the non-existing structures. If one of the specified structures already exists in the archive file, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution (**GPCNRS**) subroutine.

If an error occurs during the archival process, then the graPHIGS API issues a message and terminates the archival process. Any structures that were successfully transferred to the archive file are archived in their entirety.

> **Note:** The graPHIGS API empties the structure in the archive file before it copies the source structure. If an error occurs (such as, the copying of a structure cannot be completed), then the graPHIGS API cannot reconstruct the original structure.

Currently, the archive file resource and structure store *must* reside on the same nucleus. If the two resources are on different nuclei, then the graPHIGS API issues error 1109.

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

*number* **— specified by user, fullword integer**
Number of structure identifier entries in the list (>=1).

*lstrid* **— specified by user, array of fullword integers**
List of structure identifiers.

### Error Codes

| | |
|---|---|
| **7** | FUNCTION REQUIRES STATE AROP |
| **12** | FUNCTION REQUIRES STATE SSSL |
| **120** | WARNING, ONE OR MORE STRUCTURES DO NOT EXIST |
| **128** | STRUCTURE CONFLICT OCCURS WHEN RESOLUTION FLAG IS ABANDON |
| **134** | NUMBER OF ENTRIES IN LIST < ONE |
| **220** | SPECIFIED ARCHIVE FILE DOES NOT EXIST |
| **594** | DATA EXCEEDS CONNECTION BUFFER SIZE |
| **614** | UNKNOWN ELEMENT FOUND IN STRUCTURE *n1* |
| **1109** | FUNCTION NOT SUPPORTED |
| **1113** | FILE IS READ ONLY |

### Related Subroutines

| | |
|---|---|
| **GPARAS** | Archive All Structures |
| **GPARSN** | Archive Structure Networks |
| **GPCNRS** | Set Conflict Resolution |

### RCP code

201348612 (X'0C005604')

## GPCNRS - Set Conflict Resolution

**GPCNRS (*aflag*, *rflag*)**

### Purpose

Use **GPCNRS** to set the conflict resolution flags. The graPHIGS API uses these flags to determine how to resolve conflicts when you are archiving or retrieving one or more structures (e.g., when a structure that exists in a specified archive file has the same identifier as a structure that exists in the current structure store).

There are two conflict resolution flags: *aflag* for moving structure data to an archive file from a structure store and *rflag* for moving structure data from an archive file to a structure store. Your application can set either of these flags to: `1=MAINTAIN`, `2=ABANDON`, or `3=UPDATE`.

- If the value is set to `1=MAINTAIN`, then the graPHIGS API only transfers the structures that do not conflict.
- If the value is set to `2=ABANDON`, then the graPHIGS API does not transfer any structures when a conflict occurs.
- If the value is set to `3=UPDATE`, then the graPHIGS API transfers all the structures and replaces any conflicting structures with the new ones.

If this subroutine is not used, then the archival conflict resolution flag (*aflag*) defaults to a value of 3=UPDATE and the retrieval conflict resolution flag (*rflag*) defaults to a value of 2=ABANDON.

### Parameters

*aflag* **— returned by the graPHIGS API, fullword integer**
Archive conflict resolution (1=MAINTAIN, 2=ABANDON, 3=UPDATE).

*rflag* **— returned by the graPHIGS API, fullword integer**
Retrieve conflict resolution (1=MAINTAIN, 2=ABANDON, 3=UPDATE).

### Error Codes

| | |
|---|---|
| **127** | CONFLICT RESOLUTION FLAG IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPARAS** | Archive All Structures |
| **GPARSN** | Archive Structure Networks |
| **GPARST** | Archive Structures |
| **GPQCNR** | Inquire Conflict Resolution |
| **GPRVAS** | Retrieve All Structures |
| **GPRVSN** | Retrieve Structure Networks |
| **GPRVST** | Retrieve Structures |

### RCP code

201330965 (X'0C001115')

## GPDASA - Delete All Structures from Archive

**GPDASA (*arid*)**

### Purpose

Use**GPDASA** to delete all structures from the specified open archive file.

This subroutine is equivalent to invoking the Delete Structures from Archive (**GPDSAR**) subroutine for each structure in the selected structure store.

When this subroutine call is completed, the state of the archive file is as if your application just created it.

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

### Error Codes

| | |
|---|---|
| **7** | FUNCTION REQUIRES STATE AROP |
| **220** | SPECIFIED ARCHIVE FILE DOES NOT EXIST |
| **1113** | FILE IS READ ONLY |

### Related Subroutines

**GPDSAR** Delete Structures from Archive
**GPDSNA** Delete Structure Networks from Archive

### RCP code

201332994 (X'0C001902')

## GPDSAR - Delete Structures from Archive

**GPDSAR (*arid*, *number*, *lstrid*)**

### Purpose

Use **GPDSAR** to delete one or more structures from the specified open archive file.

> **Note:** The graPHIGS API does not check if other structures in the archive file reference the deleted structure. Therefore, the graPHIGS API does not delete the execute structure-type elements in other archived structures that reference the deleted structures.

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

*number* **— specified by user, fullword integer**
Number of structure identifier entries in the list (>=1).

*lstrid* **— specified by user, array of fullword integers**
List of structure identifiers.

### Error Codes

**7** FUNCTION REQUIRES STATE AROP
**134** NUMBER OF ENTRIES IN LIST < ONE
**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST
**1113** FILE IS READ ONLY

### Related Subroutines

**GPCEXS** Conditional Execute Structure
**GPDASA** Delete All Structure from Archive
**GPDSNA** Delete Structure Networks from Archive
**GPEXST** Execute Structure

### RCP code

201332738 (X'0C001802')

## GPDSNA - Delete Structure Networks from Archive

**GPDSNA (*arid*, *number*, *lstrid*)**

### Purpose

Use **GPDSNA** to delete one or more structure networks from the specified open archive file.

> **Note:** The graPHIGS API does not check if other structures in the archive file reference the deleted structures. Therefore, the graPHIGS API does not delete the execute structure-type elements in other archived structures that reference the deleted structures.

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

*number* **— specified by user, fullword integer**
Number of root structure identifier entries in the list (>=1).

*lstrid* **— specified by user, array of fullword integers**
List of root structure identifiers.

### Error Codes

| | |
|---|---|
| **7** | FUNCTION REQUIRES STATE AROP |
| **134** | NUMBER OF ENTRIES IN LIST < ONE |
| **220** | SPECIFIED ARCHIVE FILE DOES NOT EXIST |
| **1113** | FILE IS READ ONLY |

### Related Subroutines

| | |
|---|---|
| **GPCEXS** | Conditional Execute Structure |
| **GPDASA** | Delete All Structure from Archive |
| **GPDSAR** | Delete Structures from Archive |
| **GPEXST** | Execute Structure |

### RCP code

201331971 (X'0C001503')

## GPRVAS - Retrieve All Structures

**GPRVAS (*arid*)**

### Purpose

Use **GPRVAS** to retrieve all structures from the specified open archive file and place them into the the currently selected structure store.

If any of the specified structures in the archive file already exists in the structure store, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution (**GPCNRS**) subroutine.

> **Note:** The graPHIGS API empties the structure in the target structure store before it copies the source structure. If an error occurs (such as the copying of a structure cannot be completed), then the original structure cannot be reconstructed.

When you retrieve a structure that is an open structure, the structure is closed, emptied, retrieved, and reopened and the graPHIGS API maintains all references to the structure. The result is as though you had issued the following:

1. **GPCLST** - Close Structure
2. **GPEST** - Empty Structure
3. **GPRVST** - Retrieve Structures
4. **GPOPST** - Open Structure

If the structure retrieved contains an execute structure-type element that references a non-existing structure, then the graPHIGS API creates an empty structure.

Currently, the archive file resource and the structure store *must* reside on the same nucleus. If the two resources are on different nuclei, then the graPHIGS API issues error 1109.

**Parameters**

*arid* **— specified by user, fullword integer**
Archive file identifier.

**Error Codes**

**7** FUNCTION REQUIRES STATE AROP

**12** FUNCTION REQUIRES STATE SSSL

**128** STRUCTURE CONFLICT OCCURS WHEN RESOLUTION FLAG IS ABANDON

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**594** DATA EXCEEDS CONNECTION BUFFER SIZE

**614** UNKNOWN ELEMENT FOUND IN STRUCTURE *n1*

**1109** FUNCTION NOT SUPPORTED

**Related Subroutines**

**GPCNRS**
Set Conflict Resolution

**GPRVSN**
Retrieve Structure Networks

**GPRVST**
Retrieve Structures

**RCP code**

201348617 (X'0C005609')

## GPRVSN - Retrieve Structure Networks

**GPRVSN (*arid*, *number*, *lstrid*)**

**Purpose**

Use **GPRVSN** to retrieve one or more structure networks from the specified open archive file and place them into the currently selected structure store.

If any of the specified root structures do not exist in the specified archive file and the specified structure identifier does not exist in the current structure store, then the graPHIGS API issues a warning and creates an empty structure. If any of the specified root structures do not exist in the specified archive file and the

specified structure identifier does exist in the current structure store and the conflict resolution flag is set to 3=UPDATE, then the graPHIGS API empties the structure.

If any of the specified structures in the structure networks already exists in the structure store, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution (**GPCNRS**) subroutine.

> **Note:** The graPHIGS API empties the structure in the target structure store before it copies the source structure. If an error occurs (such as the copying of a structure cannot be completed), then the original structure cannot be reconstructed.

When you retrieve a structure that is an open structure, the structure is closed, emptied, retrieved, and reopened and the graPHIGS API maintains all references to the structure. The result is as though you had issued the following:

1. **GPCLST** - Close Structure
2. **GPEST** - Empty Structure
3. **GPRVST** - Retrieve Structures
4. **GPOPST** - Open Structure

If the structure retrieved contains an execute structure-type element that references a non-existing structure, then the graPHIGS API creates an empty structure.

Currently, the archive file resource and the structure store *must* reside on the same nucleus. If the two resources are on different nuclei, then the graPHIGS API issues error 1109.

**Parameters**

*arid* **— specified by user, fullword integer**
Archive file identifier.

*number* **— specified by user, fullword integer**
Number of root structure identifier entries in the list (>=1).

*lstrid* **— specified by user, array of fullword integers**
A list of root structure identifiers.

**Error Codes**

**7** FUNCTION REQUIRES STATE AROP

**12** FUNCTION REQUIRES STATE SSSL

**120** WARNING, ONE OR MORE STRUCTURES DO NOT EXIST

**128** STRUCTURE CONFLICT OCCURS WHEN RESOLUTION FLAG IS ABANDON

**134** NUMBER OF ENTRIES IN LIST < ONE

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**594** DATA EXCEEDS CONNECTION BUFFER SIZE

**614** UNKNOWN ELEMENT FOUND IN STRUCTURE *n1*

**1109** FUNCTION NOT SUPPORTED

**Related Subroutines**

**GPCNRS**
Set Conflict Resolution

**GPRVAS**
Retrieve All Structures

**GPRVST**
Retrieve Structures

**RCP code**

201348616 (X'0C005608')

## GPRVST - Retrieve Structures

**GPRVST (*arid*, *number*, *lstrid*)**

### Purpose

Use **GPRVST** to retrieve one or more structures from the specified open archive file and place them into the currently selected structure store.

If any of the specified structures do not exist in the specified archive file and the specified structure identifier does not exist in the current structure store, then the graPHIGS API issues a warning and creates an empty structure. If any of the specified structures do not exist in the specified archive file and the specified structure identifier does exist in the current structure store and the conflict resolution flag is set to `3=UPDATE`, then the graPHIGS API empties the structure.

If one of the specified structures already exists in the structure store, then the graPHIGS API resolves the conflict according to the value of the archive conflict resolution flag specified by the Set Conflict Resolution (**GPCNRS**) subroutine.

> **Note:** The graPHIGS API empties the structure in the target structure store before it copies the source structure. If an error occurs (such as the copying of a structure cannot be completed), then the original structure cannot be reconstructed.

When you retrieve a structure that is an open structure, the structure is closed, emptied, retrieved, and reopened and the graPHIGS API maintains all references to the structure. The result is as though you had issued the following:

1. **GPCLST** - Close Structure
2. **GPEST** - Empty Structure
3. **GPRVST** - Retrieve Structures
4. **GPOPST** - Open Structure

If the structure retrieved contains an execute structure-type element that references a non-existing structure, then the graPHIGS API creates an empty structure.

Currently, the archive file resource and the structure store *must* reside on the same nucleus. If the two resources are on different nuclei, then the graPHIGS API issues error 1109.

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

*number* **— specified by user, fullword integer**
Number of structure identifier entries in the list (>=1).

*lstrid* **— specified by user, array of fullword integers**
A list of structure identifiers.

**Error Codes**

**7** FUNCTION REQUIRES STATE AROP

**12** FUNCTION REQUIRES STATE SSSL

**120** WARNING, ONE OR MORE STRUCTURES DO NOT EXIST

**128** STRUCTURE CONFLICT OCCURS WHEN RESOLUTION FLAG IS ABANDON

**134** NUMBER OF ENTRIES IN LIST < ONE

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**594** DATA EXCEEDS CONNECTION BUFFER SIZE

**614** UNKNOWN ELEMENT FOUND IN STRUCTURE *n1*

**1109** FUNCTION NOT SUPPORTED

**Related Subroutines**

**GPCNRS**
Set Conflict Resolution

**GPRVAS**
Retrieve All Structures

**GPRVSN**
Retrieve Structure Networks

**RCP code**

201348615 (X'0C005607')

# Chapter 8. Workstation Table Operations

When your application opens a workstation, the graPHIGS API creates workstation tables that describe that workstation. Each workstation table has default settings. Use the subroutines in this section to modify some of the table entries according to your application's specifications.

Most workstation table indexes begin with 1. The following exceptions begin with 0:

- color tables
- color processing tables
- depth cue tables
- view tables

Some table entries may not be modified.

The subroutines in this section do *not* create structure elements or modify structure content. The changed table values only take effect after you update the workstation. The application may inquire the default table settings by issuing the appropriate inquiry programming calls. For a listing of the default tables for each supported workstation type, see *The graPHIGS Programming Interface: Technical Reference*.

## GPCML - Set Color Model

**GPCML (*wsid*, *model*)**

### Purpose

Use **GPCML** to set the current color model for the specified workstation to the given color model.

This subroutine enables the application to specify the color model that the graPHIGS API uses to interpret the following colors when your application sets or inquires these colors by the corresponding API subroutines:

- colors in bundle tables
- light source color
- depth cue color
- view colors
- input echo colors

The color model set with **GPCML** does not apply to colors in primitive definitions or individual direct color attributes.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*model* **— specified by user, fullword integer**
Color model (`1=RGB, 2=HSV, 3=CMY, 4=CIELUV`).

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **318** | COLOR MODEL INVALID |

### Related Subroutines

| | |
|---|---|
| **GPQCML** | Inquire Color Model |
| **GPTXCD** | Set Text Color Direct |

### RCP code

201329415 (X'0C000B07')

## GPCPR - Set Color Processing Representation

**GPCPR (*wsid*, *index*, *model*, *quant*, *data*)**

### Purpose

Use **GPCPR** to set the given color processing values into the specified entry of the workstation's color processing table.

The color processing table is 0 based, however, you cannot change entry 0 of the table. Entry 0 supports a color rendering model (*model*) of `1=RGB_NORMAL`, and a color quantization method (*quant*) of `1=WORKSTATION_DEPENDENT`. For information on color processing, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*index* — **specified by user, fullword integer**
Color processing mode table index (>=1).

*model* — **specified by user, fullword integer**
Rendering color model (`1=RGB_NORMAL, 2=RGB_B_ONLY`).

*quant* — **specified by user, fullword integer**
Color quantization method (`1=WORKSTATION_DEPENDENT, 2=BITWISE`).

*data* — **specified by user, 4 fullword integers**
Color quantization parameters. The content of this parameter is dependent on the color quantization method.

**If *quant*=1 (`WORKSTATION_DEPENDENT`)**
The *data* parameter is ignored.

**If *quant*=2 (`BITWISE`)**
The *data* parameter requires the following format:

```
 0 |R bit length| fullword integer

 4 |G bit length| fullword integer

 8 |B bit length| fullword integer

12 |padding bits| fullword integer
```

**Note:** The least significant (right most) bits of the padding data are used.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **265** | COLOR PROCESSING INDEX < ZERO |
| **266** | COLOR PROCESSING INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY |
| **267** | SPECIFIED RENDERING COLOR MODEL IS NOT SUPPORTED |
| **268** | SPECIFIED QUANTIZATION METHOD IS NOT SUPPORTED |
| **269** | ONE OF QUANTIZATION PARAMETERS IS INVALID |
| **275** | SPECIFIED ENTRY CANNOT BE CHANGED |

### Related Subroutines

| | |
|---|---|
| **GPCPI** | Set Color Processing Index |
| **GPQCPF** | Inquire Color Processing Facilities |
| **GPQCPR** | Inquire Color Processing Representation |
| **GPQCQM** | Inquire Available Color Quantization Methods |
| **GPQRCM** | Inquire Available Rendering Color Models |
| **GPXVR** | Set Extended View Representation |

### RCP code

201329418 (X'0C000B0A')

## GPCRC - Create Color Table

**GPCRC (*wsid*, *ctid*, *model*, *length*)**

### Purpose

Use **GPCRC** to create a color table with a specified color model and length for a specified workstation. All entries are initialized to the linear scale between the minimum and maximum values supported by the workstation.

This color table may be used for image processing. For information on using color tables for image processing, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ctid* **— specified by user, fullword integer**
Color table identifier (>=1).

*model* **— specified by user, fullword integer**
Color model (`1=RGB`).

*length* **— specified by user, fullword integer**
Length of color table in log2 (>0).

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |

| | |
|---|---|
| **282** | COLOR TABLE IDENTIFIER < ONE |
| **283** | COLOR TABLE IDENTIFIER ALREADY IS IN USE |
| **285** | SPECIFIED COLOR MODEL IS NOT SUPPORTED |
| **286** | COLOR TABLE SIZE EXCEEDS THE WORKSTATION MAXIMUM |
| **287** | COLOR TABLE SIZE < ONE |

### Related Subroutines

| | |
|---|---|
| **GPDFI** | Define Image |
| **GPDLC** | Delete Color Table |

### RCP code

201329425 (X'0C000B11')

## GPCSR - Set Cull Size Representation

**GPCSR (*wsid*, *index*, *size*)**

### Purpose

Use **GPCSR** to store a cull size threshold into the specified entry in the workstation's cull size table. Each entry of the cull size table is initially set to a cull size of 0.01. For a complete discussion of conditional structure execution, see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Cull size table index. (>=1).

*size* **— specified by user, short floating-point number (DC)**
Cull size (>=0).

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **278** | CULL SIZE INDEX < ONE |
| **279** | CULL SIZE INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY |
| **280** | CULL SIZE < ZERO |

### Related Subroutines

| | |
|---|---|
| **GPQCSF** | Inquire Cull Size Facilities |
| **GPQCSR** | Inquire Cull Size Representation |
| **GPTEX2** | Test Extent 2 |
| **GPTEX3** | Test Extent 3 |

### RCP code

# GPDCR - Set Depth Cue Representation

**GPDCR (*wsid*, *index*, *id*, *value*)**

### Purpose

Use **GPDCR** to set one group of the specified entry in the workstation's depth cue table.

The values in the entry are applied during traversal when the current depth cue index is set to the specified entry. Each entry of the depth cue table is initially set to a depth cue mode of1=SUPPRESSED.

The depth cue table is 0 based, however, entry 0 of the table cannot be changed. It always contains a depth cue mode of 1=SUPPRESSED.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Depth cue table index(>=1).

*id* **— specified by user, fullword integer**
Group identifier(1=DEPTH_CUE_MODE, 2=DEPTH_CUE_REFERENCE_PLANE, 3=DEPTH_CUE_SCALE_FACTOR, 4=DEPTH_CUE_COLOR).

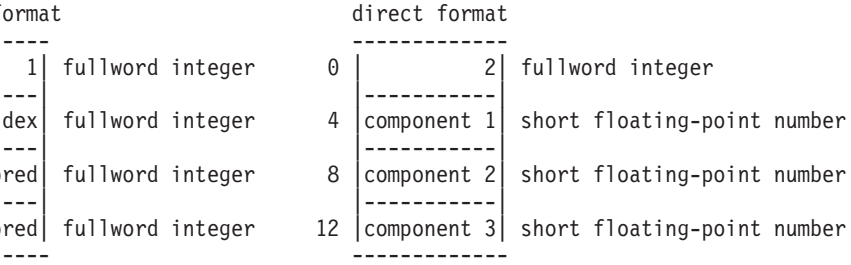
*value* **— specified by user, array of fullword quantities**
Value(s) to be set in the group of the specified group *id*The values for each group identifier (*id*)have a unique data format, listed as follows:

**Group Identifier 1 -Depth cue mode**
A fullword integer(1=SUPPRESSED, 2=ALLOWED)/

**Group Identifier 2 -Depth cue reference planes (NPC)**
Two short floating-point numbers specifying the far and near depth cue reference plane distance (0.0<=far<near<=1.0)

**Group Identifier 3 -Depth cue scale factors**
Two short floating-point numbers specifying two scale factors corresponding to the far and near reference planes(0.0<=*factor*<=1.0).

**Group Identifier 4 -Depth cue color**
Four fullwords of data with either of the following formats:




### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **92** | COLOR INDEX < ZERO |
| **93** | COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY |
| **96** | COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL |
| **97** | COLOR FORMAT PARAMETER IS INVALID |
| **261** | DEPTH CUE INDEX < ZERO |
| **262** | DEPTH CUE INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY |
| **263** | DEPTH CUE REFERENCE PLANE IS INVALID |
| **264** | DEPTH CUE SCALE FACTOR IS INVALID |
| **272** | GROUP IDENTIFIER IS INVALID |
| **275** | SPECIFIED ENTRY CANNOT BE CHANGED |
| **281** | DEPTH CUE MODE IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPCML** | Set Color Model |
| **GPDCI** | Set Depth Cue Index |
| **GPQDCF** | Inquire Depth Cue Facilities |
| **GPQDCR** | Inquire Depth Cue Representation |

### RCP code

201329420(X'0C000B0C')

## GPDLC - Delete Color Table

**GPDLC (*wsid*, *ctid*)**

### Purpose

Use **GPDLC** to delete the specified color table from the specified workstation. If the color table is used to define any images, they are automatically canceled.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ctid* **— specified by user, fullword integer**
Color table identifier (>=1).

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **282** | COLOR TABLE IDENTIFIER < ONE |
| **284** | COLOR TABLE IDENTIFIER DOES NOT EXIST |

### Related Subroutines

| | |
|---|---|
| **GPCRC** | Create Color Table |

**RCP code**

201329426(X'0C000B12')

## GPDMR - Set Data Mapping Representation

**GPDMR (*wsid, index, method, mdata, clengths, ctype, cdata*)**

### Purpose

Use **GPDMR** to set the specified data mapping values into the specified entry of the data mapping table.

The data mapping representation provides the values that the graPHIGS API uses to perform data mapping on area primitives. The entry used is selected by the data mapping index specified by Set Data Mapping Index (**GPDMI**) or Set Back Data Mapping Index (**GPBDMI**).

During traversal, if the data mapping method is inconsistent with the contents of the primitive, then the graPHIGS API renders the primitive using data mapping method `1=DM_METHOD_COLOR`.

If the *method* parameter has the value `1=DM_METHOD_COLOR`, then you do not need to supply any additional data mapping data: the graPHIGS API ignores the values in the parameters following the *method* parameter.

The data mapping table is zero-based; however, entry zero cannot be changed. Entry zero always contains a data mapping method of `1=DM_METHOD_COLOR`. Use **GPQWDT** to inquire the data mapping facilities of a specific workstation.

See *The graPHIGS Programming Interface: Understanding Concepts* for more detailed information.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Data mapping index (>=1).

*method* **— specified by user, fullword integer**
Data mapping method (`-1=IMAGE_ARRAY, 1=DM_METHOD_COLOR, 2=SINGLE_VALUE_UNIFORM, 4=BI_VALUE_UNIFORM`).

*mdata* **— specified by user, variable data**
Data mapping method descriptor. This parameter must have one of the following formats, depending on the specified data mapping method:

-1=IMAGE_ARRAY

The following data is specified:

| | | |
|---|---|---|
| WORDS 1 | 'udindex' | Fullword integer |
| 2 | 'vdindex' | Fullword integer |
| 3 | 'ulolim' | Floatingpoint number |
| 4 | 'uhilim' | Floatingpoint number |
| 5 | 'vlolim' | Floatingpoint number |
| 6 | 'vhilim' | Floatingpoint number |

*udindex* Index into the primitive's data list for the *u* data value (>=1)

*vdindex* Index into the primitive's data list for the *v* data value (>=1)

*ulolim* Lower limit of the *u* data mapping range/

*uhilim* Upper limit of the *u* data mapping range/

*vlolim* Lower limit of the *v* data mapping range/

*vhilim* Upper limit of the *v* data mapping range/

**1=DM_METHOD_COLOR**
N/A (No data mapping descriptor is returned for this method.)

**2=SINGLE_VALUE_UNIFORM**
The following data is required:

| | | |
|---|---|---|
| WORDS 1 | 'dindex' | Fullword integer |
| 2 | 'lolim' | Floatingpoint number |
| 3 | 'hilim' | Floatingpoint number |

*dindex* Index into the primitive's data list (>=1)

*lolim* Lower limit of the data mapping range

*hilim* Upper limit of the data mapping range

**4=BI_VALUE_UNIFORM**
The data mapping record required is identical to -1=IMAGE_ARRAY.

*clengths* **— specified by user, variable data**
Data mapping color data lengths. The format of this parameter is dependent on the*method* parameter:

-1=IMAGE_ARRAY

The following data is required:

| | | |
|---|---|---|
| WORDS 1 | 'x_size' | Fullword integer |
| 2 | 'y_size' | Fullword integer |
| 3 | 'oformat' | Fullword integer |

*x_size* *x* dimension of the base color data array (>=1)

*y_size* *y* dimension of the base color data array (>=1)

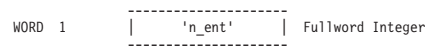
*oformat*

Data organization format (1=BASE_DATA, 2=SQUARE_MM, 3=RECT_MM). This format determines the filtering methods which may be used.

If data organization format selected is 2=SQUARE_MM or 3=RECT_MM, then *x_size* and *y_size* must be powers of 2.

**1=DM_METHOD_COLOR**
No data is required.

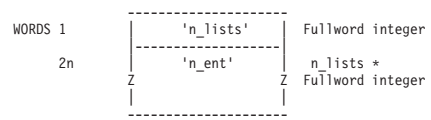
**2=SINGLE_VALUE_UNIFORM**
The following data is required:




*n_ent*

The number of entries in the color data list (>=1).

**4=BI_VALUE_UNIFORM**
The following data is required:




*n_lists*

The number of lists of data values (>=1).

*n_ent*

A list of fullword integers that specify the number of entries of each color data list (>=1). There are *n_lists* entries in this list.

*ctype* **— returned by the graPHIGS API, fullword integer**
Data mapping color data type. Supported types are:

**1=TYPE_COLOR**
Colors consist of three short floating-point numbers in the current workstation color model (0<=color_component<=1). During traversal, the current transparency coefficient is used with these color values. See **GPTCO** for more information.

**2=TYPE_PACKED_RGB**
Colors consist of four bytes. The first three bytes represent the red, green, and blue color components respectively; the fourth byte is ignored. During traversal, the current transparency coefficient is used with these color values. See **GPTCO** for more information.

**3=TYPE_COLOR_TRANS**
Colors consist of four short floating-point numbers. The first three numbers represent the color in the current workstation color model (0.0<=color_component<=1.0); the fourth number represents the transparency coefficient (0.0<=transparency_coefficient<=1.0). During traversal, these values override the current transparency coefficient. See **GPTCO** for more information.

**4=TYPE_PACKED_RGB_ALPHA**
Colors consist of four bytes. The first three bytes represent the red, green, and blue color components respectively; the fourth byte is an unsigned integer alpha ([default]) blending value that may be derived from a transparency coefficient as follows:

```
Alpha = X'FF' [default] (1.0-transparency_coefficient)
```

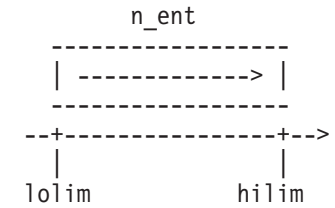
(Alpha = X'00') is fully transparent and equivalent to (transparency_coefficient=1.0). (Alpha = X'FF') is fully opaque and equivalent to (transparency_coefficient=0.0). During traversal, these values override the current transparency coefficient. See **GPTCO** for more information.

*cdata* **— specified by user, variable data**

Data mapping color data. The data mapping color data organization is defined by the *clengths* and *ctype* parameters.

The 2=SINGLE_VALUE_UNIFORM and 4=BI_VALUE_UNIFORM color lists are supplied from the *lolim* lower limits to the *hilim* upper limits. For example, the color representing the data value *lolim* is first in each list, and the color representing *hilim* is last. See Figure 3 and Figure 4. The number of color values in each 2=SINGLE_VALUE_UNIFORM color list is given by *n_ent*.

**Figure 3. SINGLE_VALUE_UNIFORM Color Data Organization**

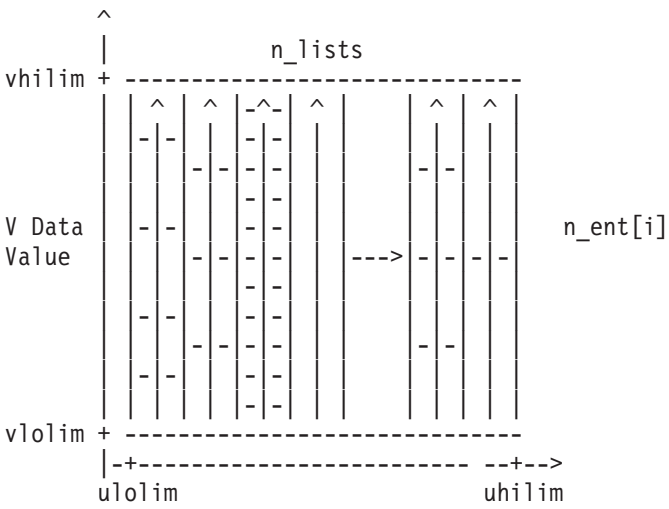



**U Data Value**

The number of color values in each of the 4=BI_VALUE_UNIFORM color lists is specified by *n_ent[i]*, so that the total number of color values in this color array is:

```
(n_ent[1] + n_ent[2] + n_ent[3] + ... + n_ent[n_lists])
```

**Figure 4. BI-VALUE_UNIFORM Color Data Organization**

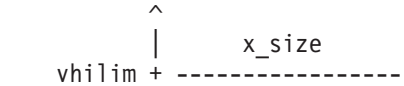


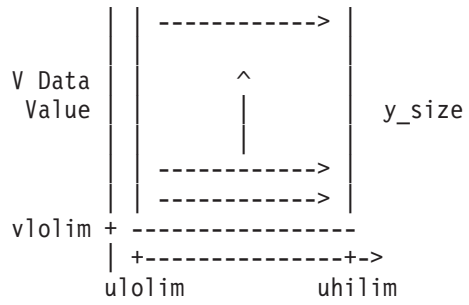

**U Data Value**

The IMAGE_ARRAY color arrays are organized according to the *oformat* field of the *clengths* parameter. BASE_DATA array color data is supplied in row order left-to-right and bottom-to-top. See Figure 5 The number of color values in this array is:

*(x_size * y_size)*

**Figure 5. IMAGE_ARRAY BASE_DATA Color Data Organization**






**U Data Value**

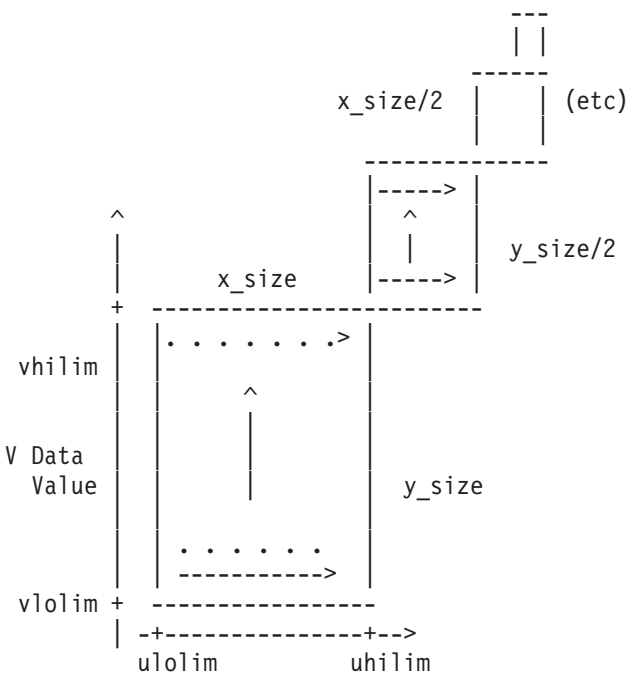
SQUARE_MM color mipmap data is supplied in the same fashion, starting with the base image and continuing with each successively smaller mipmap image, until either *x_size* or *y_size* is equal to one. See Figure 6. The number of color values in this complete array is:

*(x_size * y_size) + (x_size * y_size)/4 + (x_size * y_size)/16 + ...*

which reduces to the integer portion of:

*((4 * x_size * y_size) - (MAX(x_size,y_size) / MIN(x_size,y_size))) / 3*

**Figure 6. IMAGE_ARRAY SQUARE_DATA Color Data Organization**

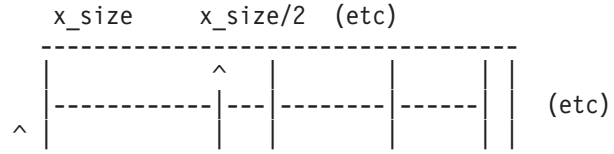


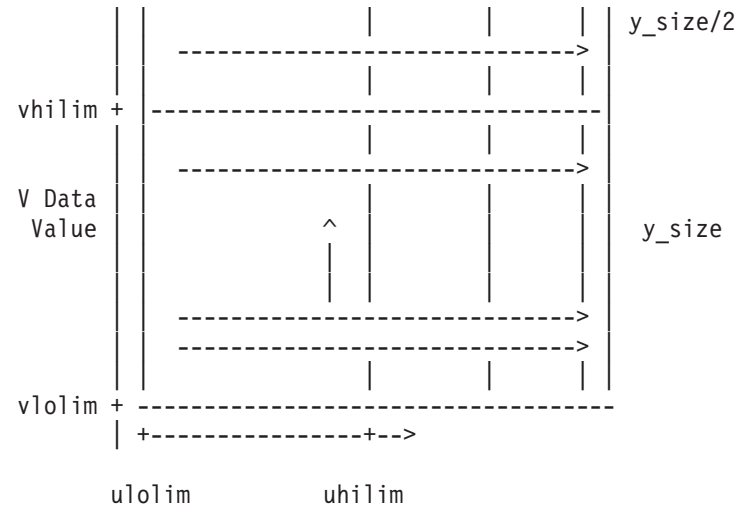

**U Data Value**

RECT_MM color mipmap data is supplied as a complete set, and organized in row order left-to-right and bottom-to-top (as though the entire set of mipmap images constituted a single base texture image). See Figure 7. The number of color values in this complete array is:

*((2 * x_size) - 1) * ((2 * y_size) - 1)*

**Figure 7. IMAGE_ARRAY RECT_DATA Color Data Organization**






**U Data Value**

## Error Codes

| | |
|---|---|
| **3** | FUNCTION REQUIRES STATE WSOP |
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **50** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **96** | COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL |
| **115** | TRANSPARENT COEFFICIENT IS INVALID |
| **275** | SPECIFIED ENTRY CANNOT BE CHANGED |
| **348** | MINIMUM PARAMETER LIMIT > MAXIMUM |
| **512** | METHOD NOT SUPPORTED |
| **630** | DATA MAPPING INDEX < ZERO |
| **634** | DATA MAPPING COLOR TYPE NOT SUPPORTED |
| **635** | DATA ORGANIZATION FORMAT IS INVALID |
| **637** | DATA LIST INDEX IS INVALID |
| **638** | COLOR DATA LENGTHS PARAMETER IS INVALID |

## Related Subroutines

| | |
|---|---|
| **GPBDFM** | Set Back Data Filtering Method |
| **GPBDMI** | Set Back Data Mapping Index |
| **GPBDM2** | Set Back Data Matrix 2 |
| **GPBTCO** | Set Back Transparency Coefficient |
| **GPDFM** | Set Data Filtering Method |
| **GPDMI** | Set Data Mapping Index |
| **GPDM2** | Set Data Matrix 2 |
| **GPQDMR** | Inquire Data Mapping Representation |
| **GPTCO** | Set Transparency Coefficient |

## RCP code

201345543 (X'0C004A07')

# GPGTXC - Set Geometric Text Culling

**GPGTXC (*wsid*, *height*, *method*)**

### Purpose

Use **GPGTXC** to set the geometric text culling height for the specified workstation.

When displaying geometric text, if the transformed height of the characters in Device Coordinates (DC) is less than the specified culling height, then the graPHIGS API replaces the text with an alternate representation if one was selected.

This subroutine is assigned escape identifier 1008.

> **Note:** This subroutine is an escape subroutine, and therefore, may not be available on all workstations. Use the Inquire List of Available Escape Subroutines (**GPQES**) subroutine to determine if this subroutine is supported by a specific workstation.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*height* **— specified by user, floating-point number (DC)**
Geometric text culling height (>=0.0).

*method* **— specified by user, fullword integer**
Display method for displaying geometric text when the height of the characters is less than the specified culling height.

**1=TEXT_DISPLAY**
each stroke of each character is displayed (the default method).

**2=BOX_DISPLAY**
a polyline is drawn around the text extent rectangle using the text color.

**3=NO_DISPLAY**
the character text is not displayed.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**105** GEOMETRIC TEXT CULLING DISPLAY METHOD IS INVALID

**106** GEOMETRIC TEXT CULLING HEIGHT < ZERO

### Related Subroutines

**GPCHH**
Set Character Height

**GPTX2**
Geometric Text 2

### RCP code

201345542 (X'0C004A06')

# GPHLF - Set Highlighting Filter

**GPHLF (*wsid*, *inclen*, *incl*, *exclen*, *excl*)**

### Purpose

Use GPHLF to set the inclusion and exclusion highlighting filters for the specified workstation. The new filters take effect when the workstation is updated.

The filters consist of class names which indicate which classes are to be included or excluded from highlighting. The same class names may exist in both the inclusion and exclusion filter. If a class name is in both filters when the workstation is updated, the class will be excluded. For more information on classes and class names see *The graPHIGS Programming Interface: Understanding Concepts*.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*inclen* **— specified by user, fullword integer**
Inclusion filter list length (>=0).

*incl* **— specified by user, array of fullword integers**
List of class names to be included in the highlighting filter (0<=class name<=workstation maximum).

*exclen* **— specified by user, fullword integer**
Exclusion filter list length (>=0).

*excl* **— specified by user, array of fullword integers**
List of class names to be excluded from the highlighting filter (0<=class name<=workstation maximum).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**321** FILTER VALUE IS INVALID

**531** FILTER LIST LENGTH < ZERO

### Related Subroutines

**GPADCN**
Add Class Name to Set

**GPQHLF**
Inquire Highlighting Filter

**GPQNCN**
Inquire Number of Available Class Names

**GPRCN**
Remove Class Name from Set

### RCP code

201335042 (X'0C002102')

# GPHR - Set Hatch Representation

**GPHR (*wsid*, *hatch*, *format*, *length*, *data*)**

### Purpose

Use **GPHR** to change a hatch pattern associated with a specific hatch index.

Some workstations require that the hatch fill be at a fixed size. The fixed sizes are a power of two. Therefore, specifying a hatch pattern with sizes that are a power of two produces the best results (i.e., no seams or discontinuity in the fill operation). See the Set Interior Style Index (**GPISI**) subroutine for the default hatch patterns in the workstation's hatch table.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

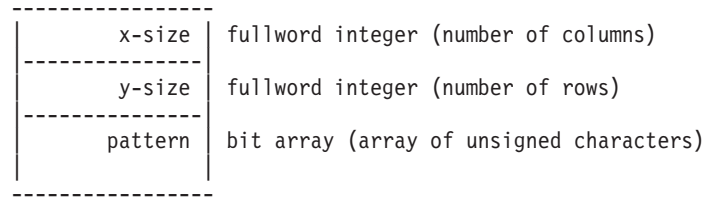
*hatch* **— specified by user, fullword integer**
Hatch table index (>=1).

*format* **— specified by user, fullword integer**
Format (1=BIT_ARRAY).

*length* **— specified by user, fullword integer**
Length of hatch pattern definition data in bytes.

*data* **— specified by user, variable data**
Hatch pattern definition data.

**1=BIT ARRAY**




**Note:** The bit array must be in row order with each row beginning on a byte boundary. Therefore, the size of the bit array is ((x hyphen size plus 7) slash 8 % % asterisk y hyphen size) bytes.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**47** HATCH INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**61** LENGTH IS INVALID

**71** FIELD IN DEFINITION DATA IS INVALID

**89** HATCH INDEX < ONE

**274** THIS FUNCTION IS NOT SUPPORTED BY THE WORKSTATION

**276** DEFINITION DATA FORMAT IS NOT SUPPORTED

**277** DEFINITION DATA EXCEEDS THE WORKSTATION TABLE CAPACITY

**Related Subroutines**

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPQHF**
Inquire Hatch Facilities

**GPQHR**
Inquire Hatch Representation

**RCP code**

201329421 (X'0C000B0D')

## GPIVF - Set Invisibility Filter

**GPIVF (*wsid*, *inclen*, *incl*, *exclen*, *excl*)**

**Purpose**

Use **GPIVF** to set the inclusion and exclusion invisibility filters for the specified workstation.

The new filters take effect when the application updates the workstation. The filters consist of class names which indicate which classes the graPHIGS API includes or excludes from invisibility. The same classes may exist in both the inclusion and exclusion filter. If a class is in both filters when the application updates the workstation, then the graPHIGS API excludes the class. For more information on classes and class names, see *The graPHIGS Programming Interface: Understanding Concepts*.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*inclen* **— specified by user, fullword integer**
Inclusion filter list length (>=0).

*incl* **— specified by user, array of fullword integers**
List of class names to be included in the invisibility filter (0<=class name<=workstation maximum).

*exclen* **— specified by user, fullword integer**
Exclusion filter list length (>=0).

*excl* **— specified by user, array of fullword integers**
List of class names to be excluded from the invisibility filter (0<=class name<=workstation maximum).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**321** FILTER VALUE IS INVALID

**531** FILTER LIST LENGTH < ZERO

**Related Subroutines**

**GPADCN**
Add Class Name to Set

**GPQIVF**
Inquire Invisibility Filter

**GPQNCN**
Inquire Number of Available Class Names

**GPRCN**
Remove Class Name from Set

**RCP code**

201335043 (X'0C002103')

## GPLNR - Set Linetype Rendering

**GPLNR (*wsid*, *ltype*, *style*, *data*)**

### Purpose

Use **GPLNR** to set the line rendering attributes for the specified line type table entry. The specified attributes are applied as part of rendering a line primitive. These attributes control the appearance of line primitives when rendered on the display.

The line rendering styles are:

- `1=WORKSTATION_DEPENDENT_RENDERING`
  Use the line type representation entry to render the line. The final appearance of the line is workstation-dependent and depends on the capabilities of the workstation.
- `2=SCALED_TO_FIT_RENDERING`
  Use the line type representation entry to render the line as follows:
  1. If the length of the line is less than the specified minimum threshold, then the graPHIGS API renders the line as a solid line.
  2. The line type pattern in the specified entry of the workstation's line type table is scaled such that the graPHIGS API renders the line using a whole number of repetitions of the scaled pattern. (The pattern may be scaled larger or smaller to fit.)

The line rendering style for line table entry 1 cannot be changed. It always defines the solid line pattern.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ltype* **— specified by user, fullword integer**
Line type. This parameter specifies the line type table entry to which the rendering attributes are applied (>=2).

*style* **— specified by user, fullword integer**
Rendering style identifier (`1=WORKSTATION_DEPENDENT_RENDERING, 2=SCALED_TO_FIT_RENDERING`).

*data* **— specified by user, array of data**
Rendering style definition data. Depending on the *style* parameter values you specified, rendering definition data is as follows:

**If *style*=1 (WORKSTATION_DEPENDENT_RENDERING)**
N/A (No data is required for workstation dependent rendering)

**If *style*=2 (SCALED_TO_FIT_RENDERING)**
The following data is required:

- Minimum threshold size (DC)

  A short floating-point number that specifies the minimum threshold size for scaling the line pattern (>=0)

- Line pattern unit size (DC)

  A short floating-point number that specifies the length of the line pattern unit of the specified entry of the line type representation table (>0)

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**63** LINETYPE VALUE < ONE

**64** SPECIFIED LINETYPE NOT AVAILABLE ON WORKSTATION

**71** FIELD IN DEFINITION DATA IS INVALID

**274** THIS FUNCTION IS NOT SUPPORTED BY THE WORKSTATION

**275** SPECIFIED ENTRY CANNOT BE CHANGED

**297** LINE RENDERING STYLE IS INVALID

### Related Subroutines

**GPLT** Set Linetype

**GPLTR**
Set Linetype Representation

**GPQLNR**
Inquire List of Line Rendering Styles

### RCP code

201329427 (X'0C000B13')

## GPLSR - Set Light Source Representation

**GPLSR (*wsid*, *index*, *type*, *color*, *data*)**

### Purpose

Use **GPLSR** to store the specified light source information into the specified entry of the workstation's light source table.

Each entry of the light source table is initially set to a light source type of `1=AMBIENT` and light source direct color values of 1.0, 1.0, and 1.0.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*index* — **specified by user, fullword integer**
Light source table index (>=1).

*type* — **specified by user, fullword integer**
Light source type (1=AMBIENT, 2=DIRECTIONAL, 3=POSITIONAL, 4=SPOT).

*color* — **specified by user, four fullwords of data**
Light source color. This parameter must have one of the following two formats:

indexed format

| | | |
|---|---|---|
| 0 | 1 | fullword integer |
| 4 | color index | fullword integer |
| 8 | ignored | fullword integer |
| 12 | ignored | fullword integer |

direct format

| | | |
|---|---|---|
| 0 | 2 | fullword integer |
| 4 | component 1 | short floating-point number |
| 8 | component 2 | short floating-point number |
| 12 | component 3 | short floating-point number |

*data* — **specified by user, array of short floating-point numbers**
Light source type dependent data. Required data for each light source type is listed below. They must be specified in the order shown in the list:

**1=AMBIENT**
none

**2=DIRECTIONAL**
Light source direction - 3 short floating-point numbers (WC)

**3=POSITIONAL**
Light source position - 3 short floating-point numbers (WC)

Attenuation coefficients - 2 short floating-point numbers. The first floating-point number must be >0. The second floating-point number must be >=0.

**4=SPOT** Light source position - 3 short floating-point numbers (WC)

Light source direction - 3 short floating-point numbers (WC)

Concentration exponent - short floating-point number (>=0)

Attenuation coefficients - 2 short floating-point numbers. The first floating-point number must be >0. The second floating-point number must be >=0.

Spread angle - short floating-point number. The spread angle must be in radians (0.0<=angle<=[default])

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**97** COLOR FORMAT PARAMETER IS INVALID

**254** LIGHT SOURCE INDEX < ONE

**255** LIGHT SOURCE INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY

**258** SPECIFIED LIGHT SOURCE TYPE IS NOT SUPPORTED

**259** ONE OF LIGHT SOURCE PARAMETERS IS INVALID

### Related Subroutines

**GPCML**
Set Color Model

**GPLMO**
Set Lighting Calculation Mode

**GPLSS**
Set Light Source State

**GPQLSF**
Inquire Light Source Facilities

**GPQLSR**
Inquire Light Source Representation

### RCP code

201329422 (X'0C000B0E')

## GPLTR - Set Linetype Representation

**GPLTR (*wsid*, *ltype*, *number*, *list*)**

### Purpose

Use **GPLTR** to set a one-dimensional pattern that defines a specific line type in the workstation's line type table.

The Set Linetype (**GPLT**) subroutine and the Set Edgetype (**GPELT**) subroutine specify a line type entry. This entry selects a line pattern that the graPHIGS API applies to subsequent lines, curves, and edges.

The pattern is defined by an array of integers which specify the length of each solid and void section. Each integer defines a multiple of the minimum size pattern section that your workstation can generate. The sections alternate between solid and void with the first section always being solid.

Line type table entry 1 cannot be changed. It always defines the solid line pattern.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ltype* **— specified by user, fullword integer**
Line type (>=2).

*number* **— specified by user, fullword integer**
Number of pattern sections. This parameter specifies the number of sections in the line pattern contained in *list* (even number>=2).

*list* **— specified by user, array of fullword unsigned integers**
Line pattern sections. Each entry of this array defines a section of the line pattern. The entries of the array alternate between solid and void sections with the first entry being solid. The value of

each entry defines the length of the section in multiples of the minimum section size supported by the workstation. The sum of the line pattern sections must be >=1.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**62** NUMBER OF LINE PATTERN SECTIONS IS INVALID

**63** LINETYPE VALUE < ONE

**64** SPECIFIED LINETYPE NOT AVAILABLE ON WORKSTATION

**274** THIS FUNCTION IS NOT SUPPORTED BY THE WORKSTATION

**275** SPECIFIED ENTRY CANNOT BE CHANGED

**277** DEFINITION DATA EXCEEDS THE WORKSTATION TABLE CAPACITY

**Related Subroutines**

**GPELT**
Set Edge Linetype

**GPLNR**
Set Linetype Rendering

**GPLT** Set Linetype

**GPQLTF**
Inquire Linetype Facilities

**GPQLTR**
Inquire Linetype Representation

**RCP code**

201329423 (X'0C000B0F')

## GPMTR - Set Marker Type Representation

**GPMTR (*wsid*, *mtype*, *format*, *length*, *data*)**

**Purpose**

Use **GPMTR** to set a two-dimensional pattern representation that defines a marker in the workstation's marker type table. The Set Marker Type (**GPMT**) subroutine specifies a marker type entry. This entry selects a marker pattern your application applies to subsequent markers.

Marker type table entry 3 cannot be changed. It always defines the asterisk marker.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

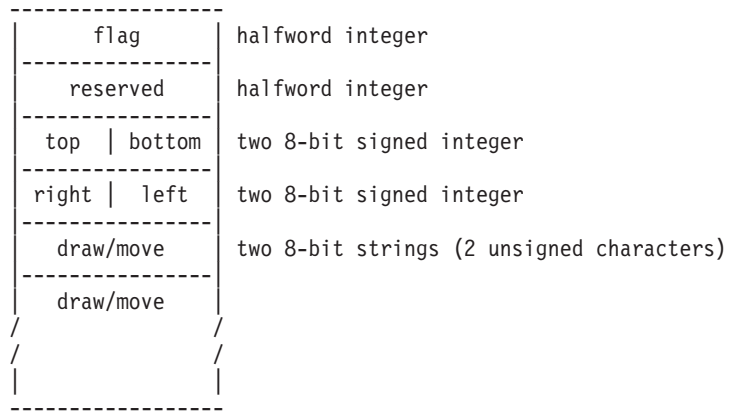
*mtype* **— specified by user, fullword integer**
Marker type table index (>=1, <>3).

*format* **— specified by user, fullword integer**
Format (1=VECTOR).

*length* **— specified by user, fullword integer**
Length of entire marker pattern definition data (even integer >=8 ) in bytes.

*data* **— specified by user, variable data**
Marker pattern definition data. This parameter must have the following format:

**Format 1=graPHIGS VECTOR FONT**



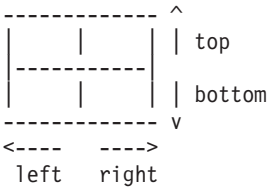

Each field specifies:

**flag**


If the fill flag=1, and the workstation supports filled characters, then the marker will be filled.

**reserved**
Must be 0.

**top, bottom, right, left**
Four 8-bit signed integers specifying the marker rectangle measured from the origin. The marker definition must be within this rectangle (top > bottom, right > left)




**draw/move**
A pair of 8-bit unsigned characters specifying a relative draw/move starting from the origin (the marker position) with the format *sxxxxxx*1 and *syyyyyyb*, where *s* is sign bit and *b* is blank bit. When the blank bit=1, then the *sxxxxxx* and *syyyyyy* strings specify a relative move; otherwise, they specify a relative draw. A workstation which supports scalable markers uses this maker definition format. The graPHIGS API maps the specified marker box height (top-bottom) to the nominal marker size of the workstation. The total data size cannot exceed the maximum marker pattern size supported by the workstation.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**61** LENGTH IS INVALID

**69** MARKER TYPE VALUE < ONE

**70** SPECIFIED MARKER TYPE NOT AVAILABLE ON WORKSTATION

**71** FIELD IN DEFINITION DATA IS INVALID

**274** THIS FUNCTION IS NOT SUPPORTED BY THE WORKSTATION

**275** SPECIFIED ENTRY CANNOT BE CHANGED

**276** DEFINITION DATA FORMAT IS NOT SUPPORTED

**277** DEFINITION DATA EXCEEDS THE WORKSTATION TABLE CAPACITY

**Related Subroutines**

**GPMT** Set Marker Type

**GPQMTF**
Inquire Marker Type Facilities

**GPQMTR**
Inquire Marker Type Representation

**GPQXTX**
Inquire Extended Text Facilities

**RCP code**

201329424 (X'0C000B10')

## GPPAR - Set Pattern Representation

**GPPAR (*wsid*, *index*, *numrow*, *numcol*, *strrow*, *strcol*, *nrow*, *ncol*, *array*)**

**Purpose**

Use **GPPAR** to set a given pattern definition in the specified entry in the workstation's pattern table.

The pattern is a grid of color indexes of dimension *nrow* x *ncol* within the array of dimension *numrow* x *numcol* starting at the position defined by *strrow*, *strcol*.

Some workstations required that the pattern fill is a fixed size. For these workstations, the graPHIGS API replicates the specified pattern to the fixed size.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Pattern index (>=1) Index of pattern table entry to be loaded.

*numrow* **— specified by user, fullword integer**
Number of rows in the input array (>=1).

*numcol* **— specified by user, fullword integer**
Number of columns in the input array (>=1).

*strrow* **— specified by user, fullword integer**
Row within array that is the start of the pattern (>=1).

*strcol* **— specified by user, fullword integer**
Column within array that is the start of the pattern (>=1).

*nrow* **— specified by user, fullword integer**
Number of rows within array to be used for pattern beginning at the starting position (>=1).

*ncol* **— specified by user, fullword integer**
Number of columns within array to be used for pattern beginning at the starting position (>=1).

*array* **— specified by user, array of fullword integers**
A grid of *numrow* x *numcol* color indexes. The array must be in row order. The pattern within this array begins at position *strrow*,*strcol*, and is of dimension *nrow* x *ncol*.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**48** PATTERN INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**85** PATTERN INDEX VALUE < ONE

**91** STARTING POINT OR DIMENSION < ONE

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**537** PATTERN OR PIXEL ARRAY EXCEEDS INPUT ARRAY SIZE

**Related Subroutines**

**GPQPAR**
Inquire Pattern Representation

**RCP code**

201329413 (X'0C000B05')

## GPVIP - Set View Input Priority

**GPVIP (*wsid*, *view*, *refview*, *flag*)**

**Purpose**

Use **GPVIP** to modify the input priority of the given view in relation to another view on the specified workstation.

When the application updates the workstation, the current view input priority for the specified view is set to the requested values.

This subroutine only changes the input priority of a view. There are no visual changes on the display surface.

**Note:** The Set View Priority (**GPVP**) subroutine is treated as a simultaneous invocation of the Set View Input Priority (**GPVIP**) and the Set View Output Priority (**GPVOP**) and subroutines.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*refview* **— specified by user, fullword integer**
Reference view index (>=0).

*flag* **— specified by user, fullword integer**
Relative priority of the specified view with respect to the reference view (`1=HIGHER, 2=LOWER`).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**59** VIEW INDEX VALUE < ZERO

**155** VIEW PRIORITY REFERENCE NUMBER IS INVALID

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**333** RELATIVE VIEW PRIORITY VALUE IS INVALID

**Related Subroutines**

**GPQRVE**
Inquire Requested View Table Entries Input

**GPVOP**
Set View Output Priority

**GPVP** Set View Priority

**RCP code**

201330690 (X'0C001002')

## GPVOP - Set View Output Priority

**GPVOP (*wsid*, *view*, *refview*, *flag*)**

**Purpose**

Use **GPVOP** to modify the output priority of the given view in relation to another view on the specified workstation.

The application draws the contents of the lower priority view before the contents of a higher priority view. The highest priority view defaults to a view of zero. This subroutine only changes the output priority of a view. This can produce visual changes on the display.

When the application updates the workstation, the graPHIGS API sets the current view output priority for the specified view to the requested values.

**Note:** The Set View Priority (**GPVP**) subroutine is treated as a simultaneous invocation of the Set View Input Priority (**GPVIP**) and the Set View Output Priority (**GPVOP**) subroutines.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*refview* **— specified by user, fullword integer**
Reference view index (>=0).

*flag* **— specified by user, fullword integer**
Relative priority of the specified view with respect to the reference view (`1=HIGHER, 2=LOWER`).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**59** VIEW INDEX VALUE < ZERO

**155** VIEW PRIORITY REFERENCE NUMBER IS INVALID

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**333** RELATIVE VIEW PRIORITY VALUE IS INVALID

**Related Subroutines**

**GPQRVO**
Inquire Requested View Table Entries Output

**GPVIP**
Set View Input Priority

**GPVP** Set View Priority

**RCP code**

201330691 (X'0C001003')

## GPVP - Set View Priority

**GPVP (*wsid*, *view*, *refview*, *flag*)**

**Purpose**

Use **GPVP** to modify both the relative input and output priority of the given view in relation to another view on the specified workstation.

This subroutine sets the requested priority reference number and the requested relative priority to the specified values. When the application updates the workstation, the graPHIGS API sets the current priority reference number and current relative priority to the requested values.

This subroutine changes the output as well as the input priority of the specified view. This can produce visual changes on the display.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*refview* **— specified by user, fullword integer**
Priority reference view index (>=0).

*flag* **— specified by user, fullword integer**
Relative priority of specified view with respect to the reference view (`1=HIGHER, 2=LOWER`).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**155** VIEW PRIORITY REFERENCE NUMBER IS INVALID

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**333** RELATIVE VIEW PRIORITY VALUE IS INVALID

**Related Subroutines**

**GPQRVE**
Inquire Requested View Table Entries Input

**GPQRVO**
Inquire Requested View Table Entries Output

**GPVIP**
Set View Input Priority

**GPVOP**
Set View Output Priority

**RCP code**

201330689 (X'0C001001')

## GPXCR - Set Extended Color Representation

**GPXCR (*wsid*, *ctid*, *start*, *number*, *color*)**

**Purpose**

Use **GPXCR** to set the specified color values starting at the specified workstation's color table entry. The color values are interpreted according to the workstation's color model.

Only modifiable color tables may be changed. Use the Inquire Extended Color Representation (**GPQXCR**) subroutine to determine the characteristics of the workstation's color table.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ctid* **— specified by user, fullword integer**
Color table identifier of a modifiable color table existing on the workstation (-1=DISPLAY_COLOR_TABLE, 0=RENDERING_COLOR_TABLE, > 0=IMAGE COLOR TABLES).

*start* **— specified by user, fullword integer**
Start color index (>=0).

*number* **— specified by user, fullword integer**
Number of entries to be set (>=1).

*color* **— specified by user, array of short floating-point numbers**
The color table values to be set in the specified color table (0.0<=component<=1.0).

The array is assumed to be in row order such as RED1, GREEN1, BLUE1, RED2, GREEN2, BLUE2, etc....

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**49** COLOR INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**92** COLOR INDEX < ZERO

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**284** COLOR TABLE IDENTIFIER DOES NOT EXIST

**289** SPECIFIED COLOR TABLE CANNOT BE MODIFIED

**517** NUMBER OF INDEXES < ONE

**Related Subroutines**

**GPCML**
Set Color Model

**GPQCHH**
Inquire Color Table Characteristics

**GPQCF**
Inquire Color Facilities

**GPQCID**
Inquire List of Color Table Identifiers

**GPQXCF**
Inquire Extended Color Facilities

**GPQXCR**
Inquire Extended Color Representation

**RCP code**

201329417 (X'0C000B09')

## GPXER - Set Extended Edge Representation

**GPXER (*wsid*, *index*, *id*, *value*)**

**Purpose**

Use **GPXER** to set one field of the specified entry in the workstation's edge bundle table. These attribute values are applied during traversal when the appropriate ASF is set to 1=BUNDLED and the current edge index is set to the specified entry. These attribute values can also be set individually.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Edge bundle table index (>=1).

*id* **— specified by user, fullword integer**
Edge group identifier (1=OFF, 2=ON, 3=GEOMETRY_ONLY).

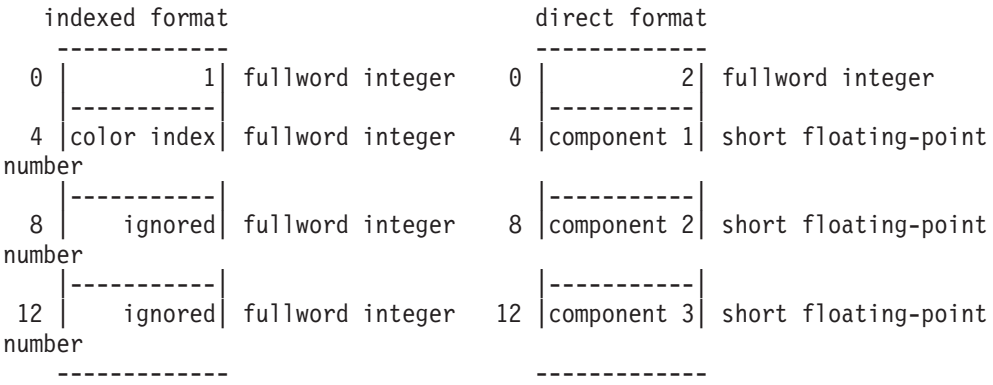
*value* **— specified by user, variable data**
The value that may be set for each field is expressed in the data format listed below:

**Group Identifier 1 - Edge flag**
A fullword integer (1=OFF, 2=ON, 3=GEOMETRY_ONLY).

**Group Identifier 2 - Edge line type**
A fullword integer. Specifies an index into the workstation's edge line type table. The table size and specified entries supported are workstation dependent. Use the Inquire Edge Facilities (**GPQEF**) subroutine to determine the supported edge line types on your workstation. The default edge line type table for supported entries is defined with the following line types: (1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE) (>=1 Any entry may be changed by GPLTR except entry 1).

**Group Identifier 3 - Edge linewidth scale factor**
A short floating-point number.

**Group Identifier 4 - Edge color**
Four fullwords of data with either of the following two formats:




**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**63** LINETYPE VALUE < ONE

**64** SPECIFIED LINETYPE NOT AVAILABLE ON WORKSTATION

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**97** COLOR FORMAT PARAMETER IS INVALID

**272** GROUP IDENTIFIER IS INVALID

**311** EDGE FLAG VALUE IS INVALID

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPCML**
Set Color Model

**GPECD**
Set Edge Color Direct

**GPECI**
Set Edge Color Index

**GPEI** Set Edge Index

**GPELT**
Set Edge Linetype

**GPESC**
Set Edge Scale Factor

**GPQEF**
Inquire Edge Facilities

**GPQXER**
Inquire Extended Edge Representation

**RCP code**

201345541 (X'0C004A05')

## GPXIR - Set Extended Interior Representation

**GPXIR (*wsid*, *index*, *id*, *value*)**

**Purpose**

Use **GPXIR** to set one field of the specified entry in the workstation's interior bundle table. These attribute values are applied during traversal when the appropriate ASF is set to `1=BUNDLED` and the current interior index is set to the specified entry. These attribute values can also be set individually.

The `1=HOLLOW` and `5=EMPTY` interior styles display nothing for the interior. If the edge flag is set to `1=OFF` and the interior style is `5=EMPTY`, then the graPHIGS API generates no visual output. The interior is detectable when the graPHIGS API encounters a primitive with with an interior style of `5=EMPTY` and the primitive is eligible for picking as determined by its visibility and detectability.

If the edge flag is `1=OFF` and the interior style is `1=HOLLOW`, then the graPHIGS API draws the boundary (solid line) When the graPHIGS API encounters a primitive with an interior style `1=HOLLOW` only the boundary of the primitive is eligible for picking as determined by its visibility and detectability.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Interior bundle table index (>=1).

*id* **— specified by user, fullword integer**
Interior group identifier (1=INTERIOR_STYLE, 2=INTERIOR_STYLE_INDEX, 3=INTERIOR_COLOR).

*value* **— specified by user, variable data**
The value that may be set for each field is expressed in the data format listed below:

**Group Identifier 1 - Interior style**
A fullword integer (1=HOLLOW, 2=SOLID, 3=PATTERN, 4=HATCH, 5=EMPTY).

**Group Identifier 2 - Interior style index**
A fullword integer (>=1) which is an index into the pattern or hatch table depending on the current interior style.

**Group Identifier 3 - Interior color**
Four fullwords of data with either of the following two formats:

```
   indexed format                       direct format
    ------------                         ------------
  0 |          1| fullword integer    0 |          2| fullword integer
    |-----------|                        |-----------|
  4 |color index| fullword integer    4 |component 1| short floating-point
number
    |-----------|                        |-----------|
  8 |    ignored| fullword integer    8 |component 2| short floating-point
number
    |-----------|                        |-----------|
 12 |    ignored| fullword integer   12 |component 3| short floating-point
number
    ------------                         ------------
```

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**83** INTERIOR STYLE NOT AVAILABLE ON WORKSTATION

**84** INTERIOR STYLE INDEX VALUE < ONE

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**97** COLOR FORMAT PARAMETER IS INVALID

**272** GROUP IDENTIFIER IS INVALID

**310** INTERIOR STYLE VALUE IS INVALID

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPCML**
Set Color Model

**GPICD**
Set Interior Color Direct

**GPICI** Set Interior Color Index

**GPII** Set Interior Index

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPQXIR**
Inquire Extended Interior Representation

**RCP code**

201345504 (X'0C004A04')

## GPXPLR - Set Extended Polyline Representation

**GPXPLR (*wsid*, *index*, *id*, *value*)**

**Purpose**

Use **GPXPLR** to set one field in the specified entry of the workstation's polyline bundle table.

These attribute values are applied during traversal when the appropriate ASF is set to `1=BUNDLED` and the current polyline index is set to the specified entry. These attribute values can also be set individually.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Polyline bundle table index (>=1).

*id* **— specified by user, fullword integer**
Polyline group identifier (`1=LINETYPE, 2=LINEWIDTH_SCALE_FACTOR, 3=POLYLINE_COLOR`).

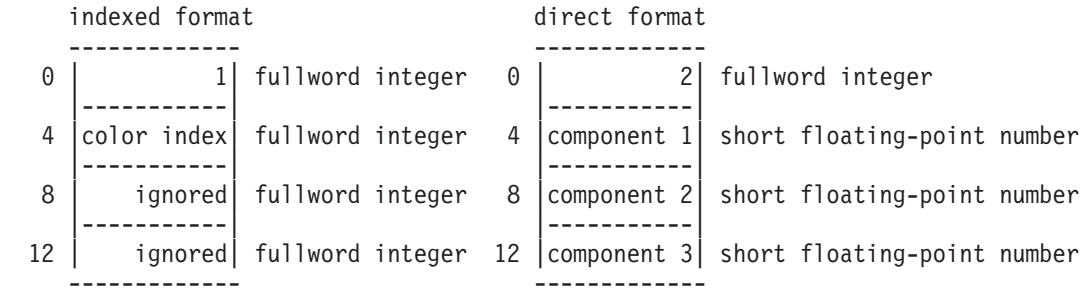
*value* **— specified by user, variable data**
The value that may be set for each field is expressed in the data format listed below:

**Group Identifier 1 - Line type**
A fullword integer (>=1). Specifies an index into the workstation's line type table. The table size and specific entries supported is workstation dependent. Use the Inquire Polyline Facilities (**GPQPLF**) subroutine to determine the supported line types on your workstation. The default line type table for supported entries is defined with the following line types: (`1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE`). Any entry may be changed by GPLTR except entry 1).

**Group Identifier 2 - Linewidth scale factor**
A short floating point number.

**Group Identifier 3 - Polyline color**

Four fullwords of data with either of the following two formats:




**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**63** LINETYPE VALUE < ONE

**64** SPECIFIED LINETYPE NOT AVAILABLE ON WORKSTATION

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**97** COLOR FORMAT PARAMETER IS INVALID

**272** GROUP IDENTIFIER IS INVALID

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPCML**
Set Color Model

**GPLT** Set Linetype

**GPLWSC**
Set Linewidth Scale Factor

**GPPLCD**
Set Polyline Color Direct

**GPPLCI**
Set Polyline Color Index

**GPPLI** Set Polyline Index

**GPQXLR**
Inquire Extended Polyline Representation

**RCP code**

201345537 (X'0C004A01')

# GPXPMR - Set Extended Polymarker Representation

**GPXPMR (*wsid*, *index*, *id*, *value*)**

### Purpose

Use **GPXPMR** to set one field of the specified entry in the workstation's polymarker bundle table.

These attribute values are applied during traversal when the appropriate ASF is set to 1=BUNDLED and the current polymarker index is set to the specified entry. These attribute values can also be set individually.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Polymarker bundle table index (>=1).

*id* **— specified by user, fullword integer**
Polymarker group identifier (1=MARKER_TYPE, 2=MARKER_SIZE_SCALE_FACTOR, 3=POLYMARKER_COLOR).

*value* **— specified by user, variable data**
The value that may be set for each field is expressed in the data format listed below:

**Group Identifier 1 - Marker type table index**
A fullword integer (>=1). Specifies an index into the marker type table of the workstation. The table size and specific entries supported are workstation dependent. Use the Inquire Polymarker Facilities (**GPQPMF**) subroutine to determine the supported marker types on your workstation. The default marker type table for supported entries is defined with the following marker types: (1=DOT, 2=PLUS_SIGN, 3=ASTERISK, 4=CIRCLE, 5=DIAGONAL_CROSS, 6-n=ASTERISK. Any entry may be changed by **GPMTR** except entry 3).

**Group Identifier 2 - Marker size scale factor**
A short floating-point number.

**Group Identifier 3 - Polymarker color**
Four fullwords of data with either of the following two formats:

```
   indexed format                    direct format
   --------------                    --------------
 0 |           1| fullword integer  0 |           2| fullword integer
   |------------|                     |------------|
 4 |color index | fullword integer  4 |component 1 | short floating-point number
   |------------|                     |------------|
 8 |     ignored| fullword integer  8 |component 2 | short floating-point number
   |------------|                     |------------|
12 |     ignored| fullword integer 12 |component 3 | short floating-point number
   --------------                    --------------
```

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**69** MARKER TYPE VALUE < ONE

**70** SPECIFIED MARKER TYPE NOT AVAILABLE ON WORKSTATION

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**97** COLOR FORMAT PARAMETER IS INVALID

**272** GROUP IDENTIFIER IS INVALID

### Related Subroutines

**GPASF**
Attribute Source Flag Setting

**GPCML**
Set Color Model

**GPPMCD**
Set Polymarker Color Direct

**GPPMCI**
Set Polymarker Color Index

**GPPMI**
Set Polymarker Index

**GPQXMR**
Inquire Extended Polymarker Representation

### RCP code

201345538 (X'0C004A02')

## GPXTXR - Set Extended Text Representation

**GPXTXR (*wsid*, *index*, *id*, *value*)**

### Purpose

Use **GPXTXR** to set one field of the specified entry in the workstation's text bundle table.

These attribute values are applied during traversal when the appropriate ASF is set to `1=BUNDLED` and the current text index is set to the specified entry. These attribute values can also be set individually.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Text index (>=1).

*id* **— specified by user, fullword integer**
Text group identifier (`1=TEXT_FONT, 2=TEXT_PRECISION, 3=CHARACTER_EXPANSION_FACTOR, 4=CHARACTER_SPACING, 5=TEXT_COLOR`).

*value* **— specified by user, variable data**
Value to be set in the field of the specified group identifier expressed in the data format listed below:

**Group Identifier 1 - Text font**
A fullword integer (1-255).

**Group Identifier 2 - Text precision**
A fullword integer (1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC).

**Group Identifier 3 - Character expansion factor**
A short floating-point value (>=0).

**Group Identifier 4 - Character spacing**
A short floating-point value.

**Group Identifier 5 - Text color**
Four fullwords of data with either of the following two formats:

```
   indexed format                    direct format
   --------------                    -------------
 0 |           1| fullword integer  0 |           2| fullword integer
   |------------|                     |------------|
 4 |color index | fullword integer  4 |component 1 | short floating-point number
   |------------|                     |------------|
 8 |    ignored | fullword integer  8 |component 2 | short floating-point number
   |------------|                     |------------|
12 |    ignored | fullword integer 12 |component 3 | short floating-point number
   --------------                    -------------
```

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**75** TEXT FONT VALUE IS INVALID

**77** CHARACTER EXPANSION FACTOR <= ZERO

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**97** COLOR FORMAT PARAMETER IS INVALID

**272** GROUP IDENTIFIER IS INVALID

**305** TEXT PRECISION VALUE IS INVALID

**Related Subroutines**

**GPASF**
Attribute Source Flag Setting

**GPCHSP**
Set Character Spacing

**GPCHXP**
Set Character Expansion Factor

**GPCML**
Set Color Model

**GPTXCD**
Set Text Color Direct

**GPTXCI**
Set Text Color Index

**GPTXFO**
Set Text Font

**GPTXI** Set Text Index

**GPTXPR**
Set Text Precision

**GPQXTR**
Inquire Extended Text Representation

**RCP code**

201345539 (X'0C004A03')

## GPXVCH - Set Extended View Characteristics

**GPXVCH (*wsid*, *view*, *number*, *charids*, *values*)**

**Purpose**

Use **GPXVCH** to set one or more characteristics of the specified view. Characteristics which may be set with this subroutine include: clipping indicators, appearance of the viewport, a value indicating if the view is displayed, and a value indicating if the display view option changes frequently.

The values specified are stored in the requested view table entries. When the application updates the workstation, the graPHIGS API sets the corresponding current values to the requested values.

The clipping indicators determine to which boundaries the contents of the view are clipped. The shielding indicator determines if the content of lower priority views may be displayed within the boundaries of the specified viewport. The border indicator specifies if a border is to be drawn around the viewport. The view active flag determines if the view and its contents are displayed.

When set to `2=ON`, the temporary view indicator indicates that the corresponding view will have its view activity changed frequently. Where possible the device support saves the current underlying screen image. This allows deactivation to consist of restoring the underlying screen image without requiring structure traversal. A typical use of the temporary view capability is the "pop-up menu" found in many applications today.

Application developers should be aware of the following points:

- The temporary view indicator acts as an indicator to the graPHIGS API that the corresponding view is active only for a short period of time. If used properly, it can greatly contribute to the interactive performance of viewport deactivation.
- The best efficiency is gained when a temporary view is the highest priority active viewport.

The workstation's view table is 0 based, however, view entry 0 cannot be changed. (For the default values for the view entry 0, see *The graPHIGS Programming Interface: Technical Reference*).

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*view* — **specified by user, fullword integer**
View index (>=1). Index of view table entry to be altered.

*number* — **specified by user, fullword integer**
Number of characteristic identifiers in the *charids* list (>=1).

*charids* — **specified by user, array of fullword integers**
List of characteristic identifiers. Each integer identifies a view characteristic to be set.

*values* — **specified by user, array of fullword integers**
List of characteristic values. Each list entry specifies the value to be applied to the corresponding characteristic identifier listed in the *charids* list.

Valid view characteristic identifiers and values are:

- 1 = View window clipping indicator (`1=NOCLIP, 2=CLIP`)
- 2 = Near clipping indicator (`1=NOCLIP, 2=CLIP`)
- 3 = Far clipping indicator (`1=NOCLIP, 2=CLIP`)
- 4 = View shielding indicator (`1=OFF, 2=ON`)
- 6 = View border indicator (`1=OFF, 2=ON`)
- 8 = View active flag (`1=INACTIVE, 2=ACTIVE`)
- 9 = Temporary view indicator (`1=OFF, 2=ON`)

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**332** CLIP INDICATOR VALUE IS INVALID

**334** TEMPORARY VIEW INDICATOR IS INVALID

**507** SHIELDING INDICATOR VALUE IS INVALID

**508** VIEW ACTIVE FLAG VALUE IS INVALID

**518** VIEW ZERO CANNOT BE MODIFIED

**547** VIEW BORDER INDICATOR IS INVALID

**591** NUMBER OF CHARACTERISTICS IDENTIFIERS IS < ONE

**592** VIEW CHARACTERISTICS IDENTIFIER IS INVALID

**Related Subroutines**

**GPQCVR**
Inquire Current View Representation

**GPQRVR**
Inquire Requested View Representation

**GPXVR**
Set Extended View Representation

**RCP code**

201330434 (X'0C000F02')

# GPXVR - Set Extended View Representation

**GPXVR (*wsid*, *view*, *id*, *value*)**

### Purpose

Use **GPXVR** to set one field in the entry of the specified workstation's view table.

The values specified are stored in the requested view table entry. When the application updates the workstation, the graPHIGS API sets the corresponding current values in the view table entry to the requested values. You can set only one field at a time with each subroutine call. Issue multiple invocations of **GPXVR** to set multiple fields.

The clipping indicators determine to which boundaries the contents of the view are clipped. The shielding indicator determines if the content of lower priority views may be displayed within the boundaries of the specified viewport. The border indicator specifies if a border is to be drawn around the viewport.

When set to `2=ON`, the temporary view indicator indicates that the corresponding view will have its view activity changed frequently. Where possible the device support saves the current underlying screen image. This allows deactivation consist of restoring the underlying screen image without requiring structure traversal. A typical use of the temporary view capability is the "pop-up menu" found in many applications today.

Application developers should be aware of the following points:

- The temporary view indicator acts as an indicator to the graPHIGS API that the corresponding view is active only for a short period of time. If used properly, it can greatly contribute to the interactive performance of viewport deactivation.
- The best efficiency is gained when a temporary view is the highest priority active viewport.

When setting a view's projection type to perspective, any image mappings currently defined to the view are canceled.

The view active flag for output determines if the view and its contents are displayed. The view active flag for input considers the view for transformation of locator and stroke input from Device Coordinate (DC) to World Coordinates (WC).

The antialiasing mode defines the algorithm that the graPHIGS API uses when rendering primitives within the specified view. It improves the quality of the image and reduces the jagged appearance of the objects. The highest quality image results when you set the antialiasing mode to `2=SUBPIXEL_ON_THE_FLY`. However, for better performance, set the antialiasing mode to `3=NON_SUBPIXEL_ON_THE_FLY` Issuing the Set Antialiasing Mode (**GPAID**) structure element specifies whether the antialiasing algorithm is to be applied to the primitives within the view.

The shield alpha value defines the initial alpha value for the view shield. The value is used to initialize the destination alpha values when alpha planes are present and shielding indicator is on. The initial shield alpha value does not affect the initial shield color, but is used to blend subsequent primitives with the view shield when blending is in effect.

Other viewing parameters can also be controlled, such as Hidden Line/Hidden Surface Removal (HLHSR) mode, transparency processing, initial color processing, and initial frame buffer mask. Color processing mode and frame buffer protect mask are traversal defaults and are applied to images as well as to structure content.

The workstation's view table is 0 based, however, view entry 0 cannot be changed. (See *The graPHIGS Programming Interface: Technical Reference* for the default values for view entry 0).

**Note:** The following functions are treated as other forms of this generic subroutine:

- GPVCH - Set View Characteristics
- GPXVCH - Set Extended View Characteristics
- GPVMT2 - Set View Matrix 2
- GPVMT3 - Set View Matrix 3
- GPVMP2 - Set View Mapping 2
- GPVMP3 - Set View Mapping 3

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View table index (>=1).

*id* **— specified by user, fullword integer**
Group identifier (1<= *id*<=24). See group identifier values defined below.

*value* **— specified by user, array of fullword quantities**
The value to be set for each group is expressed in the following data formats:

**Group Identifier 1 - Window clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP).

**Group Identifier 2 - Near clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP).

**Group Identifier 3 - Far clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP).

**Group Identifier 4 - Shielding indicator**
A fullword integer (1=OFF, 2=ON).

**Group Identifier 5 - Shielding color**
Four fullwords of data with either of the following two formats:

| | indexed format | | | | direct format | |
|---|---|---|---|---|---|---|
| 0 | 1 | fullword integer | | 0 | 2 | fullword integer |
| 4 | color index | fullword integer | | 4 | component 1 | short floating-point number |
| 8 | ignored | fullword integer | | 8 | component 2 | short floating-point number |
| 12 | ignored | fullword integer | | 12 | component 3 | short floating-point number |

**Group Identifier 6 - Border indicator**
A fullword integer (1=OFF, 2=ON).

**Group Identifier 7 - Border color**
Four fullwords of data with either of the following two formats:

| | indexed format | | | | direct format | |
|---|---|---|---|---|---|---|
| 0 | 1 | fullword integer | | 0 | 2 | fullword integer |
| 4 | color index | fullword integer | | 4 | component 1 | short floating-point number |

```
 8 |    ignored| fullword integer  8 |component 2| short floating-point number
   |-----------|                     |-----------|
12 |    ignored| fullword integer 12 |component 3| short floating-point number
   -------------                     -------------
```

**Group Identifier 8 - Reserved**
This field is reserved.

**Group Identifier 9 - Temporary view indicator**
A fullword integer (1=OFF, 2=ON).

**Group Identifier 10 - HLHSR mode**
A fullword integer (1=OFF, 2=ON_THE_FLY).

**Group Identifier 11 - Transparency processing mode**
A fullword integer (1=OFF, 2=PARTIAL_TRANSPARENT, 3=BLEND, 4=BLEND_ALL).

**Group Identifier 12 - Initial color processing mode index**
A fullword integer (>=0).

**Note:** This color processing index is not applied to shielding or border color.

**Group Identifier 13 - Initial frame buffer write protect mask**
A 32-bit bit string.

**Note:** The frame buffer mask is *not* applied to shielding or border color.

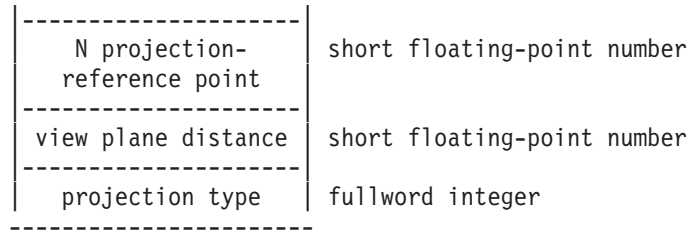
**Group Identifier 14 - Viewport, 2D form**
4 short floating-point numbers (including only Xmin, Xmax, Ymin, Ymax). For the set subroutine call, Zmin and Zmax are set to their default values.

**Group Identifier 15 - Viewport, 3D form**
6 short floating-point numbers (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

**Group Identifier 16 - View volume, 2D form**
4 short floating-point numbers specifying the view window (Umin, Umax, Vmin, Vmax). For the set subroutine call, other fields of the view volume group are set to their default values.

**Group Identifier 17 - View volume, 3D form**
10 short floating-point numbers and a fullword integer specifying a view window (Umin,Umax,Vmin,Vmax), near plane distance, far plane distance, projection reference point (*u*, *v*, *n*), view plane distance and a projection type (1=PARALLEL, 2=PERSPECTIVE).

```
-----------------------
|        Umin         | short floating-point number
|---------------------|
|        Umax         | short floating-point number
|---------------------|
|        Vmin         | short floating-point number
|---------------------|
|        Vmax         | short floating-point number
|---------------------|
| near plane distance | short floating-point number
|---------------------|
|  far plane distance | short floating-point number (=>near plane)
|---------------------|
|    U projection-    | short floating-point number
|   reference point   |
|---------------------|
|    V projection-    | short floating-point number
|   reference point   |
```




**Group Identifier 18 - View matrix, 2D form**
9 short floating-point numbers. For the input view matrix, the elements are in the following order:

```
|m11  m12  m13|
|m21  m22  m23|-->(m11,m12,m13,m21.....m33)
|m31  m32  m33|
```

The 3[default]3 matrix is expanded by the graPHIGS API into a 4[default]4 matrix as follows:

```
|a  b  c|        |a b 0 c|
|d  e  f| -----> |d e 0 f|
|g  h  i|        |0 0 1 0|
                 |g h 0 i|
```

When inquired, the matrix returned is the expanded 4[default]4 matrix. For the set subroutine call, other elements are set to their default values.

**Group Identifier 19 - View matrix, 3D form**
16 short floating-point numbers. For the input view matrix, the elements must be in the following order:

```
| m11 m12 m13 m14 |
| m21 m22 m23 m24 |---> (m11,m12,m13,m14,m21,m22,.....m44)
| m31 m32 m33 m34 |
| m41 m42 m43 m44 |
```

**Group Identifier 20 - View input active flag**
A fullword integer specifying the view that is to be made active/inactive for input (1=INACTIVE, 2=ACTIVE)

**Group Identifier 21 - View output active flag**
A fullword integer specifying the view that is to be made active/inactive for output (1=INACTIVE, 2=ACTIVE).

**Group Identifier 22 - View mapping matrix, 2D form**
9 short floating-point numbers including:

```
|m11  m12  m13|
|m21  m22  m23| --->(m11,m12,m13,m21.....m33)
|m31  m32  m33|
```

**Group Identifier 23 - View mapping matrix, 3D form**
16 short floating-point numbers including:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| --->(m11,m12,m13,m14,m21,m22,.....m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

**Group Identifier 24 - Antialiasing mode**
A fullword integer (1=OFF, 2=SUBPIXEL_ON_THE_FLY, 3=NON_SUBPIXEL_ON_THE_FLY).

**Group Identifier 25 - Shield alpha value**
A fullword integer (0<=*alpha*<=255). The default value of the shield alpha value is 255 (opaque).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**44** INVALID WINDOW DEFINITION

**55** PRP IS POSITIONED ON THE VIEW PLANE

**59** VIEW INDEX VALUE < ZERO

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

**97** COLOR FORMAT PARAMETER IS INVALID

**251** SPECIFIED HLHSR MODE IS NOT SUPPORTED

**253** SPECIFIED ANTIALIASING MODE IS NOT SUPPORTED

**260** SPECIFIED TRANSPARENT PROCESSING MODE IS NOT SUPPORTED

**265** COLOR PROCESSING INDEX < ZERO

**266** COLOR PROCESSING INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY

**272** GROUP IDENTIFIER IS INVALID

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**330** INVALID VIEWPORT

**331** PROJECTION TYPE IS INVALID

**332** CLIP INDICATOR VALUE IS INVALID

**334** TEMPORARY VIEW INDICATOR IS INVALID

**336** FAR CLIPPING PLANE IN FRONT OF NEAR CLIPPING PLANE

**507** SHIELDING INDICATOR VALUE IS INVALID

**508** VIEW ACTIVE FLAG VALUE IS INVALID

**518** VIEW ZERO CANNOT BE MODIFIED

**522** VIEW MATRIX IS SINGULAR

**547** VIEW BORDER INDICATOR IS INVALID

**639** SPECIFIED ALPHA VALUE IS INVALID

**Related Subroutines**

**GPAID**
Set Antialiasing Identifier

**GPCML**
Set Color Model

**GPCPI**
Set Color Processing Index

**GPEVM2**
Evaluate View Mapping Matrix 2

**GPEVM3**
Evaluate View Mapping Matrix 3

**GPFBM**
Set Frame Buffer Protect Mask

**GPQAMO**
Inquire Available Antialiasing Modes

**GPQCVR**
Inquire Current View Representation

**GPQHMO**
Inquire Available HLHSR Modes

**GPQRVR**
Inquire Requested View Representation

**GPQTMO**
Inquire Available Transparency Modes

**RCP code**

201330435 (X'0C000F03')

# Chapter 9. Display Subroutines

The data stored in a structure store or image board resource cannot be displayed until an affiliation is created that ties the data together with a view in a workstation. The subroutines in this chapter are used to create and delete these ties.

For structure store resources, this bond is created and deleted by associating and disassociating structure networks to a view. To associate a structure network to a view, the owning structure store resource must first be associated to the workstation (either by using the Associate Structure Store with Workstation [**GPASSW**] [page GPASSW - Associate Structure Store with Workstation] subroutine or by automatic structure store association by the Open Workstation [**GPOPWS**] [page GPOPWS - Open Workstation] subroutine).

For image board resources, the association is formed and dispelled by creating and deleting image mappings. This view association can only occur if the Define Image (**GPDFI**) subroutine has been used to define an image on a workstation.

In addition to the subroutines that perform the above ties, this chapter also defines those subroutines that associate and disassociate structure networks to a workstation (useful only to preload data into those workstations that require a special data format) and those subroutines that empty one or more views of a workstation.

## GPARV - Associate Root with View

**GPARV (*wsid*, *view*, *strid*, *prior*)**

### Purpose

Use **GPARV** to add a structure network to the traversal list for the specified view at the location defined by the priority parameter.

This subroutine implicitly performs an Associate Root with Workstation (**GPARW**) subroutine.

This subroutine is necessary because the association of a structure network to a view is required in order to actually display the data in the structures in the network.

The specified structure is searched for in the structure store associated with the workstation. If there is no structure store associated with the workstation, an error is generated. If the specified structure does not exist in the associated structure store, an empty structure is created. If the specified structure is already associated with the workstation, its location in the traversal list is modified according to the specified priority.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*view* — **specified by user, fullword integer**
View index (>=0).

*strid* — **specified by user, fullword integer**
Structure identifier.

*prior* — **specified by user, short floating-point number**
Priority (0.0<= *prior*<=1.0).

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **59** | VIEW INDEX VALUE < ZERO |
| **126** | PRIORITY VALUE IS INVALID |
| **224** | SPECIFIED VIEW DOES NOT HAVE ASSOCIATED STRUCTURE STORE |
| **323** | VIEW INDEX EXCEEDS VIEW TABLE CAPACITY |

### Related Subroutines

| | |
|---|---|
| **GPARW** | Associate Root with Workstation |
| **GPQAR** | Inquire Set of Associated Roots |
| **GPQNSP** | Inquire Number of Structure Priorities Supported |
| **GPQNV** | Inquire Number of Definable View Table Entries |
| **GPQRV** | Inquire Set of Roots in View |
| **GPQVR** | Inquire Set of Views Which Contain Root |

### RCP code

201334017 (X'0C001D01')

## GPARW - Associate Root with Workstation

**GPARW (*wsid*, *strid*)**

### Purpose

Use **GPARW** to convert a structure network within the currently selected structure store into the workstation specific format.

This subroutine has its meaning and effect only for the workstation that requires structures in a workstation specific format. For a workstation which can directly traverse the common format, this subroutine is ignored and has no effect.

If the selected structure is not associated with any view of the specified workstation, no action is performed. If the specified structure does not exist, an empty structure is created.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*strid* — **specified by user, fullword integer**
Structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |

### Related Subroutines

**RCP code**

201333505 (X'0C001B01')

## GPCIM2 - Create Image Mapping 2

**GPCIM2 (*wsid*, *imap*, *view*, *index*, *origin*, *size*, *P*, *Q*, *R*, *method*, *prior*)**

### Purpose

Use **GPCIM2** to define the mapping of an image to a specified view of a workstation.

A rectangle on the specified image is mapped into a parallelogram on the World Coordinates' (WC) *x*, *y* plane and is associated with the specified view. The parallelogram is identified by the specified image mapping identifier. If the specified image mapping identifier is already in use, the existing image mapping is deleted and a new image mapping is created. This subroutine only supports views in parallel projection. If the specified view has its projection type set to perspective, the image mapping is *not* defined and an error message is issued. If the projection type of a view is set to perspective and image mappings are currently defined to that view, the image mappings for the view are deleted before the projection type is set to perspective.

The priority parameter defines the relationship of this image mapping to others within the same view. Higher priority images will be displayed last.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*imap* — **specified by user, fullword integer**
Image mapping identifier.

*view* — **specified by user, fullword integer**
View index (>=0).

*index* — **specified by user, fullword integer**
Defined image index (>=1).

*origin* — **specified by user, two fullword integers**
Origin of the image rectangle (*x*, *y*).

*size* — **specified by user, two fullword integers**
Size of the image rectangle (*sx*, *sy*).

*P* — **specified by user, 2 short floating-point numbers (WC)**
Lower left corner of the mapped rectangle in WC.

*Q* — **specified by user, 2 short floating-point numbers (WC)**
Lower right corner of the mapped rectangle in WC.

*R* — **specified by user, 2 short floating-point numbers (WC)**
Top left corner of the mapped rectangle in WC.

*method* — **specified by user, fullword integer**
Image mapping method (`1=PIXEL_BY_PIXEL`).

*prior* **— specified by user, short floating-point number**
Priority (0.0<= *prior*<=1.0).

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **59** | VIEW INDEX VALUE < ZERO |
| **126** | PRIORITY VALUE IS INVALID |
| **236** | RECTANGLE DEFINITION IS INVALID |
| **288** | IMAGE INDEX NOT WITHIN WORKSTATION TABLE RANGE |
| **290** | SPECIFIED IMAGE INDEX IS NOT DEFINED |
| **294** | SPECIFIED IMAGE MAPPING METHOD IS NOT SUPPORTED |
| **323** | VIEW INDEX EXCEEDS VIEW TABLE CAPACITY |
| **331** | PROJECTION TYPE IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPDIM** | Delete Image Mapping |
| **GPEAV** | Empty All Views |
| **GPEV** | Empty View |
| **GPQIMC** | Inquire Image Mapping Characteristics |
| **GPQIMF** | Inquire Image Mapping Facilities |
| **GPQIMI** | Inquire Image Mapping of Image |
| **GPQIMV** | Inquire Image Mapping on View |
| **GPQIMW** | Inquire Image Mapping on Workstation |

### RCP code

201346307 (X'0C004D03')

## GPCIM3 - Create Image Mapping 3

**GPCIM3 (*wsid*, *imap*, *view*, *index*, *origin*, *size*, *P*, *Q*, *R*, *method*, *prior*)**

### Purpose

Use **GPCIM3** to define the image mapping of an image to a specified view of a workstation.

A rectangle on the specified image is mapped into a parallelogram in World Coordinates (WC) and is associated with the specified view. The parallelogram is identified by the specified image mapping identifier. If the specified image mapping identifier is already in use, the existing image mapping is deleted and a new image mapping is created.

This subroutine only supports views in parallel projection. If the specified view has its projection type set to perspective, the image mapping is *not* defined and an error message is issued. If the projection type of a view is set to perspective and image mappings are currently defined to that view, the image mappings for the view are deleted before the projection type is set to perspective.

The priority parameter defines the relationship of this image mapping to others within the same view. Higher priority images will be displayed last.

### Parameters

| | |
|---|---|
| *wsid* — specified by user, fullword integer | Workstation identifier. |
| *imap* — specified by user, fullword integer | Image mapping identifier. |
| *view* — specified by user, fullword integer | View index (>=0). |
| *index* — specified by user, fullword integer | Defined image index (>=1). |
| *origin* — specified by user, two fullword integers | Origin of the image rectangle (*x*, *y*). |
| *size* — specified by user, two fullword integers | Size of the image rectangle (*sx*, *sy*). |
| *P* — specified by user, 3 short floating-point numbers (WC) | Lower left corner of the mapped rectangle in WC. |
| *Q* — specified by user, 3 short floating-point numbers (WC) | Lower right corner of the mapped rectangle in WC. |
| *R* — specified by user, 3 short floating-point numbers (WC) | Top left corner of the mapped rectangle in WC. |
| *method* — specified by user, fullword integer | Image mapping method (`1=PIXEL_BY_PIXEL`). |
| *prior* — specified by user, short floating-point number | Priority (0.0<= *prior*<=1.0). |

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **59** | VIEW INDEX VALUE < ZERO |
| **126** | PRIORITY VALUE IS INVALID |
| **236** | RECTANGLE DEFINITION IS INVALID |
| **288** | IMAGE INDEX NOT WITHIN WORKSTATION TABLE RANGE |
| **290** | SPECIFIED IMAGE INDEX IS NOT DEFINED |
| **294** | SPECIFIED IMAGE MAPPING METHOD IS NOT SUPPORTED |
| **323** | VIEW INDEX EXCEEDS VIEW TABLE CAPACITY |
| **331** | PROJECTION TYPE IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPDIM** | Delete Image Mapping |
| **GPEAV** | Empty All Views |
| **GPEV** | Empty View |
| **GPQIMC** | Inquire Image Mapping Characteristics |
| **GPQIMF** | Inquire Image Mapping Facilities |
| **GPQIMI** | Inquire Image Mapping of Image |
| **GPQIMV** | Inquire Image Mapping on View |
| **GPQIMW** | Inquire Image Mapping on Workstation |

### RCP code

201346308 (X'0C004D04')

## GPDARW - Disassociate All Roots from Workstation

**GPDARW (*wsid*)**

### Purpose

Use **GPDARW** to remove all structure networks in the currently selected structure store from the workstation.

This subroutine is equivalent to invoking the Disassociate Root from Workstation (**GPDRW**)subroutine for each structure in the currently selected structure store.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |

### Related Subroutines

**GPDRW** Disassociate Root from Workstation

### RCP code

201334529 (X'0C001F01')

## GPDIM - Delete Image Mapping

**GPDIM (*wsid*, *imap*)**

### Purpose

Use **GPDIM** to delete the specified image mapping definition from the specified workstation.

If the specified image mapping does not exist, no action is performed.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*imap* — **specified by user, fullword integer**
Image mapping identifier.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |

### Related Subroutines

| | |
|---|---|
| **GPCIM2** | Create Image Mapping 2 |
| **GPCIM3** | Create Image Mapping 3 |
| **GPQIMI** | Inquire Image Mapping of Image |
| **GPQIMV** | Inquire Image Mapping on View |

### RCP code

201346309 (X'0C004D05')

## GPDRAV - Disassociate Root from All Views

**GPDRAV (*wsid*, *strid*)**

### Purpose

Use **GPDRAV** to remove a structure network within the currently selected structure store from traversal lists of all views of the specified workstation.

If the currently selected structure store is not associated with the specified workstation, the specified structure does not exist or it is not associated with any view, no action is taken.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*strid* — **specified by user, fullword integer**
Structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |

### Related Subroutines

None

### RCP code

201334274 (X'0C001E02')

## GPDRV - Disassociate Root from View

**GPDRV (*wsid*, *view*, *strid*)**

### Purpose

Use **GPDRV** to remove a structure network from the traversal list of the specified workstation view.

If there is no structure store associated with the workstation or if the specified structure is not a root of the view, this subroutine is ignored.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*view* — **specified by user, fullword integer**
View index (>=0).

*strid* — **specified by user, fullword integer**
Structure identifier.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **59** | VIEW INDEX VALUE < ZERO |
| **323** | VIEW INDEX EXCEEDS VIEW TABLE CAPACITY |

### Related Subroutines

None

### RCP code

201334273 (X'0C001E01')

## GPDRW - Disassociate Root from Workstation

**GPDRW (*wsid*, *strid*)**

### Purpose

Use **GPDRW** to remove a structure network from the workstation.

For a workstation which does not require structures in the workstation specific format, this subroutine is equivalent to the Disassociate Root from All Views (**GPDRAV**) subroutine. For a workstation which requires structures in the workstation specific format, this subroutine also deletes structures in the workstation dependent format from the workstation storage.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*strid* — **specified by user, fullword integer**
Structure identifier.

### Error Codes

| | |
|---|---|
| **12** | FUNCTION REQUIRES STATE SSSL |
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |

### Related Subroutines

| | |
|---|---|
| **GPDRAV** | Disassociate Root from All Views |

### RCP code

201333761 (X'0C001C01')

## GPEAV - Empty All Views

**GPEAV (*wsid*)**

### Purpose

Use **GPEAV** to remove structure networks and image mappings that exist in all views from the specified workstation.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |

### Related Subroutines

None

### RCP code

201334276 (X'0C001E04')

## GPEV - Empty View

**GPEV (*wsid*, *view*)**

### Purpose

Use **GPEV** to remove structure networks and image mappings that exist in the specified view.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*view* — **specified by user, fullword integer**
View index (>=0).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

### Related Subroutines

None

### RCP code

201334275 (X'0C001E03')

# Chapter 10. Transformation Subroutines

The transformation subroutines found in this section fall into three general categories: modeling clipping, modeling transformations and workstation transformations.

**Modeling Clipping**

The four modeling clipping subroutines are:

| | |
|---|---|
| **GPMCI** | Set Modeling Clipping Indicator |
| **GPMCV2** | Set Modeling Clipping Volume 2 |
| **GPMCV3** | Set Modeling Clipping Volume 3 |
| **GPRMCV** | Restore Modeling Clipping Volume |

The modeling clipping subroutines create modeling clipping structure elements, which modify the current modeling clipping values that the graPHIGS API applies to primitives during traversal.

**Modeling Transformations**

The four modeling transformation subroutines are:

| | |
|---|---|
| **GPGLX2** | Set Global Transformation 2 |
| **GPGLX3** | Set Global Transformation 3 |
| **GPMLX2** | Set Modeling Transformation 2 |
| **GPMLX3** | Set Modeling Transformation 3 |

The modeling transformation subroutines create transformation structure elements, which modify the current transformation values that the graPHIGS API applies to primitives during traversal.

**Workstation Transformations**

The three workstation transformation subroutines are:

| | |
|---|---|
| **GPDCMM** | Set Device Coordinate Mapping Method |
| **GPWSX2** | Set Workstation Transformation 2 |
| **GPWSX3** | Set Workstation Transformation 3 |

The workstation transformation subroutines **GPWSX2** and **GPWSX3** allow the application to modify the mapping of Normalized Projection Coordinates (NPC) into Device Coordinates (DC) for a specified workstation. **GPDCMM** controls how an image is displayed on a specified workstation.

> **Note:** When the graPHIGS API inserts a structure element into an open structure following the element pointer, the pointer moves to the new element.

## GPBDMF - Set Back Data Morphing Factors

**GPBDMF (*flength, fdata*)**

**Purpose**

Use **GPBDMF** to insert a Set Back Data Morphing Factors structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Back Data Morphing Factors structure element, depending on the current edit mode.

During traversal, the graPHIGS API combines the values of *fdata* with a primitive's data morphing vectors to modify its rendered data mapping data. If the face distinguish mode (**GPFDMO**) is `1=NONE`, then the graPHIGS API uses the current data morphing scale factors when performing data mapping on the back facing portions of area primitives. If the face distinguish mode is `2=COLOR_SURFACE_PROPERTIES`, then the graPHIGS API uses these values when performing data mapping on only back facing portions of area primitives.

In data morphing, the graPHIGS API combines the data mapping data values ($x_1$, $x_2$, ..., $x_{ndata}$ ) with the back data morphing scale factors in the *fdata* parameter ($s_1$, $s_{\#EMPTY>2}$, ..., $s_{nscale}$) and the data morphing vectors (($d_{1,1}$, $d_{1,2}$, ..., $d_{1,ndata}$), ($d_{\#EMPTY>2,1}$, $d_{2,2}$, ..., $d_{\#EMPTY>2,ndata}$), ..., ($d_{nvector,1}$, $d_{nvector,2}$, ..., $d_{\#EMPTY>nvector,ndata}$)) to obtain the new back data mapping data values ($x'_1$, $x'_2$, ..., $x'_{ndata}$). This combination is of the form:

$$x'_1 = s_1 x_1 + s_2 d_{1,1} + s_3 d_{2,1} +...+ s_{nscale} d_{nvector,1}$$

$$x'_2 = s_1 x_2 + s_2 d_{1,2} + s_3 d_{2,2} +...+ s_{nscale\ nvector,2}$$

...

$$x'_{ndata} = s_1 x_{ndata} + s_2 d_{1,\ ndata} + s_3 d_{2},ndata> +...+ s_{nscale} d_{nvector,ndata}$$

These equations show that the number of morphing scale factors should be one more than the number of morphing vectors in the affected primitive (*nscale*=*nvector*+1). However, if the number of morphing vectors and scale factors disagree at traversal time, then 0 value vectors and scale factors are assumed wherever necessary. For example, if you supply too many scale factors for a given primitive (*nscale*>*nvector*+1), then the graPHIGS API ignores the extra scale factors, as if there were additional 0 valued morphing vectors in the primitive definition. If you supply too few scale factors (*nscale*<*nvector*+1), then the graPHIGS API ignores the extra morphing vectors, just as if there were additional scale factors with value zero in this function call.

The traversal default for data morphing is *flength*=1 and *fdata*={1.0}.

Use **GPQWDT** to inquire the morphing facilities of a specified workstation.

### Parameters

*flength*— **specified by user, fullword integer (>=1)**
Number of morphing factors.

*fdata* — **specified by user, array of short floating-point numbers**
List of morphing factors. The number of entries in this list is given by the *flength* parameter.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **61** | LENGTH IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPDMF** | Set Data Morphing Factors |
| **GPDMR** | Set Data Mapping Representation |
| **GPQWDT** | Inquire Workstation Description |

**RCP code**

201343521 (X'0C004221')

# GPDCMM - Set Device Coordinate Mapping Method

**GPDCMM(*wsid*, *method*, *length*, *data*)**

## Purpose

Use**GPDCMM** to control how an image is displayed on a specified workstation. Only workstations which use the facilities of a window system (e.g., X-Windows) have use for this procedure. This subroutine specifies the method for transforming the graPHIGS API coordinate values to window coordinate values of a window system.

The graPHIGS API uses the *method* parameter to determine how the graPHIGS API performs the coordinate transformation. The valid methods include:

**1=MAPPED**
The graPHIGS API device coordinate range is scaled to fit the size of the window. The graPHIGS API preserves the aspect ratio of the workstation's device coordinate range. If the application makes the window smaller (or larger), then the size of the data of the window decreases (or increases) in direct proportion but the amount of data in the window remains the same. This is the default method.

**2=DIRECT**
The graPHIGS API displays the graPHIGS API device coordinate range directly into the window without scaling the contents. The graPHIGS API uses the lower, left corner of the window as the origin of the device coordinate range. If you make the window smaller than the device coordinate range, then the graPHIGS API discards (clips) the data outside of this specified range. The size of the root window is the maximum size that the workstation supports. If a window is larger than the maximum device coordinate extents, then the area beyond the extents is unused.

With the*length* and *data* parameters, your application must specify a rendering scale factor to scale primitive nominal sizes (line widths, marker sizes, edge widths, and annotation heights).This scale factor affects the nominal size for all primitives that have nominal sizes. In addition, the graPHIGS API applies this scale factor to the scale factor structure elements (i.e., marker size scale factor)When the value is set to 1.0, no scaling is performed. The specified values take effect on the next invocation of the Update Workstation (**GPUPWS**)subroutine.

Use the Inquire Mapped Display Surface Size(**GPQMDS**) subroutine to inquire the size of the mapped display surface.

This subroutine is ignored on workstations that do not use a window to display the graPHIGS API output.

## Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*method* **— specified by user, fullword integer**
Window mapping method (`1=MAPPED`, `2=DIRECT`).

*length* **— specified by user, fullword integer**
Length, in bytes, of the method-dependent data (>=0).

*data* — **specified by user, variable length data**
Method-dependent data. Depending on the value specified by the *method*parameter, method-dependent data is as follows:

**If *method* is `1=MAPPED`**
N/A (no data is required).

**If *method* is `2=DIRECT`**
Rendering scale factor (>0.0)

A short floating-point number that specifies the rendering scale factor that the graPHIGS API uses when rendering line width, marker size, edge width, and annotation height primitives. If the value is set to 1.0, then no scaling is performed.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **509** | DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH |
| **512** | METHOD NOT SUPPORTED |
| **516** | SCALE FACTOR IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPCCV** | Convert Coordinate Values |
| **GPGWIN** | Get Window |
| **GPQMDS** | Inquire Mapped Display Surface Size |

### RCP code

201330181 (X'0C000E05')

## GPDMF - Set Data Morphing Factors

**GPDMF (*flength, fdata*)**

### Purpose

Use **GPDMF** to insert a Set Data Morphing Factors structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Data Morphing Factors structure element, depending on the current edit mode.

During traversal, the graPHIGS API combines the values of *fdata* with a primitive's data morphing vectors to modify its rendered data mapping data.

In data morphing, the graPHIGS API combines the data mapping data values ($x_1$, $x_2$, ..., $x_{ndata}$) with the data morphing scale factors in the *fdata* parameter ($s_1$, $s_2$, ..., $s_{nscale}$) and the data morphing vectors (($d_{1,1}$, $d_{1,\ 2}$, ..., $d_{1,\ ndata}$), ($d_{2,\ 1}$, $d_{2,2}$, ..., $d_{2,ndata}$),..., ($d_{n\ vector,1}$, $d_{nvector,2}$, ..., $d_{nvector,ndata}$)) to obtain the new data mapping data values ($x'_1$, $x'_2$, ..., $x'_{ndata}$) as follows:

$$x'_1 = s_1\ x_1 + s_2\ d_1,1 + s_3\ d_{2,\ 1} +...+ s_{nscale}\ d_{nvector,1}$$

$$x'_2 = s_1\ x_2 + s_2\ d_1,2 + s_3\ d_2,2 +...+ s_{nscale}\ d_{nvector,2}$$

...

$x'_{ndata} = s_1 x_{ndata} + s_2 d_{1,\ ndata} + s_3 d_{2,\ ndata} +...+ s_{nscale} d_{\ nvector,ndata}$

These equations show that the number of morphing scale factors should be one more than the number of morphing vectors in the affected primitive (*nscale*=*nvector*+1). However, if the number of morphing vectors and scale factors disagree at traversal time, then 0 value vectors and scale factors are assumed wherever necessary. For example, if you supply too many scale factors for a given primitive (*nscale*>*nvector*+1), then the graPHIGS API ignores the extra scale factors, as if there were additional 0 valued morphing vectors in the primitive definition. If you supply too few scale factors (*nscale*<*nvector*+1), then the graPHIGS API ignores the extra morphing vectors, just as if there were additional scale factors with value zero in this function call.

The traversal default for data morphing is *flength*=1 and *fdata*={1.0}.

Use **GPQWDT** to inquire the morphing facilities of a specified workstation.

### Parameters

*flength* **— specified by user, fullword integer (>=1)**
Number of morphing factors.

*fdata* **— specified by user, array of short floating-point numbers**
List of morphing factors. The number of entries in this list is given by the *flength* parameter.

### Error Codes

| | |
|---|---|
| **5** | FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL) |
| **61** | LENGTH IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPBDMF** | Set Back Data Morphing Factors |
| **GPDMR** | Set Data Mapping Representation |
| **GPQWDT** | Inquire Workstation Description |
| **GPVMF** | Set Vertex Morphing Factors |

### RCP code

201343520(X'0C004220')

## GPGLX2 - Set Global Transformation 2

**GPGLX2 (*matrix*)**

### Purpose

Use **GPGLX2** to insert a two-dimensional, Set Global Transformation 2 structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Global Transformation 2 structure element depending on the current edit mode.

When encountered during traversal, this element is expanded by the graPHIGS API into a 4 x 4 matrix as follows:

```
|a b c|          |a b 0 c|
|d e f| ------>  |d e 0 f|
|g h i|          |0 0 1 0|
                 |g h 0 i|
```

and causes the expanded matrix to become the current global transformation for the current structure. The resultant matrix, in conjunction with the local modeling transformation, transforms all subsequent primitives from the Modeling Coordinate (MC) system to the World Coordinate (WC) system.

> **Note:** When inquired, the matrix returned by the Inquire Element Content (**GPQE**) subroutine is the expanded 4 x 4 matrix; the matrix returned by the Inquire List of Element Data (**GPQED**) subroutine is the 3 x 3 matrix.

### Parameters

*matrix* **— specified by user, 9 short floating-point numbers**
Transformation matrix (3 x 3)

The elements must be in the following order for the input transformation matrix:

```
|m11 m12 m13|
|m21 m22 m23|---> (m11,m12,m13,m21,m22,m23,m31,m32,m33)
|m31 m32 m33|
```

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

### Related Subroutines

None

### RCP code

201329668 (X'0C000C04')

## GPGLX3 - Set Global Transformation 3

**GPGLX3 (*matrix*)**

### Purpose

Use **GPGLX3** to insert a three-dimensional, Set Global Transformation 3 structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Global Transformation 3 structure element depending on the current edit mode.

When encountered during traversal, this element causes the specified matrix to replace the current global transformation for the current structure. The resultant matrix, in conjunction with the local modeling transformation, transforms all subsequent primitives from the Modeling Coordinate (MC) system to the World Coordinate (WC) system.

### Parameters

*matrix* **— specified by user, 16 short floating-point numbers**
Transformation matrix (4 x 4).

The elements must be in the following order for the input transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24|---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**Related Subroutines**

None

**RCP code**

201329667 (X'0C000C03')

## GPMCI - Set Modeling Clipping Indicator

**GPMCI (*indic*)**

**Purpose**

Use **GPMCI** to insert a Set Modeling Clipping Indicator structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Modeling Clipping Indicator structure element, depending on the current edit mode.

Use this subroutine to set the value of the current modeling clipping indicator (*indic*). During traversal, the graPHIGS API uses the value of the modeling clipping indicator to determine whether to perform modeling clipping on subsequent output primitives.

The traversal default for the modeling clipping indicator is `1=NOCLIP`.

Not all graPHIGS API workstations support modeling clipping. Use **GPQWDT** to determine if a particular workstation supports modeling clipping.

**Parameters**

*indic* **— specified by user, fullword integer**
Modeling clipping indicator (`1=NOCLIP, 2=CLIP`).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**332** CLIP INDICATOR VALUE IS INVALID

**Related Subroutines**

**GPMCV2**
Set Modeling Clipping Volume 2

**GPMCV3**
Set Modeling Clipping Volume 3

**GPRMCV**
Restore Modeling Clipping Volume

**GPQWDT**
Inquire Workstation Description

**RCP code**

201329674 (X'0C000C0A')

## GPMCV2 - Set Modeling Clipping Volume 2

**GPMCV2 (*oper, number, lhspace*)**

### Purpose

Use **GPMCV2** to insert a Set Modeling Clipping Volume 2 structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Modeling Clipping Volume 2 structure element, depending on the current edit mode.

This element modifies the current modeling clipping volume. Each modeling clipping half-space (*lhspace*) contains a point and a vector defined in modeling coordinates (MC). The graPHIGS API expands each two-dimensional half-space to a three-dimensional half-space by setting the *z* coordinate of both the point and the vector to the value 0.0.

The current modeling transformation transforms each pair of half-spaces (consisting of a point and vector) from the Modeling Coordinate (MC) system to the World Coordinate (WC) system, and defines a boundary (plane) in WC. The transformed point is on this plane and the transformed vector defines a normal to the plane which points into the acceptance half-space. The clipping volume is obtained by intersecting all acceptance half-spaces in the list specified by this element.

During traversal, the volume specified by this element either replaces or intersects the current clipping volume, depending on the value specified by the modeling clipping operator *oper*. The graPHIGS API uses the resultant clipping volume to render subsequent primitives. Transformation elements encountered during traversal do not affect the resultant clipping volume. The resultant clipping volume is called the *acceptance region* because primitives that lie within it are accepted for display. The graPHIGS API clips portions of subsequent primitives outside the acceptance region.

If the number (*number*) of modeling clipping half-spaces is set to 0, then the acceptance region is all of world coordinate space (WC).

During traversal, if the workstation does not support the specified modeling clipping operator, if the specified number of clipping half-spaces exceeds the maximum supported by the workstation, or if any half-space is found to be degenerate, then the graPHIGS API ignores this structure element.

During traversal, if the graPHIGS API encounters a Set Modeling Clipping Volume 2 or 3 structure element and the current composite modeling transformation matrix is singular, then the graPHIGS API sets the effective clipping volume to the null volume and clips all subsequent primitives.

### Parameters

*oper*— **specified by user, fullword integer**
Modeling clipping operator (`1=REPLACE_VOLUME, 2=INTERSECT_VOLUME`).

*number*— **specified by user, fullword integer**
Number of modeling clipping half-spaces (>=0).

*lhspace*— **specified by user, array of short floating-point values**
Modeling clipping half-spaces. Each modeling clipping half-space is defined by a point and a normal. The point and normal are specified in the order *Px, Py, Nx,* and *Ny.*

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**627** NUMBER OF HALF-SPACES < ZERO

**628** OPERATOR IS INVALID

**Related Subroutines**

**GPMCI**
Set Modeling Clipping Indicator

**GPRMCV**
Restore Modeling Clipping Volume

**GPMCV3**
Set Modeling Clipping Volume 3

**GPQWDT**
Inquire Workstation Description

**RCP code**

201329673 (X'0C000C09')

## GPMCV3 - Set Modeling Clipping Volume 3

**GPMCV3 (*oper, number, lhspace*)**

**Purpose**

Use **GPMCV3** to insert a Set Modeling Clipping Volume 3 structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Modeling Clipping Volume 3 structure element, depending on the current edit mode.

This element specifies the current modeling clipping volume. Each modeling clipping half-space (*lhspace*) contains a point and a vector defined in modeling coordinates (MC). The current modeling transformation transforms each pair of half-spaces (consisting of a point and vector) from the Modeling Coordinate (MC) system to the World Coordinate (WC) system, and defines a boundary (plane) in WC. The transformed point is on this plane, and the transformed vector defines a normal to the plane which points into the acceptance half-space. The clipping volume is obtained by intersecting all acceptance half-spaces in the list specified by this element.

During traversal, the volume specified by this element either replaces or intersects the current clipping volume, depending on the value specified by the modeling clipping operator *oper*. The graPHIGS API uses the resultant clipping volume to render subsequent primitives. Transformation elements encountered during traversal do not affect the resultant clipping volume. The resultant clipping volume is called the *acceptance region* because primitives that lie within it are accepted for display. The graPHIGS API clips portions of subsequent primitives outside the acceptance region.

If the number (*number*) of modeling clipping half-spaces is set to 0, then the acceptance region is all of world coordinate space (WC).

During traversal, if the workstation does not support the specified modeling clipping operator, if the specified number of clipping half-spaces exceeds the maximum supported by the workstation, or if any half-space is found to be degenerate, then the graPHIGS API ignores this structure element.

During traversal, if the graPHIGS API encounters a Set Modeling Clipping Volume 2 or 3 structure element and the current composite modeling transformation matrix is singular, then the graPHIGS API sets the effective clipping volume to the null volume and clips all subsequent primitives.

**Parameters**

*oper***— specified by user, fullword integer**
Modeling clipping operator (`1=REPLACE_VOLUME, 2=INTERSECT_VOLUME`).

*number***— specified by user, fullword integer**
Number of modeling clipping half-spaces (>=0).

*lhspace***— specified by user, array of short floating-point values**
Modeling clipping half-spaces. Each modeling clipping half-space is defined by a point and a normal. The point and normal are specified in the order *Px, Py, Pz, Nx, Ny,* and *Nz.*

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**627** NUMBER OF HALF-SPACES < ZERO

**628** OPERATOR IS INVALID

**Related Subroutines**

**GPMCI**
Set Modeling Clipping Indicator

**GPMCV2**
Set Modeling Clipping Volume 2

**GPRMCV**
Restore Modeling Clipping Volume

**GPQWDT**
Inquire Workstation Description

**RCP code**

201329672 (X'0C000C08')

## GPMLX2 - Set Modeling Transformation 2

**GPMLX2 (*matrix*, *type*)**

**Purpose**

Use **GPMLX2** to insert a two-dimensional, Set Modeling Transformation 2 structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Modeling Transformation 2 structure element depending on the current edit mode.

When encountered during traversal, the 3 x 3 modeling matrix is expanded into a 4 x 4 matrix as follows:

```
|a b c|          |a b 0 c|
|d e f| ----->   |d e 0 f|
|g h i|          |0 0 1 0|
                 |g h 0 i|
```

Depending on the composition type, when this element is encountered during traversal, the specified matrix either replaces, is pre-concatenated with, or is post-concatenated with the current local modeling transformation matrix. The resultant matrix, in conjunction with the global modeling transformation, transforms all subsequent primitives from the Modeling Coordinate (MC) system to the World Coordinate (WC) system.

> **Note:** When inquired, the matrix returned by the Inquire Element Content (**GPQE**) subroutine is the expanded 4 x 4 matrix; the matrix returned by the Inquire List of Element Data (**GPQED**) subroutine is the 3 x 3 matrix.

**Parameters**

*matrix* **— specified by user, 9 short floating-point numbers**
Transformation matrix (3 x 3).

For the transformation matrix, the elements must be in the following order:

```
|m11 m12 m13|
|m21 m22 m23| ---> (m11,m12,m13,m21,m22,m23,m31,m32,m33)
|m31 m32 m33|
```

*type* **— specified by user, fullword integer**
Composition type (1=PRECONCATENATE, 2=POSTCONCATENATE, 3=REPLACE)

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**319** COMPOSITION TYPE VALUE IS INVALID

**Related Subroutines**

None

**RCP code**

201329666 (X'0C000C02')

# GPMLX3 - Set Modeling Transformation 3

**GPMLX3 (*matrix*, *type*)**

**Purpose**

Use **GPMLX3** to insert a three-dimensional, Set Modeling Transformation 3 structure element into the open structure following the element pointer or replace the element pointed at by the element pointer with a Set Modeling Transformation 3 structure element depending on the current edit mode.

Depending on the composition type, when this element is encountered during traversal, the specified matrix either replaces, is pre-concatenated with, or is post-concatenated with the current local modeling

transformation matrix. The resultant matrix, in conjunction with the global modeling transformation, transforms all subsequent primitives from the Modeling Coordinate (MC) system to the World Coordinate (WC) system.

**Parameters**

*matrix* **— specified by user, 16 short floating-point numbers**
Transformation matrix (4[default]4).

For the transformation matrix, the elements must be in the following order:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

*type* **— specified by user, fullword integer**
Composition type (1=PRECONCATENATE, 2=POSTCONCATENATE, 3=REPLACE).

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**319** COMPOSITION TYPE VALUE IS INVALID

**Related Subroutines**

None

**RCP code**

201329665 (X'0C000C01')

## GPRMCV - Restore Modeling Clipping Volume

**GPRMCV**

**Purpose**

Use **GPRMCV** to insert a Restore Modeling Clipping Volume structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Restore Modeling Clipping Volume structure element, depending on the current edit mode.

During traversal, the Restore Modeling Clipping Volume structure element restores the current modeling clipping volume in the graPHIGS API traversal state list to:

- the volume inherited by the structure if executed via **GPEXST**

  or
- the volume saved by the previous Push Set TSL (Traversal State List) subroutine (**GPPSTS** page GPPSTS - Push Set TSL)

  or
- the initial modeling clipping volume.

The graPHIGS API uses this volume to clip all subsequent primitives during traversal.

**Error Codes**

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**Related Subroutines**

**GPMCI**
Set Modeling Clipping Indicator

**GPMCV2**
Set Modeling Clipping Volume 2

**GPMCV3**
Set Modeling Clipping Volume 3

**GPPSTS**
Push Set TSL

**RCP code**

201329675 (X'0C000C0B')

## GPVMF - Set Vertex Morphing Factors

**GPVMF (*flength,fdata*)**

**Purpose**

Use **GPVMF** to insert a Set Vertex Morphing Factors structure element into the open structure following the element pointer, or to replace the element pointed at by the element pointer with a Set Vertex Morphing Factors structure element, depending upon the edit mode.

During traversal, the graPHIGS API combines the value of *fdata* with a primitive's vertex morphing values to modify its rendered vertex coordinates.

In vertex morphing, the graPHIGS API combines the vertex coordinate values *(x, y, z)* with the vertex morphing scale factors in the *fdata* parameter ($s_1$, $s_2$, ..., $s_{nscale}$ ) and the vertex morphing vectors

(( $dx_1$, $dy_1$, $dz_1$), ( $dx_2$, $dy_2$, $dz_2$), ..., ( $dx_{nvector}$, $dy_{nvector}$, $dz_{nvector}$ ))

to create the new vertex coordinate values (x', y', z') as follows:

$x' = s_1\ x + s_2\ dx_1 + s_3\ dx_{\#EMPTY>2} +...+ s_{nscale}\ dx_{\#EMPTY>nvector}$

$y' = s_1\ y + s_2\ dy_1 + s_3\ dy_{\#EMPTY>2} +...+ s_{nscale}\ dy_{\#EMPTY>nvector}$

$z' = s_1\ z + s_2\ dz_1 + s_3\ dz_{\#EMPTY>2} +...+ s_{nscale}\ dz_{\#EMPTY>nvector}$

These equations show that the number of morphing scale factors should be one more than the number of morphing vectors in the affected primitive (*nscale*=*nvector*+1). However, if the number of morphing vectors and scale factors disagree at traversal time, then 0 value vectors and scale factors are assumed wherever necessary. For example, if you supply too many scale factors for a given primitive (*nscale*>*nvector*+1), then the graPHIGS API ignores the extra scale factors, as if there were additional 0 valued morphing vectors in the primitive definition. If you supply too few scale factors (*nscale*<*nvector*+1), then the graPHIGS API ignores the extra morphing vectors, just as if there were additional morphing scale factors with value zero in this function call.

The traversal default for vertex morphing is *flength*=1 and *fdata*={1.0}.

Use **GPQWDT** to inquire the morphing facilities of a specified workstation.

### Parameters

*flength* **— specified by user, fullword integer (>=1)**
Number of morphing factors.

*fdata* **— specified by user, array of short floating-numbers**
List of morphing factors. The number of entries in this list is given by the *flength* parameter.

### Error Codes

**5** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**61** LENGTH IS INVALID

### Related Subroutines

**GPBDMF**
Set Back Data Morphing Factors

**GPDMF**
Set Data Morphing Factors

**GPQWDT**
Inquire Workstation Description

### RCP code

201343519 (X'0C00421F')

## GPWSX2 - Set Workstation Transformation 2

**GPWSX2 (*wsid*, *window*, *viewpt*)**

### Purpose

Use **GPWSX2** to set the requested two-dimensional workstation transformation for the specified workstation.

The *x* component and *y* component of the current workstation window and viewport are set to the requested values when the workstation is updated. The Zmin of the window is set to zero; the Zmax of the window is set to the smaller of Xmax - Xmin and the Ymax - Ymin. The Zmin of the viewport is set to zero; the Zmax of the viewport is set to the larger of Xmax - Xmin and Ymax - Ymin.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*window* **— specified by user, 4 short floating-point numbers (NPC)**
Workstation window (Xmin, Xmax, Ymin, Ymax).

*viewpt* **— specified by user, 4 short floating-point numbers (DC)**
Workstation viewport (Xmin, Xmax, Ymin, Ymax).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**44** INVALID WINDOW DEFINITION

**330** INVALID VIEWPORT

### Related Subroutines

**GPQWSX**
Inquire Workstation Transformation

### RCP code

201330180 (X'0C000E04')

## GPWSX3 - Set Workstation Transformation 3

**GPWSX3 (*wsid*, *window*, *viewpt*)**

### Purpose

Use **GPWSX3** to set the requested three-dimensional workstation window and viewport for the specified workstation values.

The current workstation window and viewport are set to the requested values when the workstation is updated.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*window* **— specified by user, 6 short floating-point numbers (NPC)**
Workstation window (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*viewpt* **— specified by user, 6 short floating-point numbers (DC)**
Workstation viewport (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**44** INVALID WINDOW DEFINITION

**330** INVALID VIEWPORT

### Related Subroutines

**GPQWSX**
Inquire Workstation Transformation

### RCP code

201330179 (X'0C000E03')

# Chapter 11. Input Subroutines

Input subroutines allow users to supply input to your application. There are six logical input device classes: locator, stroke, valuator, choice, pick, and string. There are three modes of interaction with the input devices: sample, request, and event. Input subroutines also allow a program to emulate physical device input for one or more logical devices.

The subroutines discussed in this section perform the following operations:

- initialization of an input device
- setting the operating mode of an input device
- requesting input from a device
- sampling an input device's current value
- managing the event queue
- retrieving input values from the event queue
- emulating a physical device
- defining a cursor.

To determine the actual input capabilities of a specific workstation, use the inquire subroutines (see Chapter 16 "Inquire Subroutines").

## GPAWEV - Await Event

**GPAWEV (*time*, *major*, *class*, *minor*)**

### Purpose

Use **GPAWEV** to move the next event from the event queue into the current event report. If the input queue is empty, the graPHIGS API is placed in a wait state until at least one of the following occurs:

1. An event is added to the event queue.
2. The time specified in the timeout parameter has elapsed.
3. An error is reported by any nucleus connected to the shell.

If the time value is zero, no wait takes place. If the time value is not zero, a wait takes place for the specified time interval. The maximum time interval is 55,800 seconds (15.5 hours).

The timing is done using the operating system's timing facility. (See *The graPHIGS Programming Interface: Writing Applications*.)

When a timeout or error situation occurs, the graPHIGS API returns zero for the event class parameter and major/minor code parameters are not set. Otherwise, the graPHIGS API returns the major code, class, and minor code of the event in the current event report. For the details of event class, major/minor codes and event data, see *The graPHIGS Programming Interface: Technical Reference*.

The application must use the appropriate Get subroutine call to obtain the value(s) of the input residing in the current event report.

### Parameters

*time* **— specified by user, short floating-point number**
Timeout interval in seconds (>=0.0).

*major* **— returned by the graPHIGS API, fullword integer**
Major event code.

*class* **— returned by the graPHIGS API, fullword integer**
Event class.

*minor* **— returned by the graPHIGS API, fullword integer**
Minor event code.

### Error Codes

| | |
|---|---|
| **147** | EVENT QUEUE HAS OVERFLOWED |
| **151** | TIMEOUT VALUE < ZERO |
| **168** | INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION |
| **532** | TIME INTERVAL IS TOO LARGE |

**Note:** The operation is performed even if error 147 occurs.

### Related Subroutines

| | |
|---|---|
| **GPCHMO** | Set Choice Mode |
| **GPEVHN** | Define Event Handling Subroutine |
| **GPFLEV** | Flush Device Event |
| **GPFWEV** | Flush Workstation Event |
| **GPGTCH** | Get Choice |
| **GPGTLC** | Get Locator |
| **GPGTPK** | Get Pick |
| **GPGTSK** | Get Stroke |
| **GPGTST** | Get String |
| **GPGTVL** | Get Valuator |
| **GPGTXP** | Get Extended Pick |
| **GPLCMO** | Set Locator Mode |
| **GPPKMO** | Set Pick Mode |
| **GPSBMS** | Send Broadcast Message |
| **GPSKMO** | Set Stroke Mode |
| **GPSPMS** | Send Private Message |
| **GPSSTH** | Set Structure Store Threshold |
| **GPSTMO** | Set String Mode |
| **GPQCEV** | Inquire Current Event |
| **GPQIQO** | Inquire Input Queue Overflow |
| **GPQSEV** | Inquire More Simultaneous Events |

### RCP code

201338113 (X'0C002D01')

## GPBKAC - Set Break Action

**GPBKAC (*wsid*, *trigger*)**

### Purpose

Use **GPBKAC** to set the break action to a given trigger type and trigger qualifier at the specified workstation.

This subroutine lets the application specify which operator action is to be interpreted as "break" for input device operations on the given workstation. For example, the default break action on the 5080 workstation is the alternate-cancel key combination.

The break action is the operator action that terminates an outstanding Request input subroutine at a workstation. This break action is used to terminate Request operations for all device classes.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*trigger* — **specified by user, 2 fullword integers**
Trigger to be used on the specified workstation for the break action. The trigger consists of a trigger type followed by a trigger qualifier. Positive integers as trigger types are button type physical device numbers. The trigger qualifier for a button device is the physical button number. If a trigger type is for a keyboard and the qualifier is between 0 and 255, the workstation's primary character set is used to interpret the qualifier.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **38** | WORKSTATION HAS ONLY OUTPUT CAPABILITIES |
| **567** | A TRIGGER TYPE VALUE IS INVALID |
| **568** | A TRIGGER QUALIFIER VALUE IS INVALID |
| **572** | WORKSTATION DOES NOT SUPPORT PROGRAMMABLE BREAK ACTION |

### Related Subroutines

| | |
|---|---|
| **GPIT** | Set Input Device Trigger |
| **GPQABK** | Inquire Actual Break Capabilities |
| **GPQBK** | Inquire Break Capabilities |
| **GPQBKS** | Inquire Break Action State |
| **GPQDBK** | Inquire Default Break Action |

### RCP code

201327878 (X'0C000506')

## GPCHMO - Set Choice Mode

**GPCHMO (*wsid*, *device*, *mode*, *echosw*)**

### Purpose

Use **GPCHMO** to set the operating mode of a choice input device.

After the choice mode is set, its echoing state is set to `1=NOECHO` or `2=ECHO` based on the *echosw* parameter. Depending on the specified operating mode, `1=REQUEST`, `2=SAMPLE`, or `3=EVENT`, an interaction with the given device may begin or end.

> **Note:** The input device is reset with the initialization values when the **GPCHMO** subroutine is called with *mode* parameters set to SAMPLE or EVENT.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*device* — **specified by user, fullword integer**
Choice device number.

*mode* — **specified by user, fullword integer**
Operating mode (1=REQUEST, 2=SAMPLE, 3=EVENT).

*echosw* — **specified by user, fullword integer**
Echo switch (1=NOECHO, 2=ECHO).

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **140** | DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE |
| **168** | INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION |
| **326** | OPERATING MODE IS INVALID |
| **327** | ECHO SWITCH VALUE IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPAWEV** | Await Event |
| **GPINCH** | Initialize Choice |
| **GPQLI** | Inquire List of Logical Input Devices |
| **GPRQCH** | Request Choice |
| **GPSMCH** | Sample Choice |

### RCP code

201335812 (X'0C002404')

## GPCUR - Set Cursor Representation

**GPCUR (*wsid*, *index*, *format*, *shape*)**

### Purpose

Use **GPCUR** to set the cursor shape into the specified cursor shape table entry of the specified workstation.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*index* — **specified by user, fullword integer**
Cursor shape table index (>=1).

*format* — **specified by user, fullword integer**
Format of the cursor definition in *shape*.

Valid formats are:

**1 -** Fixed Size Raster, 1 bit per pixel.
**>=2 -** Reserved.

*shape* **— specified by user, variable data**
Cursor definition data. This parameter contains the information required to define a cursor shape. The content of this parameter is interpreted according to the *format* parameter.

**Type 1** cursor format requires a rectangular array of pixels, each 1 bit deep. A value of one in a pixel location indicates that the cursor color should be displayed for the screen pixel which is overlayed by the cursor pixel. A value of zero indicates that the underlying screen pixel should not be modified or overlayed. The following syntax is required for this format:

| Field | Offset (bytes) | Length (bytes) | Type/Interpretation |
|---|---|---|---|
| ncol | 0 | 4 | fullword integer. *x* size of pixel array (number of columns) |
| nrow | 4 | 4 | fullword integer. *y* size of pixel array (number of rows) |
| pixels | 8 | (ncol+31)/32*nrow*4 | Array of bits, 1/pixel. Each row starts on a word (32 bit) boundary. |

For this format, the *x* size and *y* size of the specified pixel array must match the maximum pixel array size as returned by the Inquire Cursor Facilities (**GPQCUF**) subroutine.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **37** | WORKSTATION IS NOT OF CATEGORY OUTIN |
| **181** | CURSOR PIXEL ARRAY SIZE IS INVALID |
| **182** | CURSOR SHAPE TABLE INDEX NOT WITHIN WORKSTATION TABLE RANGE |
| **183** | SPECIFIED CURSOR FORMAT IS NOT SUPPORTED |

### Related Subroutines

| | |
|---|---|
| **GPCUS** | Set Cursor Shape |
| **GPQCUF** | Inquire Cursor Facilities |

### RCP code

201344772 (X'0C004704')

## GPCUS - Set Cursor Shape

**GPCUS (*wsid*, *ctype*)**

### Purpose

Use **GPCUS** to select a cursor definition from the workstation's cursor shape table. The graPHIGS API uses this definition for the graphics cursor of the input devices which are controlled by a two-dimensional vector type physical input device. The cursor definition selected by this subroutine is used as a prompt of every input device which has a prompt that is the graphic cursor.

Optional cursor definitions include, full screen cross hair, disable (none), two color logical input, and user-defined. Only workstations which use the facilities of a window system (e.g., X-Windows) can specify none (-2) or two color cursor logical input (-3). A cursor definition of (-2) (none) allows the application to disable the graPHIGS API cursors and to use the cursor facilities of the window system instead.

The default cursor is a single color cursor with a shape type of zero. This is compatible with Version 1 of the graPHIGS API.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*ctype* — **specified by user, fullword integer**
Cursor definition. This parameter must contain one of the following values that are available on the workstation:

| | |
|---|---|
| **-3** | Two color cursor logical input. |
| **-2** | None. |
| **-1** | Full screen cross hair cursor. |
| **0** | Logical input device dependent cursor. |
| **>=1** | User defined cursor shape. This value is used as an index to the workstation's cursor shape table and the cursor pointed to by this index is used as the graphical cursor. |

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **37** | WORKSTATION IS NOT OF CATEGORY OUTIN |
| **180** | CURSOR SHAPE TYPE IS NOT SUPPORTED |
| **182** | CURSOR SHAPE TABLE INDEX NOT WITHIN WORKSTATION TABLE RANGE |

### Related Subroutines

**GPCUR** Set Cursor Representation

### RCP code

201344771 (X'0C004703')

## GPEPD - Emulate Physical Device

**GPEPD (*wsid*, *category*, *device*, *value*)**

### Purpose

Use **GPEPD** to emulate input from a physical device to one or more logical input devices. The set of logical input devices which will receive this physical input is fixed.

The physical device must first be disabled before it can be emulated through this subroutine. An error is generated otherwise.

If you are using physical device emulation on the X workstation, refer to *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*category* **— specified by user, fullword integer**
Physical device category (1=BUTTON, 2=SCALAR, 3=2D_VECTOR).

*device* **— specified by user, fullword integer**
Physical device number (>=1).

*value* **— specified by user, array of fullword integers**
Input values. This array contains a list of values that are to be passed to the measure mapping process of the logical input devices that receive their input from the specified physical device. The length and content of this array is determined by the category of physical input device as follows:

| Category | Number of values to be passed in |
|---|---|
| 1=BUTTON | 1 |
| 2=SCALAR | 1 |
| 3=2D_VECTOR | 2 |

If the physical device generates **absolute values**, then the specified *value* must be in the range defined for the physical input device. To determine the range for the physical input device, use Inquire Physical Device Characteristics (**GPQPDC**) subroutine.

If the physical device generates **relative values**, then the specified *value* can by any positive or negative integer which represents increments of physical motion. The value is used to compute a **change** in the measure of the logical device using the following formula:

```
                      Logical Range high  - Logical Range low
------------------------------------------------------------------------------- x value
#of Physical Device Increments / Unit of Physical Motion x #Units of Physical Motion
```

The **change** is then added to the current measure of the logical device. If the result exceeds a limit of the range, then the measure is set to that limit.

**Logical Range**
Specified on the Initialize Device subroutine call. It is the range of physical input values that the logical device can input.

**Unit of Physical Motion**
For a dial physical device, a unit of physical motion equals one turn of the dial (unless this is changed by the Initialize Valuator [**GPINVL**] [page GPINVL - Initialize Valuator] subroutine). For a relative vector device, the WDT range of the device's physical motion equals the amount of motion necessary to move the cursor from one edge of the screen to its opposite edge.

> **Note:** For keyboard physical devices, if the *value* is between 0 and 255, the workstation's primary character set is used to interpret the *value*.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**160** PHYSICAL INPUT DEVICE CATEGORY IS INVALID

**161** PHYSICAL INPUT VALUE IS INVALID

**162** PHYSICAL INPUT DEVICE HAS NOT BEEN DISABLED

**Related Subroutines**

**GPPDMO**
Set Physical Device Mode

**GPQPDC**
Inquire Physical Device Characteristics

**GPQSPD**
Inquire Source Physical Device

**RCP code**

201344770 (X'0C004702')

## GPEVHN - Define Event Handling Subroutine

**GPEVHN (*event-handler*, *anchor*)**

**Purpose**

Use **GPEVHN** to specify the address of an application-defined event handling procedure.

If the specified application event handler routine address is not zero, the graPHIGS API gives control to the routine for each event received by the shell from any connecting nucleus. When the routine address is zero, the graPHIGS API performs event handling without calling any application defined event handler.

The application event handler defined by this subroutine is invoked from the graPHIGS API event handling routine running in a special run time environment. The environment depends on the operating system on which the graPHIGS API shell is running and on the communication mechanism used to communicate with each nucleus. If two graPHIGS API nuclei are connected to the graPHIGS API shell and communication methods for these nuclei are different, the application event handler may be invoked in two different environments. Therefore, the application event handler:

- Must be written in a programming language which is environment independent.
- Must follow the standard linkage convention of the operating system on which the graPHIGS API shell is running.
- Must not issue any graPHIGS API subroutine call.
- Should not issue nor invoke any system subroutine which may require the normal run time environment.
- Must return to the graPHIGS API event handler.

For the details of event class, major/minor codes and event data, see *The graPHIGS Programming Interface: Technical Reference*.

When the application event handler is invoked, it receives the following parameters by the "call by reference mechanism" via the operating system's standard linkage convention. The application event handler should never modify any parameter other than the return flag:

*anchor* — **variable data**
Application anchor. This is the second parameter of the **GPEVHN** subroutine. Anchor is a user defined data area that the graPHIGS API passes to your event handler. The parameter is passed without any checking or modification.

*major* — **fullword integer**
Major code of the event.

*class* — **fullword integer**
Event class of the event.

*minor* — **fullword integer**
Minor code of the event.

*length* — **fullword integer**
Length of event data in bytes.

*data* — **variable length data**
Event data. For the format of event data see *The graPHIGS Programming Interface: Technical Reference*.

*sflag* — **fullword integer**
Status flag. Flags representing the event queue status. The following bits are set:

**bit 0-28**
Reserved, is set to 0.

**bit 29** Event queue space flag (1=NO_MORE_SPACE_AVAILABLE, 0=SPACE_AVAILABLE).

**bit 30** Event queue overflow flag (1=ALREADY_OVERFLOWED, 0=NOT_OVERFLOWED_YET). When this bit is 1, bit 29 is always 1.

**bit 31** More simultaneous event flag (1=MORE_SIMULTANEOUS_EVENT, 0=NO_MORE_SIMULTANEOUS_EVENT).

*rflag* — **fullword integer**
Return flag. Flags specifying required operations. The following bits should be set:

| | |
|---|---|
| bit 0-30 | Reserved, should be set to 0. |
| bit 31 | Discard flag (1=SHOULD_BE_DISCARDED, 0=SHOULD_BE_ENQUEUED). Even if the application specifies 0 for this bit, the event is discarded when there is no more space available, that is, when bit 29 of the status flag is 1. If this situation occurs in the process of simultaneous events, it also causes deletion of all events within the simultaneous event group and the event queue enters an overflow state. Note that in the case of queue overflow the deletion of a set of simultaneous events includes deletion of events already reported to the application event handler. |

**Parameters**

*event-handler* — **specified by user, fullword integer**
Address of the application event handler routine or a fullword integer of 0.

*anchor* — **specified by user, variable data**
Parameter to be passed back to the application event handler. Anchor is for user defined data.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201338117 (X'0C002D05')

## GPFLEV - Flush Device Event

**GPFLEV (*wsid*, *class*, *device*)**

**Purpose**

Use **GPFLEV** to discard all input events from the specified logical input device.

All events received from the specified input device, matching the specified device class, workstation identifier and device number, are removed from the event queue.

If the current event report includes an event matching the specified input device, the current event report is also removed.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier (event major code).

*class* **— specified by user, fullword integer**
Device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*device* **— specified by user, fullword integer**
Device number (event minor code).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**147** EVENT QUEUE HAS OVERFLOWED

**328** INPUT CLASS VALUE IS INVALID

**Note:** The operation is performed even if error 147 occurs.

**Related Subroutines**

**GPFWEV**
Flush Workstation Event

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201338114 (X'0C002D02')

## GPFWEV - Flush Workstation Event

**GPFWEV (*wsid*)**

### Purpose

Use **GPFWEV** to discard all events from the specified workstation.

All events from the specified workstation, with an event class range between 1 and 200 will be removed from the event queue. All other events with classes greater than 200 will remain on the event queue.

If the current event report includes an event from the specified workstation, the current event report is also removed.

For information about events, see *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier (event major code).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**147** EVENT QUEUE HAS OVERFLOWED

**Note:** The operation is performed even if error 147 occurs.

### Related Subroutines

**GPFLEV**
Flush Device Event

### RCP code

201338116 (X'0C002D04')

## GPGTCH - Get Choice

**GPGTCH (*choice*)**

### Purpose

Use **GPGTCH** to retrieve a choice input value from the current event report. The device to which this value corresponds was identified on the previous invocation of the Await Event (**GPAWEV**) subroutine. The event is not removed from the current event report until the next invocation of **GPAWEV**.

### Parameters

*choice* **— returned by the graPHIGS API, fullword integer**
Choice value of event in current event report.

### Error Codes

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**519** NO CURRENT EVENT REPORT AVAILABLE

**Related Subroutines**

**GPAWEV**
Await Event

**RCP code**

201336068 (X'0C002504')

## GPGTLC - Get Locator

**GPGTLC (*view*, *pos*)**

**Purpose**

Use **GPGTLC** to retrieve a locator input value from the current event report. The event is not removed from the current event report until the next invocation of the Await Event (**GPAWEV**) subroutine.

The view index indicates the view table entry whose matrix was used to convert the locator point to World Coordinates (WC). This was the view active for input with the highest input priority at the indicated screen location.

The device to which this value corresponds was identified on the previous invocation of **GPAWEV**.

Locator input is returned from the view active for input with the highest input priority under the cursor. View zero is the highest priority view unless modified by your application.

**Parameters**

*view* **— returned by the graPHIGS API, fullword integer**
View index.

*pos* **— returned by the graPHIGS API, 3 short floating-point numbers (WC)**
Locator position.

**Error Codes**

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**519** NO CURRENT EVENT REPORT AVAILABLE

**Related Subroutines**

**GPAWEV**
Await Event

**GPVIP**
Set View Input Priority

**GPVP** Set View Priority

**RCP code**

201336065 (X'0C002501')

## GPGTMS - Get Message

**GPGTMS (*ilen*, *olen*, *string*)**

### Purpose

Use **GPGTMS** to retrieve a message string from the current event report. The event is not removed from the current event report until the next invocation of the Await Event (**GPAWEV**) subroutine.

### Parameters

*ilen* **— specified by user, fullword integer**
Length of the message to be returned in bytes. This value specifies the size of array provided in the *string* parameter. If the message length in the current event report is longer than this length, the overage is truncated.

*olen* **— returned by the graPHIGS API, fullword integer**
Length of the message actually returned in bytes.

*string* **— returned by the graPHIGS API, variable length character string.**
A character string supplied by the sender.

### Error Codes

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**505** LENGTH OF RETURN ARRAY < ZERO

**519** NO CURRENT EVENT REPORT AVAILABLE

### Related Subroutines

**GPAWEV**
Await Event

**GPSBMS**
Send Broadcast Message

**GPSPMS**
Send Private Message

### RCP code

201336072 (X'0C002508')

## GPGTPK - Get Pick

**GPGTPK (*length*, *depth*, *pickpath*)**

### Purpose

Use **GPGTPK** to retrieve a pick input value from the current event report. The event is not removed from the current event report until the next invocation of the Await Event (**GPAWEV**) subroutine.

This value consists of a pick path describing the position of the picked primitive in the structure network. The pick path is returned in the order specified in the Initialize Pick (**GPINPK**) subroutine, that is, `1=TOP_FIRST` or `2=BOTTOM_FIRST`. If **GPINPK** has not been called, the default value is `1=TOP_FIRST`.

The device to which this value corresponds was identified on the previous invocation of **GPAWEV** (like other Get subroutines).

### Parameters

*length* **— specified by user, fullword integer**
Length of the array provided by the application for return of the pick path data (>=0).

This value is specified as the size of the pick path array in *pickpath* entries. Each entry in the path is three fullword integers. If the actual pick information is longer than the pick path array provided by the application, the overage is truncated.

*depth* **— returned by the graPHIGS API, fullword integer**
Depth of pick path returned in the *pickpath* parameter.

This value is specified as the size of the pick path array in *pickpath* entries. Each entry in the path is three fullword integers.

*pickpath* **— returned by the graPHIGS API, array of fullword integers**
Array containing the pick path. The array consists of structure identifier, pick identifier, and element number triplets, making up the path to the picked primitive. The path is returned in the order specified by the initialize pick subroutine.

If the actual pick information is longer than the *pickpath* array provided by the application, then the overage is truncated.

Pick path data is set in the *pickpath* parameter in the following order: Entry 1, Entry 2, ...Entry N.

| | | | |
|---|---|---|---|
| Entry 1 | Structure ID 1 | Pick ID 1 | Element #1 |
| Entry 2 | Structure ID 2 | Pick ID 2 | Element #2 |
| . | . | . | . |
| . | . | . | . |
| . | . | . | . |
| Entry N | Structure ID N | Pick ID N | Element #N |

### Error Codes

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**505** LENGTH OF RETURN ARRAY < ZERO

**519** NO CURRENT EVENT REPORT AVAILABLE

### Related Subroutines

**GPAWEV**
Await Event

**GPGTXP**
Get Extended Pick

### RCP code

201336069 (X'0C002505')

## GPGTSK - Get Stroke

**GPGTSK (*length*, *view*, *npoint*, *pointlist*)**

### Purpose

Use **GPGTSK** to retrieve a stroke input device value from the current event report. The event is not removed from the current event report until the next invocation of the Await Event (**GPAWEV**) subroutine.

The number of points returned is limited to the current input buffer size found in the stroke data record at the time the device was placed in event mode. The view index indicates the view table entry whose matrix was used to convert the stroke points to World Coordinates (WC). This view was the view active for input with the highest input priority containing all the stroke locations.

The device to which this value corresponds was identified on the previous invocation of **GPAWEV**.

Stroke input is returned from the view active for input with the highest input priority which contains all the points. View 0 is the highest priority view, unless modified by your application.

### Parameters

*length* **— specified by user, fullword integer**
Length of point list array provided by application for the return of the stroke points. This value is specified as the maximum number of points that can be stored in the *pointlist* (>=0).

*view* **— returned by the graPHIGS API, fullword integer**
View index.

*npoint* **— returned by the graPHIGS API, fullword integer**
Number of points returned in the *pointlist*.

*pointlist* **— returned by the graPHIGS API, array of short floating-point numbers (WC)**
Coordinates of points in initial stroke buffer.

The points are ordered similarly to the *pointlist* parameters on output primitives.

### Error Codes

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**505** LENGTH OF RETURN ARRAY < ZERO

**519** NO CURRENT EVENT REPORT AVAILABLE

### Related Subroutines

**GPAWEV**
Await Event

### RCP code

201336066 (X'0C002502')

## GPGTST - Get String

**GPGTST (*ilen*, *olen*, *string*)**

### Purpose

Use **GPGTST** to retrieve a string input value from the current event report. The event is not removed from the current event report until the next invocation of the Await Event (**GPAWEV**) subroutine.

The length of the returned string is less than, or equal to, the buffer size found in the string data record at the time the device was placed in event mode.

The device to which this value corresponds was identified on the previous invocation of **GPAWEV**.

**Parameters**

*ilen* **— specified by user, fullword integer**
Length of string array in bytes provided by application to return string data (>=0).

*olen* **— returned by the graPHIGS API, fullword integer**
Length of the character string actually returned in bytes.

*string* **— returned by the graPHIGS API, variable length character string**
Character string.

**Error Codes**

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**505** LENGTH OF RETURN ARRAY < ZERO

**519** NO CURRENT EVENT REPORT AVAILABLE

**Related Subroutines**

**GPAWEV**
Await Event

**RCP code**

201336070 (X'0C002506')

## GPGTVL - Get Valuator

**GPGTVL (*value*)**

**Purpose**

Use **GPGTVL** to retrieve a valuator input value from the current event report. The event is not removed from the current event report until the next invocation of the Await Event (**GPAWEV**) subroutine.

The value returned is in the range found in the valuator record at the time that the device was placed in event mode.

The device to which this value corresponds was identified on the previous invocation of **GPAWEV**.

**Parameters**

*value* **— returned by the graPHIGS API, short floating-point number**
Value of logical valuator when event was triggered.

**Error Codes**

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**519** NO CURRENT EVENT REPORT AVAILABLE

**Related Subroutines**

**GPAWEV**
Await Event

**RCP code**

201336067 (X'0C002503')

## GPGTXP - Get Extended Pick

**GPGTXP (*maxdepth*, *view*, *point*, *modelling*, *depth*, *pickpath*)**

### Purpose

Use **GPGTXP** to retrieve a pick input value from the current event report. The event is not removed from the current event report until the next invocation of the Await Event (**GPAWEV**) subroutine.

This subroutine can be used only for the pick event generated by a pick input device with the extended pick device type. For the pick event generated by a pick input device with the normal pick device type, the Get Pick (**GPGTPK**) subroutine should be used.

### Parameters

*maxdepth* **— specified by user, fullword integer**
Maximum pick path depth to be returned (>=1). This value specifies the size of the array provided in the *pickpath* parameter in the unit of four fullwords. If the actual pick information is longer than the pick path array provided by the application, the overage is truncated.

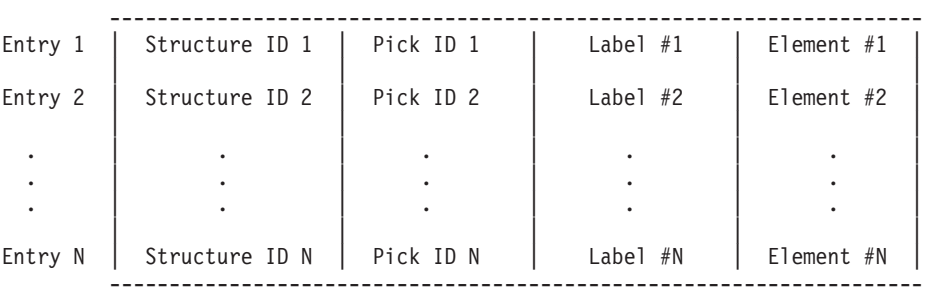
*view* **— returned by the graPHIGS API, fullword integer**
Index of the view where the pick event occurred.

*point* **— returned by the graPHIGS API, 3 short floating-point numbers**
Position of the center of pick aperture in NPC when the pick event occurred. For the normal pick input device such as a tablet, only the *x* value and the *y* value are meaningful. *z* value always contains a constant (0).

*modelling* **— returned by the graPHIGS API, 16 short floating-point numbers**
Composite modelling transformation applied to the picked primitive.

*depth* **— returned by the graPHIGS API, fullword integer**
Depth of the actual pick path returned.

*pickpath* **— returned by the graPHIGS API, array of fullword integers**
Pick path for picked primitive. List of pick path quadruples. Each quadruple represents a structure identifier, pick identifier, label and element number of the picked primitive or an execute structure element of the *pickpath*. The path is returned in the order specified by the Initialize Pick (**GPINPK**) subroutine. Pick path data is set in the *pickpath* parameter in the following order: Entry 1, Entry 2, ...Entry N.



### Error Codes

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**505** LENGTH OF RETURN ARRAY < ZERO

**519** NO CURRENT EVENT REPORT AVAILABLE

### Related Subroutines

**GPAWEV**
Await Event

**GPGTPK**
Get Pick

### RCP code

201336071 (X'0C002507')

## GPGWIN - Get Window

**GPGWIN (*ilen*, *olen*, *data*)**

### Purpose

Use **GPGWIN** to retrieve the workstation event data from the current event report. The event is not removed from the current event report until the next invocation of the Await Event (**GPAWEV**) subroutine.

The data returned in the *data* parameter is defined by the event class of the current event. The event class is returned as an output parameter on **GPAWEV**, or can be obtained using the Inquire Current Event (**GPQCEV**) subroutine.

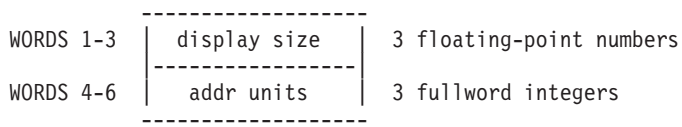
### Parameters

*ilen* **— specified by user, fullword integer**
The length of the data area in bytes provided by the application to return the window event data (>=0).

*olen* **— returned by the graPHIGS API, fullword integer**
The length, in bytes, of the window event data actually returned.

*data* **— returned by the graPHIGS API, variable-length data**
Window event data. The format and contents of the returned data depends on the class of the window event. The supported data are:

**Event Class 105 - Window Resize Notification Event**




**display size - 3 floating-point numbers**
Window size in device coordinates.

**addr units - 3 fullword integers**
Window size in address units. This event is enabled by using the Escape (**GPES**) subroutine with function 1009 (Window Resize Notification Control).

**Event Class 106 - Window Expose Event**


**vflags - Variable length byte string**
View expose flags. There is one flag bit in the byte string for each view on the workstation. The view index can be used to directly access the corresponding flag bit. The maximum number of views supported on workstations is currently 64. Therefore, the maximum number of bytes returned is currently 8 bytes.

Each flag bit can have one of the following values:

- 0 - the view is not effected by the expose event.
- 1 - the view is effected by the expose event.

This event is enabled by using the Escape (**GPES**) subroutine with function 1011 (Window Exposure Notification Control).

### Error Codes

**150** GET FUNCTION DOES NOT MATCH CURRENT EVENT CLASS

**505** LENGTH OF RETURN ARRAY < ZERO

**519** NO CURRENT EVENT REPORT AVAILABLE

### Related Subroutines

**GPES** Escape

### RCP code

201336073 (X'0C002509')

## GPICS - Set Input Character Set

**GPICS (*wsid*, *class*, *device*, *csid*)**

### Purpose

Use **GPICS** to set the character set identifier for the specified input device on the workstation. The identifier specifies the character set used to interpret character data received from or sent to the logical input device.

This subroutine is valid only when the specified device is in Request mode.

In addition, the input character set affects the display of the initial string. For example, this subroutine can be used to specify the language used for string device echo.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*class* **— specified by user, fullword integer**
Device class (1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING).

*device* **— specified by user, fullword integer**
Device number (>=1).

*csid* **— specified by user, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

> **Note:** The supported character set identifier is workstation dependent. See *The graPHIGS Programming Interface: Technical Reference* for details.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**328** INPUT CLASS VALUE IS INVALID

**542** CHARACTER SET IDENTIFIER IS INVALID

**550** CHARACTER SET ID IS NOT SUPPORTED ON WORKSTATION

**Related Subroutines**

**GPQISF**
Inquire Input Character Set Facilities

**RCP code**

201337345 (X'0C002A01')

## GPIDMO - Set Input Device Mode

**GPIDMO (*wsid*, *class*, *device*, *state*, *deact*, *echosw*, *trigger*, *break*, *reset*)**

**Purpose**

Use **GPIDMO** to set the operating mode of the specified logical input device on the specified workstation.

The operating mode consists of six individual switches. Your application can set each of these switches individually to a specified value. Set a switch to zero to preserve its previous setting.

When an application process issues this subroutine and sets the mode of an inactive device to `2=ACTIVE`, the application becomes the owner of the active device. While the device is owned, the graPHIGS API rejects any attempt by any other application process to activate the device. The application process is owner of the device until it performs some action that deactivates the device.

The graPHIGS API sends an input event to the application when the state switch (*state*) of the logical device is set to `2=ACTIVE` and one of the following conditions exists:

- The primary trigger of a logical device fires and the primary trigger switch (*trigger*) is set to `2=ON`. Additionally, if the auto deactivate switch (*deact*) is set to `2=ON`, then the graPHIGS API deactivates the device.
- The workstation's break action fires and its break action switch (*break*) is set to `2=ON`. Additionally, the graPHIGS API deactivates the device.

For a description of the events returned as a result of a trigger firing or a break action, see *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*class* **— specified by user, fullword integer**
Input device class (1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING).

*device* **— specified by user, fullword integer**
Logical input device number (>=1).

*state* **— specified by user, fullword integer**
State switch (1=DEVICE_INACTIVE, 2=DEVICE_ACTIVE). Set this parameter to a value of zero to preserve the current state switch.

*deact* **— returned by the graPHIGS API, fullword integer**
Auto deactivate switch (1=OFF, 2=ON). Set this parameter to a value of zero to preserve the current auto deactivate switch.

When the primary trigger of a logical device fires and an event is sent to the application, the graPHIGS API deactivates the device if its auto deactivate switch is set to 2=ON.

*echosw* **— returned by the graPHIGS API, fullword integer**
Echo switch (1=NOECHO, 2=ECHO).

*trigger* **— returned by the graPHIGS API, fullword integer**
Primary trigger switch (1=OFF, 2=ON). Set this parameter to a value of zero to preserve the current trigger switch.

When the primary trigger of a logical device fires and its trigger switch is set to 2=ON, the graPHIGS API sends an input event to the application.

*break* **— returned by the graPHIGS API, fullword integer**
Break switch (1=OFF, 2=ON). Set this parameter to a value of zero to preserve the current break switch.

When the workstation's break action fires, the graPHIGS API deactivates all active logical devices on the workstation that have their break swtiches set to 2=ON. Additionally, the graPHIGS API sends a break action to the application for each of the devices.

*reset* **— returned by the graPHIGS API, fullword integer**
Auto reset switch (1=OFF, 2=ON). Set this parameter to a value of zero to preserve the current auto reset switch.

When an input event is sent to the application for a logical input device, the graPHIGS API resets its device measure to its initial value if the auto reset switch is set to 2=ON.

**Error Codes**

**3** FUNCTION REQUIRES STATE WSOP

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**327** ECHO SWITCH VALUE IS INVALID

**328** INPUT CLASS VALUE IS INVALID

**329** ONE OF THE SPECIFIED SWITCH VALUES IS INVALID

### Related Subroutines

**GPQID**
Inquire Input Device State

### RCP code

201335814 (X'0C002406')

## GPIEC - Set Input Echo Color

**GPIEC (*wsid*, *color*)**

### Purpose

Use **GPIEC** to change the echo color for all input devices on the specified workstation.

This subroutine is assigned escape identifier 1006.

> **Note:** This subroutine is an escape subroutine, and therefore, may not be available on all workstations. Use the Inquire List of Available Escape Subroutines (**GPQES**) subroutine to determine if this subroutine is supported by a specific workstation.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*color* **— specified by user, 3 short floating-point numbers**
Echo color. This parameter contains three color components which are interpreted in the color space defined by the current workstation color model (0.0<=*color*<=1.0).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**96** COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL

### Related Subroutines

**GPQCML**
Inquire Color Model

**GPQES**
Inquire List of Available Escape Subroutines

### RCP code

201344773 (X'0C004705')

## GPINCH - Initialize Choice

**GPINCH (*wsid*, *device*, *choice*, *echo*, *area*, *datalen*, *data*)**

### Purpose

Use **GPINCH** to specify initial values for a given choice device.

The Initialize Choice subroutine stores the initial choice number, prompt/echo type, echo area, and data record in the workstation state list for the specified device. For a keyboard choice device, an initial choice number less than 256 is interpreted using the workstation's input device character set. For details on the specific devices available on different workstation types, see *The graPHIGS Programming Interface: Technical Reference* or use the appropriate inquiry subroutines.

**Note:** The choice device must be in Request mode.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Choice device number (>=1).

*choice* **— specified by user, fullword integer**
Initial choice number (>=0).

*echo* **— specified by user, fullword integer**
Prompt/echo type (>=1).

*area* **— specified by user, 6 short floating-point numbers (DC)**
Echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*datalen* **— specified by user, fullword integer**
Data record length.

**Note:** The data record length parameter must equal zero (*datalen*=0) if no data record is required for the given prompt/echo type. For prompt/echo types which require a data record, the specified data record length must be greater than, or equal to, 12.

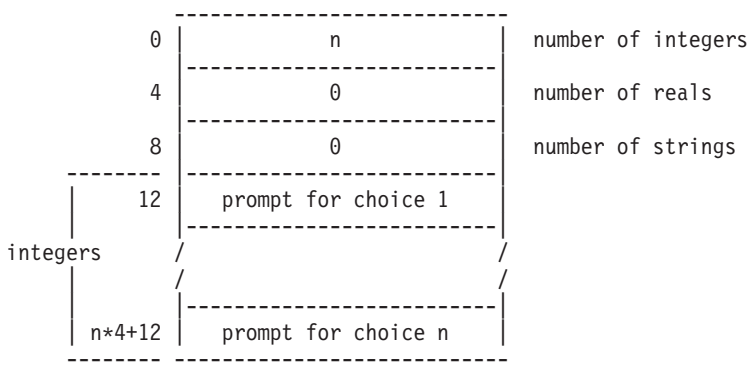
*data* **— user specified, variable data**
Choice data record.

**Prompt/echo Types**

**Type One**
designates the current choice number using a workstation-dependent technique.

**Type Two**
lets you indicate choice numbers by invoking the prompting capability.

Physical input devices commonly used as choice logical input devices have a built-in prompting capability. If the value of the i(th) element of the prompt array in the choice data record is `1=OFF`, prompting of the i(th) alternative of the specified choice input device is turned off. An value of `2=ON` indicates that prompting for that alternative is turned on. Choice echo Type Two requires the following data record:



Prompt values are 1=OFF, 2=ON.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**144** PROMPT/ECHO TYPE NOT AVAILABLE ON WORKSTATION

**145** ECHO AREA BOUNDARY VALUE IN ERROR

**146** FIELD IN INPUT DEVICE DATA RECORD IN ERROR

**152** INITIAL CHOICE VALUE < ZERO OR IS INVALID

**324** PROMPT/ECHO TYPE < ONE

**501** DATA RECORD WAS NOT SPECIFIED BUT IS REQUIRED

**502** FIELD IN DATA RECORD NOT SUPPORTED ON WORKSTATION

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**Related Subroutines**

**GPCHMO**
Set Choice Mode

**GPQCH**
Inquire Choice Device State

**GPQDCH**
Inquire Default Choice Device Data

**GPQDS**
Inquire Maximum Display Surface Size

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201335300 (X'0C002204')

## GPINLC - Initialize Locator

**GPINLC (*wsid*, *device*, *view*, *pos*, *echo*, *area*, *datalen*, *data*)**

**Purpose**

Use **GPINLC** to initialize the specified locator device.

The Initialize Locator subroutine stores the initial locator position, initial view index, prompt/echo type, echo area, and locator data record in the workstation state list for the specified device.

Two positions are required for some locator prompt/echo types: the initial locator position, which remains fixed during input operation, and the current locator position, which varies dynamically as you use the locator.

**Note:** The locator device must be in Request mode.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Locator device number (>=1).

*view* **— specified by user, fullword integer**
Initial view index (>=0)

*pos* **— specified by user, 3 short floating-point numbers (WC)**
Initial locator position.

*echo* **— specified by user, fullword integer**
Prompt/echo type (>=1).

*area* **— specified by user, 6 short floating-point numbers (DC)**
Echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*datalen* **— specified by user, fullword integer**
Data record length.

> **Note:** The data record length parameter must equal zero (*datalen*=0) if no data record is required for the given prompt/echo type. For prompt/echo types which require a data record, the specified data record length must be greater than, or equal to, 12.

*data* **— specified by user, variable length data**
Locator data record.

**Prompt/echo Types**

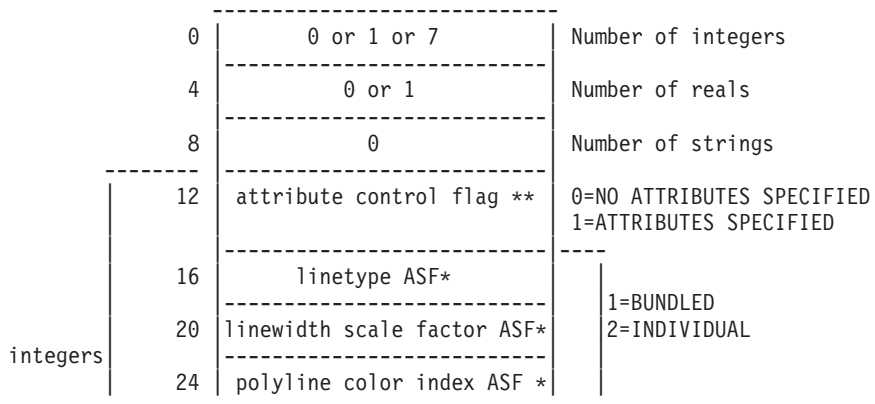
**Type One**
designates the current position of the locator using a workstation-dependent technique.

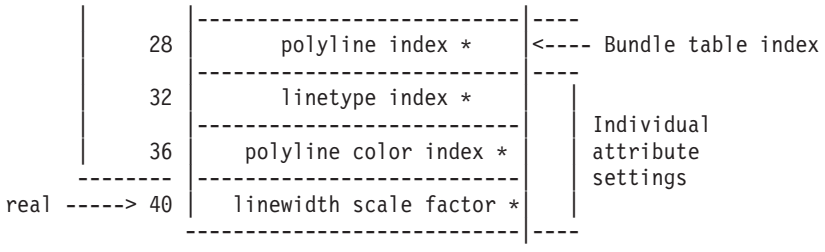
**Type Two**
the crosshair, designates the current position of the locator by spanning the display surface or device echo area with both a vertical and a horizontal line. The lines intersect at the current locator position. Whether the crosshair spans the entire display surface or only the echo area depends on the capabilities of the workstation.

**Type Three**
designates the current position of the locator using a tracking cross.

**Type Four**
designates the current position of the locator using a rubber band line connecting the initial locator position given by this subroutine and the current locator position. If no attributes are specified within the data record, workstation dependent defaults are used.

Type Four requires the following data record:



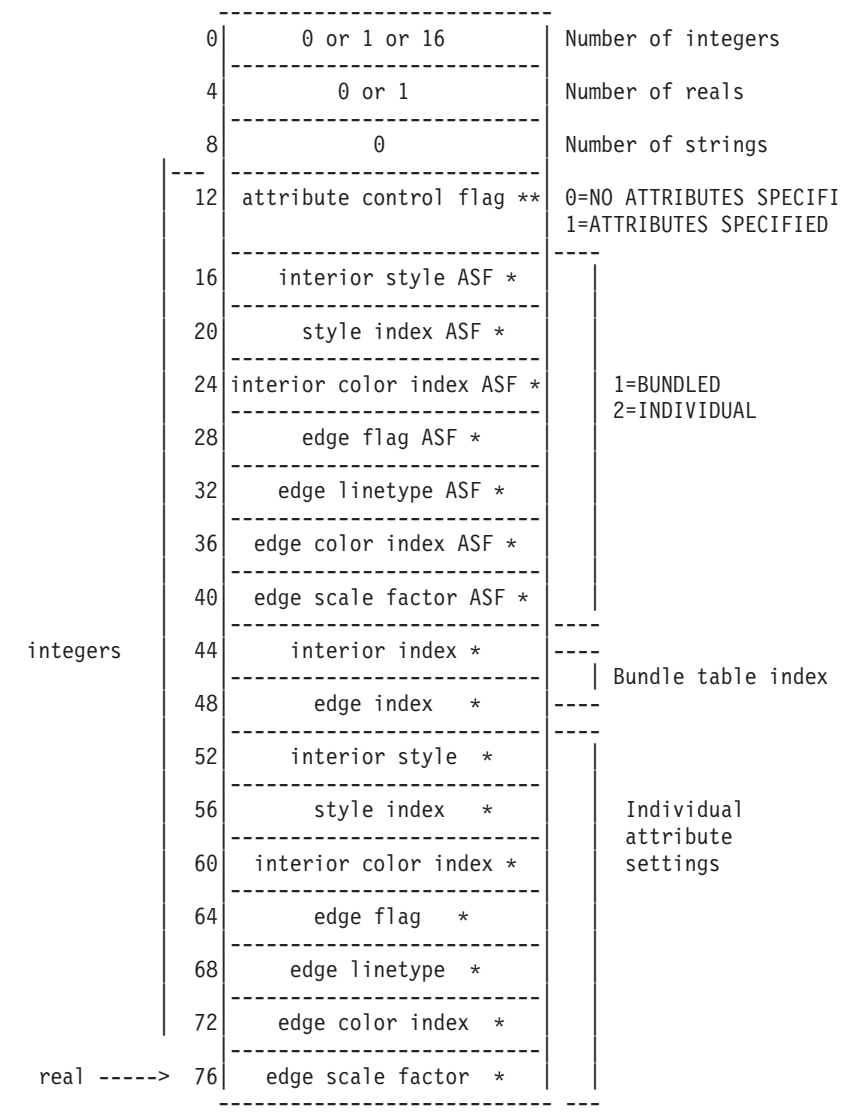



```
*  Polyline Attributes
Only present if attribute control flag=1
**  Attribute Control Flag
Only present if number of integers >0
```

**Type Five**

designates the current position of the locator using a rubberband rectangle. The diagonal of the rectangle forms a line connecting the initial locator position given by this subroutine and the current locator position. If no attributes are specified within the data record, workstation dependent defaults are used.

Type Five requires the following data record:




```
*  Polygon Attributes
```

```
Only present if attribute control flag=1
**  Attribute Control Flag
Only present if number of integers >0
```

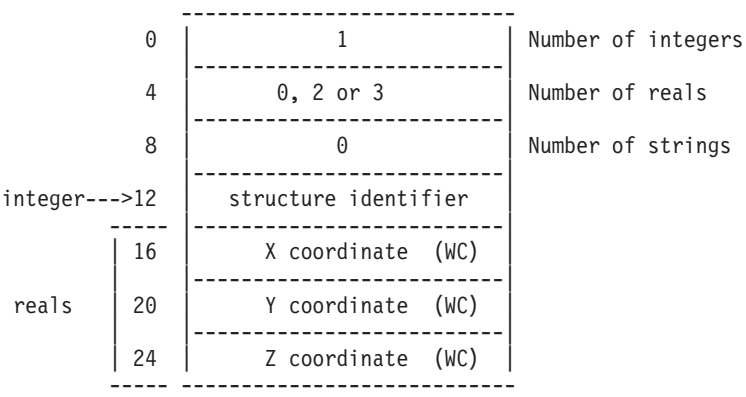
**Type Seven**

designates the current position of the locator using a specified structure network attached to the tracking cross.

The structure identifier of the root structure to be dragged is specified in the data record and, optionally, the World Coordinate (WC) point on the structure to which the locator is attached. If the attachment point is not specified, the point (0, 0, 0) is used. If for the attachment point, only *x*, *y* coordinates are specified, a *z* coordinate of zero is used. The structure to be dragged must be associated with the workstation before it can be attached to the locator device.

The structure is conceptually dragged in Viewing Coordinate (VC) space, before the view projection is applied. The structure definition may vary while the locator is active. For example, the application may choose to scale the structure as it is dragged using a modeling transformation. The dragged structure network is clipped to the locator device echo area.

Type Seven requires the following data record:




**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**144** PROMPT/ECHO TYPE NOT AVAILABLE ON WORKSTATION

**145** ECHO AREA BOUNDARY VALUE IN ERROR

**146** FIELD IN INPUT DEVICE DATA RECORD IN ERROR

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**324** PROMPT/ECHO TYPE < ONE

**501** DATA RECORD WAS NOT SPECIFIED BUT IS REQUIRED

**502** FIELD IN DATA RECORD NOT SUPPORTED ON WORKSTATION

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**Related Subroutines**

**GPLCMO**

Set Locator Mode

**GPQDLC**
Inquire Default Locator Device Data

**GPQDS**
Inquire Maximum Display Surface Size

**GPQLC**
Inquire Locator Device State

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201335297 (X'0C002201')

## GPINPK - Initialize Pick

**GPINPK (*wsid*, *device*, *depth*, *pickpath*, *echo*, *area*, *datalen*, *data*, *order*)**

### Purpose

Use **GPINPK** to initialize the specified pick device.

The Initialize Pick subroutine stores the prompt/echo type, echo area, initial pick path depth, initial pick path, pick data record and pick path order in the workstation state list for the specified device.

**Note:** The pick device must be in Request Mode.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Pick device number (>=1).

*depth* **— specified by user, fullword integer**
Initial pick path depth (>=0).

*pickpath* **— specified by user, array of fullword integers**
Initial pick path. Specified as a list of triplets where each triplet represents the structure identifier, pick identifier, and element number of the initial path.

Pick path data is set in the *pickpath* parameter in the following order: Entry 1, Entry 2, ...Entry N.

| | | | |
|---|---|---|---|
| Entry 1 | Structure ID 1 | Pick ID 1 | Element #1 |
| Entry 2 | Structure ID 2 | Pick ID 2 | Element #2 |
| . | . | . | . |
| . | . | . | . |
| . | . | . | . |
| Entry N | Structure ID N | Pick ID N | Element #N |

**Note:** The initial pick path is ignored.

*echo* **— specified by user, fullword integer**
Prompt/echo type (>=1).

*area* **— specified by user, 6 short floating-point numbers (DC)**
Echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*datalen* **— specified by user, fullword integer**
Data record length in bytes (>=0).

> **Note:** The data record length parameter must equal zero (*datalen*=0) if no data record is required for the given prompt/echo type. For prompt/echo types which require a data record, the specified data record length must be greater than or equal to 12.

*data* **— specified by user, variable data**
Pick data record.

*order* **— specified by user, fullword integer**
Pick path order for returning subsequent pick input values (`1=TOP_FIRST, 2=BOTTOM_FIRST`). The default value is `1=TOP_FIRST`.

**Prompt/echo Types**

**Type One**
uses a workstation-dependent technique that highlights the picked primitive. No data record is required.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**144** PROMPT/ECHO TYPE NOT AVAILABLE ON WORKSTATION

**145** ECHO AREA BOUNDARY VALUE IN ERROR

**146** FIELD IN INPUT DEVICE DATA RECORD IN ERROR

**156** PICK PATH ORDER IS INVALID

**158** INVALID ELEMENTS IN THE INITIAL PICK PATH

**324** PROMPT/ECHO TYPE < ONE

**502** FIELD IN DATA RECORD NOT SUPPORTED ON WORKSTATION

**506** NUMBER OF INITIAL VALUES < ZERO

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**513** NUMBER OF INITIAL VALUES EXCEEDS DEVICE MAXIMUM

**Related Subroutines**

**GPADCN**
Add Class Name to Set

**GPPKF**
Set Pick Filter

**GPPKID**
Set Pick Identifier

**GPPKMO**
Set Pick Mode

**GPRCN**
Remove Class Name from Set

**GPQDPK**
Inquire Default Pick Device Data

**GPQDS**
Inquire Maximum Display Surface Size

**GPQLI**
Inquire List of Logical Input Devices

**GPQPK**
Inquire Pick Device State

**RCP code**

201335301 (X'0C002205')

## GPINSK - Initialize Stroke

**GPINSK (*wsid*, *device*, *view*, *npoint*, *width*, *pointlist*, *echo*, *area*, *buflen*, *editpos*, *datalen*, *data*)**

**Purpose**

Use **GPINSK** to initialize the specified stroke device.

The Initialize Stroke subroutine stores the initial stroke, initial view index, prompt/echo type, echo area and stroke data record in the workstation state list for the specified device.

For all prompt/echo types, the input buffer size is compared to the maximum input buffer size for stroke devices on that workstation. If the requested buffer size is greater, the maximum buffer size for stroke devices is substituted in the stored record. If the initial stroke is longer than the buffer size, an error is issued.

When a stroke measure process begins, it acquires a buffer of the current input buffer size. The initial stroke pointlist is copied into this buffer, and the editing position is placed at the initial buffer editing position. The replacement of points begins at this initial position. The *x*, *y*,*z* and time intervals (where possible) of the data record control the frequency and density of stroke points.

**Note:** The stroke device must be in Request mode.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Stroke device number (>=1).

*view* **— specified by user, fullword integer**
Initial view index (>=0).

*npoint* **— specified by user, fullword integer**
Number of points in initial stroke (>=0).

*width* **— specified by user, fullword integer**
Number of fullwords between subsequent *x* values in the initial pointlist (>=3).

*pointlist* **— specified by user, array of short floating-point numbers (WC)**
Coordinates of initial stroke points specified as a list of three-dimensional points.

*echo* **— specified by user, fullword integer**
Prompt/echo type (>=1).

*area* **— specified by user, 6 short floating-point numbers (DC)**
Echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*buflen* **— specified by user, fullword integer**
Stroke input buffer size (>=0). This value is specified as the maximum number of points the user may enter into the stroke input buffer.

*editpos* **— specified by user, fullword integer**
Editing position within the initial stroke buffer (1<=*editpos*<= *npoint +1*).

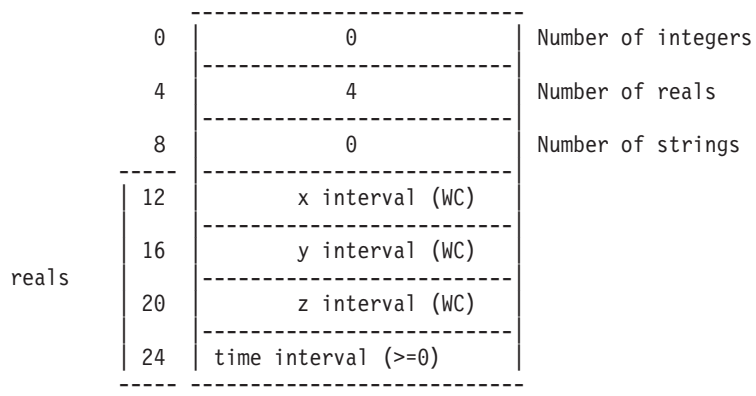
*datalen* **— specified by user, fullword integer**
Data record length, in bytes (>=0).

> **Note:** The data record length parameter must equal zero (*datalen*=0) if no data record is required for the given prompt/echo type. For prompt/echo types which require a data record, the specified data record length must be greater than, or equal to, 12.

*data* **— specified by user, variable data**
Stroke data record.

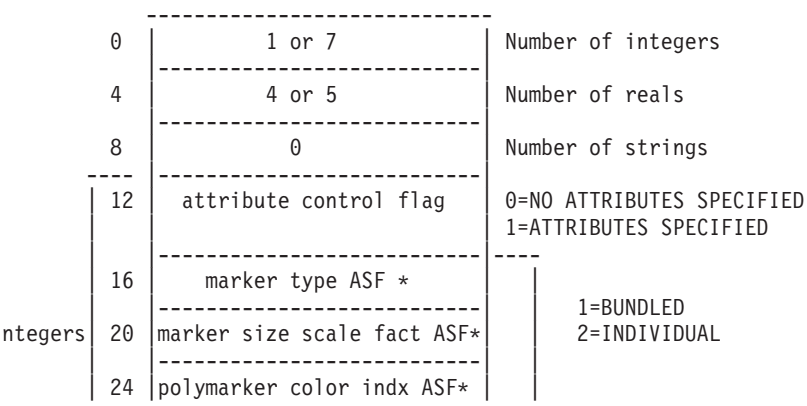
### Prompt/echo Types

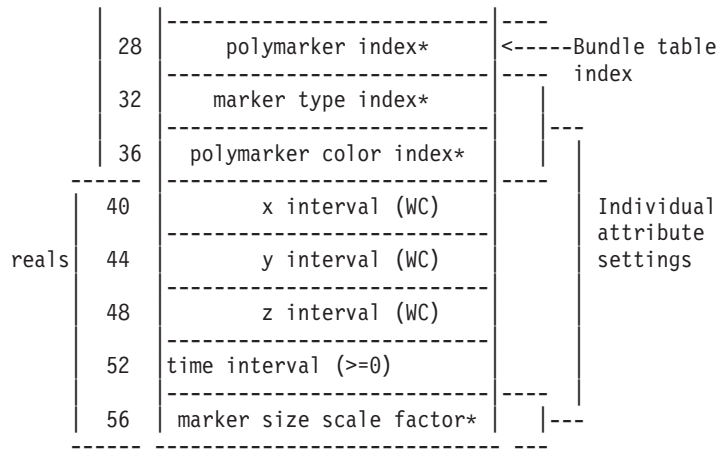
**Type One**
displays the current stroke using a workstation-dependent technique. Stroke echo Type One requires the following data record:




**Type Three**
displays a marker at each point of the current stroke. The marker representation is selected by a marker index entry, which is stored in the stroke data record.

Stroke echo Type Three requires the following data record:

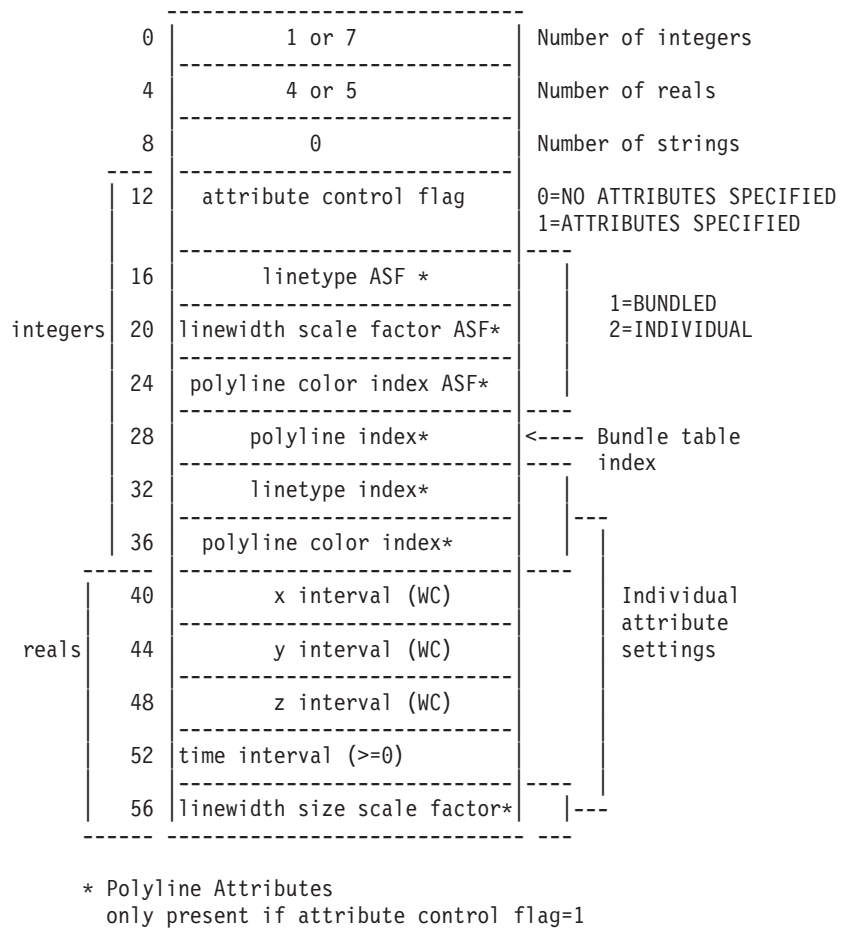





**Type Four**

displays a line joining successive points in the current stroke. A polyline index entry in the stroke data record selects the line representation used.

Stroke echo Type Four requires the following data record:




**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**144** PROMPT/ECHO TYPE NOT AVAILABLE ON WORKSTATION

**145** ECHO AREA BOUNDARY VALUE IN ERROR

**146** FIELD IN INPUT DEVICE DATA RECORD IN ERROR

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**324** PROMPT/ECHO TYPE < ONE

**325** NUMBER OF POINTS IN INITIAL STROKE < ZERO

**501** DATA RECORD WAS NOT SPECIFIED BUT IS REQUIRED

**502** FIELD IN DATA RECORD NOT SUPPORTED ON WORKSTATION

**506** NUMBER OF INITIAL VALUES < ZERO

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**513** NUMBER OF INITIAL VALUES EXCEEDS DEVICE MAXIMUM

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**577** BUFFER LENGTH IS < ZERO

**578** BUFFER LENGTH EXCEEDS DEVICE MAXIMUM

**579** INITIAL POSITION IS < ONE OR > NUMBER OF INITIAL VALUES PLUS ONE

**580** INITIAL POSITION EXCEEDS BUFFER SIZE

### Related Subroutines

**GPQDS**
Inquire Maximum Display Surface Size

**GPQDSK**
Inquire Default Stroke Device Data

**GPQLI**
Inquire List of Logical Input Devices

**GPQSK**
Inquire Stroke Device State

**GPSKMO**
Set Stroke Mode

### RCP code

201335298 (X'0C002202')

## GPINST - Initialize String

**GPINST (*wsid*, *device*, *length*, *string*, *echo*, *area*, *buflen*, *cursor*, *datalen*, *data*)**

### Purpose

Use **GPINST** to initialize the specified string input device.

The Initialize String subroutine stores the initial string, prompt/echo type, echo area, and string data record in the workstation state list for the specified device. The string device's input character set is used to interpret the initial string and prompt strings.

The input string returned is the size of the string input buffer (*buflen*). However, if your application specifies a prompt/echo type 2 (*echo*) and the *buflen* size *plus* the prompt are greater than the device's maximum buffer size, then the input string returned will be the device's maximum buffer size *minus* the prompt string.

**Note:** The string device must be in Request mode.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier of the device to be initialized.

*device* **— specified by user, fullword integer**
String device number (>=1).

*length* **— specified by user, fullword integer**
Length of initial string, in bytes (>=0).

*string* **— specified by user, variable length character string**
Initial string.

*echo* **— specified by user, fullword integer**
Prompt/echo type to be initialized (>=1).

*area* **— specified by user, 6 short floating-point numbers (DC)**
Echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*buflen* **— specified by user, fullword integer**
String input buffer size, in bytes (>=0).

*cursor* **— specified by user, fullword integer**
Initial cursor position in the string buffer relative to the position of the initial string (>=1). If a prompt string is used (echo type=2), the initial string is placed in the string buffer immediately after the prompt string. Otherwise, the initial string is placed at position one in the string buffer.

If the string input buffer size parameter is zero, the initial cursor position parameter is not checked, and the graPHIGS API uses the value of zero for the initial position.

*datalen* **— specified by user, fullword integer**
Data record length, in bytes (>=0).

**Note:** The data record length parameter must equal zero (*datalen*=0) if no data record is required for the given prompt/echo type. For prompt/echo types which require a data record, the specified data record length must be greater than or equal to 12.

*data* **— specified by user, variable data**
String data record.

For all prompt/echo types the input buffer size is compared to maximum input buffer size for string devices in the WDT. If the requested buffer size is greater, the maximum input buffer size for string devices is substituted in the stored record. If the initial string is longer than the buffer size, an error is issued.

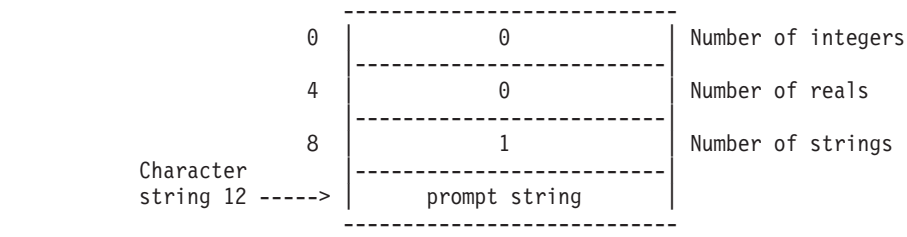
**Prompt/echo Types**

**Type One**
displays the current string value within the echo area using a workstation-dependent technique.

**Type Two**
displays an application-specified prompt string which precedes the initial string. This prompt string

is passed in by way of the data record and is placed in the string device input buffer. It is not returned to the user. The prompt may not be typed over and is not returned with the string device input. The initial cursor position is specified relative to the initial string position in the string buffer.

String echo Type Two requires the following data record:




The prompt string includes the length of the string in the first byte of the input. This length (in bytes) is inclusive of the length field. Therefore, a length of one specifies a null prompt string.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**144** PROMPT/ECHO TYPE NOT AVAILABLE ON WORKSTATION

**145** ECHO AREA BOUNDARY VALUE IN ERROR

**146** FIELD IN INPUT DEVICE DATA RECORD IN ERROR

**324** PROMPT/ECHO TYPE < ONE

**501** DATA RECORD WAS NOT SPECIFIED BUT IS REQUIRED

**502** FIELD IN DATA RECORD NOT SUPPORTED ON WORKSTATION

**506** NUMBER OF INITIAL VALUES < ZERO

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**513** NUMBER OF INITIAL VALUES EXCEEDS DEVICE MAXIMUM

**577** BUFFER LENGTH IS < ZERO

**578** BUFFER LENGTH EXCEEDS DEVICE MAXIMUM

**579** INITIAL POSITION IS < ONE OR > NUMBER OF INITIAL VALUES PLUS ONE

**580** INITIAL POSITION EXCEEDS BUFFER SIZE

**Related Subroutines**

**GPQDS**
Inquire Maximum Display Surface Size

**GPQDST**
Inquire Default String Device Data

**GPQLI**
Inquire List of Logical Input Devices

**GPQST**
Inquire String Device State

**GPSTMO**
Set String Mode

**RCP code**

201335302 (X'0C002206')

# GPINVL - Initialize Valuator

**GPINVL (*wsid*, *device*, *ivalue*, *echo*, *area*, *lovalue*, *hivalue*, *datalen*, *data*)**

### Purpose

Use **GPINVL** to initialize the specified valuator device.

The Initialize Valuator subroutine stores the initial value, prompt/echo type, echo area, and valuator data record in the WSL entry for the specified workstation.

For all valuator prompt/echo types a low value and a high value specify the range for input from that valuator. The values from the physical device are scaled linearly to the specified range.

> **Note:** The valuator device must be in Request mode.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier of the device to be initialized.

*device* **— specified by user, fullword integer**
Valuator device number to be initialized (>=1).

*ivalue* **— specified by user, short floating-point number**
Initial value.

*echo* **— specified by user, fullword integer**
Prompt/echo type (>=1).

*area* **— specified by user, 6 short floating-point numbers (DC)**
Echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*lovalue* **— specified by user, short floating-point number**
Low end of range for valuator.

*hivalue* **— specified by user, short floating-point number**
High end of range for valuator.

*datalen* **— specified by user, fullword integer**
Data record length, in bytes (>=0)

> **Note:** The data record length parameter must equal zero (*datalen*=0) if no data record is required for the given prompt/echo type. For prompt/echo types which require a data record, the specified data record length must be greater than, or equal to, 12.

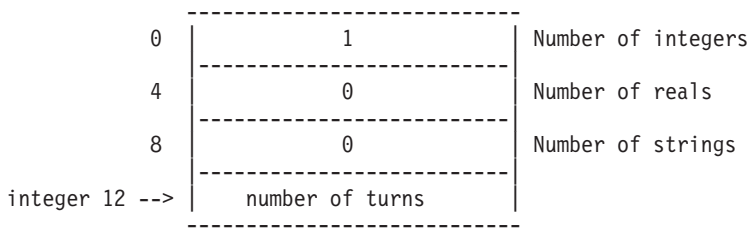
*data* **— specified by user, variable data**
Valuator data record.

### Prompt/echo Types

- **Type One** designates the current valuator value using a workstation-dependent technique.
- **Type Three** displays a digital representation of the current valuator value within the echo area.

- **Type Four** passes a data record containing the number of turns of the dial to be mapped onto the specified valuator range. The current valuator value is designated using a workstation-dependent technique.

  Valuator echo Type Four requires the following data record:




  **Note:** For emulation purposes, one turn of the dial equals one unit of physical motion.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**144** PROMPT/ECHO TYPE NOT AVAILABLE ON WORKSTATION

**145** ECHO AREA BOUNDARY VALUE IN ERROR

**146** FIELD IN INPUT DEVICE DATA RECORD IN ERROR

**324** PROMPT/ECHO TYPE < ONE

**501** DATA RECORD WAS NOT SPECIFIED BUT IS REQUIRED

**502** FIELD IN DATA RECORD NOT SUPPORTED ON WORKSTATION

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**511** INVALID VALUATOR RANGE

**515** INITIAL VALUATOR VALUE NOT WITHIN RANGE

**Related Subroutines**

**GPQDVL**
Inquire Default Valuator Device Data

**GPQVL**
Inquire Valuator Device State

**GPVLMO**
Set Valuator Mode

**RCP code**

201335299 (X'0C002203')

## GPIPKC - Set Initial Pick Correlation State

**GPIPKC (*wsid*, *device*, *state*)**

**Purpose**

Use **GPIPKC** to change the initial pick correlation state of the pick device.

Similar to other input device initialization subroutines, **GPIPKC** can only be called when the device is in Request mode, and takes effect the next time the device becomes active.

This subroutine is assigned escape identifier 1004.

> **Note:** This subroutine is an escape subroutine and therefore may not be available on all workstations. Use the Inquire List of Available Escape Subroutines (**GPQES**) subroutine, to determine if this subroutine is supported by a specific workstation.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Device number (>=1).

*state* **— specified by user, fullword integer**
Initial pick correlation state (`1=OFF, 2=ON`).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**166** INITIAL PICK CORRELATION STATE IS INVALID

### Related Subroutines

**GPQES**
Inquire List of Available Escape Subroutines

**GPPKMO**
Set Pick Mode

### RCP code

201337349 (X'0C002A05')

## GPIT - Set Input Device Trigger

**GPIT (*wsid*, *class*, *devnum*, *listid*, *tnum*, *triglist*)**

### Purpose

Use **GPIT** to set the input device trigger for a given input device at the specified workstation. The specified trigger list replaces the current trigger list for the input device. Similar to other input device initialization subroutines, **GPIT** can only be called when the device is in Request mode, and takes effect the next time the device becomes active.

The primary trigger (trigger list identifier zero) always exists and causes the input to be returned to the application. Secondary triggers (trigger list identifiers beginning with 1) provide the user additional control of the device measure. For example, secondary trigger 3, for stroke device #2 on the 5080 workstation,

causes a point to be added to the input buffer. The number of secondary triggers for a given input device can be determined programmatically using the Inquire Number of Secondary Triggers (**GPQNST**) subroutine.

Not all input devices let the trigger be programmed. This information, as well as the supported trigger types and qualifiers, can be determined using the Inquire Input Trigger Capabilities (**GPQIT**) subroutine.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*class* — **specified by user, fullword integer**
Input device class (1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING).

*devnum* — **specified by user, fullword integer**
Device number (>0).

*listid* — **specified by user, fullword integer**
Trigger list identifier (>=0).

*tnum* — **specified by user, fullword integer**
Number of entries in the trigger list which follows (>=0). The primary trigger list must always contain at least one entry. A secondary trigger may have an empty trigger list (zero entries), in which case the trigger is always inoperative.

*triglist* — **specified by user, array of fullword integers**
List of trigger descriptor triplets. The list is an array of trigger descriptors in which each descriptor consists of 3 fullword integers designating the trigger type, low trigger qualifier, and high trigger qualifier. The trigger type field has the following meanings:

| Type | Meaning |
|---|---|
| **-1** | Change of the measure of the corresponding physical input device.<br><br>This type is valid only for the primary (0) trigger list identifier.<br><br>The low qualifier specifies the granularity of movement which causes the trigger to fire. The granularity is specified as the amount that the physical device measure must change since the last trigger was fired in order for the trigger to be fired again. The high qualifier must be zero. |
| **0** | Reserved for an implementation dependent trigger that is only valid as the default value. This value cannot be specified. |
| **>0** | Physical device number within the button category.<br><br>The trigger qualifiers for this trigger type are a range of choices on the indicated physical device. Trigger qualifiers less than 256 for keyboard trigger types are interpreted using the device's input character set. The parameter *tnum* identifies the number of triplets (trigger descriptors) in the trigger list. |

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**328** INPUT CLASS VALUE IS INVALID

**565** WARNING, A TRIGGER QUALIFIER VALUE IS INVALID

**567** A TRIGGER TYPE VALUE IS INVALID

**568** A TRIGGER QUALIFIER VALUE IS INVALID

**569** DEVICE DOES NOT SUPPORT PROGRAMMABLE TRIGGERS

**570** SPECIFIED TRIGGER LIST IDENTIFIER DOES NOT EXIST

**574** RANGE INVALID, LOW VALUE EXCEEDS HIGH VALUE

**575** NUMBER OF ENTRIES IN TRIGGER LIST IS INVALID

**576** PRIMARY TRIGGER LIST MUST HAVE AT LEAST ONE ENTRY

**595** A TRIGGER TYPE IS INCOMPATIBLE WITH THE TRIGGER LIST IDENTIFIER

**Related Subroutines**

**GPQDIT**
Inquire Default Input Device Triggers

**GPQIT**
Inquire Input Trigger Capabilities

**GPQNST**
Inquire Number of Secondary Triggers

**RCP code**

201340417 (X'0C003601')

## GPLCMO - Set Locator Mode

**GPLCMO (*wsid*, *device*, *mode*, *echosw*)**

**Purpose**

Use **GPLCMO**to set the operating mode of the specified locator device.

After the Locator Mode is set, its echoing state is set to 1=NOECHO or 2=ECHO, based on the *echosw* parameter. Depending on the specified operating mode, 1=REQUEST, 2=SAMPLE, or 3=EVENT, an interaction with the given device may begin or end.

> **Note:** The input device is reset with the initialization values when the **GPLCMO** subroutine is called with *mode* parameters set to SAMPLE or EVENT.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Locator device number (>=1).

*mode* **— specified by user, fullword integer**
Operating mode (1=REQUEST, 2=SAMPLE, 3=EVENT).

*echosw* **— specified by user, fullword integer**
Echo switch (1=NOECHO, 2=ECHO).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**326** OPERATING MODE IS INVALID

**327** ECHO SWITCH VALUE IS INVALID

### Related Subroutines

**GPQLI**
Inquire List of Logical Input Devices

### RCP code

201335809 (X'0C002401')

## GPPDMO - Set Physical Device Mode

**GPPDMO (*wsid*, *category*, *device*, *mode*)**

### Purpose

Use **GPPDMO** to enable or disable the physical input device from generating input values to the set of logical input devices to which it is connected.

The default state of all physical devices is `2=ENABLED`, if the physical device is actually present. The set of physical devices that are connected to a specific logical input device can be obtained by issuing the Inquire Source Physical Device (**GPQSPD**) subroutine.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*category* **— specified by user, fullword integer**
Physical device category (`1=BUTTON, 2=SCALAR, 3=2D_VECTOR`).

*device* **— specified by user, fullword integer**
Physical device number (>=1).

*mode* **— specified by user, fullword integer**
Physical device mode (`1=DISABLED, 2=ENABLED`).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**160** PHYSICAL INPUT DEVICE CATEGORY IS INVALID

**163** PHYSICAL DEVICE MODE IS INVALID

**169** PHYSICAL INPUT DEVICE CANNOT BE DISABLED

### Related Subroutines

**GPEPD**
Emulate Physical Device

**GPQPDC**
Inquire Physical Device Characteristics

**GPQSPD**
Inquire Source Physical Device

**RCP code**

201344769 (X'0C004701')

## GPPKAP - Set Pick Aperture

**GPPKAP (*wsid*, *device*, *size*)**

**Purpose**

Use **GPPKAP** to set the pick aperture size for the specified pick device.

The aperture is a square the length of whose side is specified by the *size* parameter in Device Coordinates (DC).

**Note:** The pick device must be in Request mode.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier of pick device.

*device* **— specified by user, fullword integer**
Pick device number (>=1)

*size* **— specified by user, short floating-point number (DC)**
Aperture size (>=0)

This is specified as the length of a side of a square in Device Coordinates.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**554** PICK APERTURE < ZERO

**Related Subroutines**

**GPPKMO**
Set Pick Mode

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201337348 (X'0C002A04')

## GPPKF - Set Pick Filter

**GPPKF (*wsid*, *device*, *inclen*, *incl*, *exclen*, *excl*)**

### Purpose

Use **GPPKF** to set the pick inclusion and exclusion filters for the given pick device.

The filters consist of class names which indicate to the specified workstation which class names to include and which to exclude from pickability (detectability).

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Pick device number (>=1).

*inclen* **— specified by user, fullword integer**
Inclusion filter list length (>=0).

*incl* **— specified by user, array of fullword integers**
List of class names (>=0).

*exclen* **— specified by user, fullword integer**
Exclusion fliter list length (>=0).

*excl* **— specified by user, array of fullword integers**
List of class names (>=0).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**321** FILTER VALUE IS INVALID

**531** FILTER LIST LENGTH < ZERO

### Related Subroutines

**GPADCN**
Add Class Name to Set

**GPQLI**
Inquire List of Logical Input Devices

**GPRCN**
Remove Class Name from Set

### RCP code

201335041 (X'0C002101')

## GPPKMO - Set Pick Mode

**GPPKMO (*wsid*, *device*, *mode*, *echosw*)**

### Purpose

Use **GPPKMO** to set the operating mode of the specified pick device.

After the pick mode is set, its echoing state is set to 1=NOECHO or 2=ECHO based on the *echosw* parameter. Depending on the specified operating mode, 1=REQUEST, 2=SAMPLE, or 3=EVENT, an interaction with the given device may begin or end.

> **Note:** The input device is reset with the initialization values when the **GPPKMO** subroutine is called with *mode* parameters set to SAMPLE or EVENT.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Pick device number (>=1)

*mode* **— specified by user, fullword integer**
Operating mode (1=REQUEST, 2=SAMPLE, 3=EVENT)

*echosw* **— specified by user, fullword integer**
Echo switch (1=NOECHO, 2=ECHO)

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**326** OPERATING MODE IS INVALID

**327** ECHO SWITCH VALUE IS INVALID

### Related Subroutines

**GPQLI**
Inquire List of Logical Input Devices

### RCP code

201335813 (X'0C002405')

## GPPKSC - Set Pick Selection Criteria

**GPPKSC (*wsid*, *device*, *criteria*)**

### Purpose

Use **GPPKSC** to set the criteria that is used to select the primitive(s) that are to be accumulated during pick correlation.

As with other input device control subroutines, **GPPKSC** can only be called when the specified input device is in Request mode.

If a workstation does not support hidden line/hidden surface removal, then the graPHIGS API ignores the visible aspect of the criteria. Therefore, criteria 4-6 behaves as criteria 1-3.

This subroutine is assigned escape identifier 1005.

> **Note:** This subroutine is an escape subroutine and therefore may not be available on all workstations. Use the Inquire List of Available Escape Subroutines (**GPQES**) subroutine to determine if this subroutine is supported by a specific workstation. See also the workstation description information in *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Device number (>=1).

*criteria* **— specified by user, fullword integer**
Pick selection criteria (`1=FIRST`, `2=LAST`, `3=ALL`, `4=FIRST_VISIBLE`, `5=LAST_VISIBLE`, `6=ALL_VISIBLE`). `1=FIRST` and `2=LAST` refer to the order in which views, structures and elements are traversed for output.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**167** PICK SELECTION CRITERIA IS INVALID

### Related Subroutines

**GPPKMO**
Set Pick Mode

**GPQES**
Inquire List of Available Escape Subroutines

### RCP code

201337350 (X'0C002A06')

## GPRQCH - Request Choice

**GPRQCH (*wsid*, *device*, *status*, *choice*)**

### Purpose

Use **GPRQCH** to have the graPHIGS API execute a request to the specified choice device. The choice input value, which is the current measure of the choice device, is returned.

A status of `1=NONE` means that a break action occurred.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Choice device number (>=1).

*status* **— returned by the graPHIGS API, fullword integer**
Status (1=NONE, 2=OK).

*choice* **— returned by the graPHIGS API, fullword integer**
Choice number.

A choice number of zero means no choice.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**Related Subroutines**

**GPCHMO**
Set Choice Mode

**GPINCH**
Initialize Choice

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201335562 (X'0C00230A')

## GPRQLC - Request Locator

**GPRQLC (*wsid*, *device*, *status*, *view*, *pos*)**

**Purpose**

Use **GPRQLC** to have the graPHIGS API execute a request to the specified locator device.

The locator position and the index of the view whose matrix was used to convert the location to World Coordinates (WC) are returned.

Locator input is returned from the view active for input with the highest input priority under the cursor. View 0 is the highest priority view, unless modified by your application.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Locator device number (>=1).

*status* **— returned by the graPHIGS API, fullword integer**
Status (1=NONE, 2=OK).

*view* — **returned by the graPHIGS API, fullword integer**
View index.

*pos* — **returned by the graPHIGS API, 3 short floating-point numbers (WC)**
Locator position.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**Related Subroutines**

**GPINLC**
Initialize Locator

**GPLCMO**
Set Locator Mode

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201335559 (X'0C002307')

## GPRQPK - Request Pick

**GPRQPK (*wsid*, *device*, *length*, *status*, *depth*, *pickpath*)**

**Purpose**

Use **GPRQPK** to have the graPHIGS API execute a request to the specified pick device.

The pick path information is returned in the order specified in the Initialize Pick (**GPINPK**) subroutine, that is, `1=TOP_FIRST` or `2=BOTTOM_FIRST`. If **GPINPK** has not been called, the default value is `1=TOP_FIRST`.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*device* — **specified by user, fullword integer**
Pick device number (>=1).

*length* — **specified by user, fullword integer**
Length of pick path array provided by the application in which the graPHIGS API returns the pick information (>=0).

This value is specified as the size of the pick path array in pick path entries. If the actual pick information is longer than the *pickpath* array provided by the application, the overage is truncated.

*status* — **returned by the graPHIGS API, fullword integer**
Status (`1=NONE, 2=OK`).

*depth* — **returned by the graPHIGS API, fullword integer**
Depth of pick path returned in the *pickpath* parameter.

*pickpath* **— returned by the graPHIGS API, array of fullword integers**
Pick path to picked primitive. List of pick path triplets. Each triplet represents the structure identifier, the pick identifier, and the element number of the pick path.

If the actual pick information is longer than the pick path array provided by the application, the overage is truncated.

Pick path data is returned in the *pickpath* parameter in the following order: Entry 1, Entry 2, ...Entry N.

| | | | |
|---|---|---|---|
| Entry 1 | Structure ID 1 | Pick ID 1 | Element #1 |
| Entry 2 | Structure ID 2 | Pick ID 2 | Element #2 |
| . | . | . | . |
| . | . | . | . |
| . | . | . | . |
| Entry N | Structure ID N | Pick ID N | Element #N |

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**505** LENGTH OF RETURN ARRAY < ZERO

**Related Subroutines**

**GPADCN**
Add Class Name to Set

**GPINPK**
Initialize Pick

**GPPKF**
Set Pick Filter

**GPPKID**
Set Pick Identifier

**GPPKMO**
Set Pick Mode

**GPQLI**
Inquire List of Logical Input Devices

**GPRCN**
Remove Class Name from Set

**GPRQXP**
Request Extended Pick

**RCP code**

201335563 (X'0C00230B')

# GPRQSK - Request Stroke

**GPRQSK (*wsid*, *device*, *length*, *status*, *view*, *npoint*, *pointarray*)**

### Purpose

Use **GPRQSK** to have the graPHIGS API execute a request to the specified stroke device.

The graPHIGS API returns a sequence of World Coordinate (WC) points and the view table index whose matrix was used to convert the stroke locations to World Coordinates (WC).

Stroke input is returned from the view active for input with the highest input priority that contains all the points. View 0 is the highest priority view, unless modified by your application.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Stroke device number (>=1).

*length* **— specified by user, fullword integer**
Length of *pointarray* provided by the application for the graPHIGS API to return stroke data (>=0).

This value is specified as the size of the *pointarray* in point entries.

*status* **— returned by the graPHIGS API, fullword integer**
Status (1=NONE, 2=OK).

*view* **— returned by the graPHIGS API, fullword integer**
View table index.

*npoint* **— returned by the graPHIGS API, fullword integer**
Number of points returned in point array (>=0).

*pointarray* **— returned by the graPHIGS API, array of short floating-point numbers (WC)**
Coordinates (3D) of points in stroke returned as a pointlist of width=3.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**505** LENGTH OF RETURN ARRAY < ZERO

### Related Subroutines

**GPINSK**
Initialize Stroke

**GPQLI**
Inquire List of Logical Input Devices

**GPSKMO**
Set Stroke Mode

### RCP code

201335560 (X'0C002308')

# GPRQST - Request String

**GPRQST (*wsid*, *device*, *length*, *status*, *number*, *string*)**

### Purpose

Use **GPRQST** to have the graPHIGS API execute a request to the specified string device.

The graPHIGS API returns a character string from the device.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
String device number (>=1).

*length* **— specified by user, fullword integer**
Length of string array in bytes provided by the application for the graPHIGS API to return the string information (>=1).

*status* **— returned by the graPHIGS API, fullword integer**
Status of string input (`1=NONE, 2=OK`).

*number* **— returned by the graPHIGS API, fullword integer**
Number of bytes returned.

*string* **— returned by the graPHIGS API, variable length character string**
Character string.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**505** LENGTH OF RETURN ARRAY < ZERO

### Related Subroutines

**GPINST**
Initialize String

**GPQLI**
Inquire List of Logical Input Devices

**GPSTMO**
Set String Mode

### RCP code

201335564 (X'0C00230C')

## GPRQVL - Request Valuator

**GPRQVL (*wsid*, *device*, *status*, *value*)**

### Purpose

Use **GPRQVL** to have the graPHIGS API execute a request to the specified valuator device.

The value returned is in the range specified by your application through the Initialize Valuator (**GPINVL**) subroutine.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Valuator device number (>=1).

*status* **— returned by the graPHIGS API, fullword integer**
Status of valuator input (1=NONE, 2=OK).

*value* **— returned by the graPHIGS API, short floating-point number**
Value.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

### Related Subroutines

**GPINVL**
Initialize Valuator

**GPQLI**
Inquire List of Logical Input Devices

**GPVLMO**
Set Valuator Mode

### RCP code

201335561 (X'0C002309')

## GPRQXP - Request Extended Pick

**GPRQXP (*wsid*, *device*, *maxdepth*, *status*, *view*, *point*, *modelling*, *depth*, *pickpath*)**

### Purpose

Use **GPRQXP** to have graPHIGS API execute a request to the specified pick device. This subroutine can be used only for a pick device which provides the extended pick information. For a pick device which does

not return the extended information, the Request Pick (**GPRQPK**) subroutine should be used. The Inquire Pick Measure Type (**GPQPKT**) subroutine can be used to determine if the pick device provides the extended pick information.

The pick path information is returned in the order specified in the Initialize Pick (**GPINPK**) subroutine that is, 1=TOP_FIRST or 2=BOTTOM_FIRST. If **GPINPK** has not been called, the default value is 1=TOP_FIRST.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Pick device number (>=1).

*maxdepth* **— specified by user, fullword integer**
Length of pick path array provided by the application in which the graPHIGS API returns the pick information (>=0).

This value is specified as the size of the pick path array in terms of pick path entries.

*status* **— returned by the graPHIGS API, fullword integer**
Status (1=NONE, 2=OK).

*view* **— returned by the graPHIGS API, fullword integer**
Index of the view which was used to render the instance of the primitive that was picked.

*point* **— returned by the graPHIGS API, 3 short floating-point numbers**
Position of the center of the pick aperture in NPC when the pick was detected. For the typical physical input device that is used to drive the logical pick device (for example, tablet), only the*x* and *y* values are meaningful. The *z* value always contains a constant (0).

*modelling* **— returned by the graPHIGS API, 16 short floating-point numbers**
Composite modelling transformation that was used to transform the instance of the primitive that was picked.

*depth* **— returned by the graPHIGS API, fullword integer**
Depth of the pick path returned in the *pickpath* parameter.

*pickpath* **— returned by the graPHIGS API, array of fullword integers**
Pick path for picked primitive. List of pick path quadruples. Each quadruple represents a structure identifier, pick identifier, label and element number of the picked primitive or an execute structure element of the *pickpath*. Pick path data is returned in the order (1=TOP_FIRST, 2=BOTTOM_FIRST) that was last set through an Initialize Pick (**GPINPK**) subroutine.

If the actual pick information is longer than the *pickpath* array provided by the application, the overage is truncated. Pick path data is returned in the *pickpath* parameter in the following order: Entry 1, Entry 2, ...Entry N.

| | | | | |
|---|---|---|---|---|
| Entry 1 | Structure ID 1 | Pick ID 1 | Label #1 | Element #1 |
| Entry 2 | Structure ID 2 | Pick ID 2 | Label #2 | Element #2 |
| . | . | . | . | . |
| . | . | . | . | . |
| . | . | . | . | . |
| Entry N | Structure ID N | Pick ID N | Label #N | Element #N |

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**141** INPUT DEVICE NOT IN CORRECT MODE

**164** PICK DEVICE DOES NOT PROVIDE EXTENDED INFORMATION

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**505** LENGTH OF RETURN ARRAY < ZERO

### Related Subroutines

**GPADCN**
Add Class Name to Set

**GPINPK**
Initialize Pick

**GPPKF**
Set Pick Filter

**GPPKID**
Set Pick Identifier

**GPPKMO**
Set Pick Mode

**GPQPKT**
Inquire Pick Measure Type

**GPRCN**
Remove Class Name from Set

### RCP code

201335566 (X'0C00230E')

## GPSKMO - Set Stroke Mode

**GPSKMO (*wsid*, *device*, *mode*, *echosw*)**

### Purpose

Use **GPSKMO** to set the operating mode of the specified stroke device.

After the stroke mode is set, its echoing state is set to `1=NOECHO` or `2=ECHO` based on the *echosw* parameter. Depending on the specified operating mode, `1=REQUEST, 2=SAMPLE`, or `3=EVENT`, an interaction with the given device may either begin or end.

> **Note:** The input device is reset with the initialization values when the **GPSKMO** subroutine is called with *mode* parameters set to SAMPLE or EVENT.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Stroke device number (>=1).

*mode* **— specified by user, fullword integer**
Operating mode (`1=REQUEST, 2=SAMPLE, 3=EVENT`).

*echosw* **— specified by user, fullword integer**
Echo switch (1=NOECHO, 2=ECHO).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**326** OPERATING MODE IS INVALID

**327** ECHO SWITCH VALUE IS INVALID

**Related Subroutines**

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201335810 (X'0C002402')

## GPSMCH - Sample Choice

**GPSMCH (*wsid*, *device*, *choice*)**

**Purpose**

Use **GPSMCH** to immediately retrieve the current measure of the specified choice device.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Choice device number (>=1).

*choice* **— returned by the graPHIGS API, fullword integer**
Choice number.

A choice number of zero means no choice.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**Related Subroutines**

**GPCHMO**
Set Choice Mode

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201335556 (X'0C002304')

## GPSMLC - Sample Locator

**GPSMLC (*wsid*, *device*, *view*, *pos*)**

### Purpose

Use **GPSMLC** to immediately retrieve the current measure of the specified locator device.

The measure consists of a locator position in World Coordinates (WC), and the index of the view table entry whose matrix was used to convert the location to World Coordinates.

Locator input is returned from the view active for input with the highest input priority under the cursor. View 0 is the highest priority view, unless modified by your application.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Locator device number (>=1).

*view* **— returned by the graPHIGS API, fullword integer**
View table index.

*pos* **— returned by the graPHIGS API, 3 short floating-point numbers (WC)**
Locator position.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

### Related Subroutines

**GPLCMO**
Set Locator Mode

**GPQLI**
Inquire List of Logical Input Devices

### RCP code

201335553 (X'0C002301')

## GPSMPK - Sample Pick

**GPSMPK (*wsid*, *device*, *length*, *depth*, *pickpath*)**

### Purpose

Use **GPSMPK** to immediately retrieve the current measure of the specified pick device.

The *pickpath* information is returned in the order specified in the Initialize Pick (**GPINPK**) subroutine, that is, `1=TOP_FIRST` or `2=BOTTOM_FIRST`. If **GPINPK** has not been called, the default value is `1=TOP_FIRST`.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*device* — **specified by user, fullword integer**
Pick device number (>=1).

*length* — **specified by user, fullword integer**
Length of pick path array provided by the application in which the graPHIGS API returns the pick information (>=0).

This value is specified as the size of the pick path array in pick path entries. If the actual pick information is longer than the *pickpath* array provided by the application, the overage is truncated.

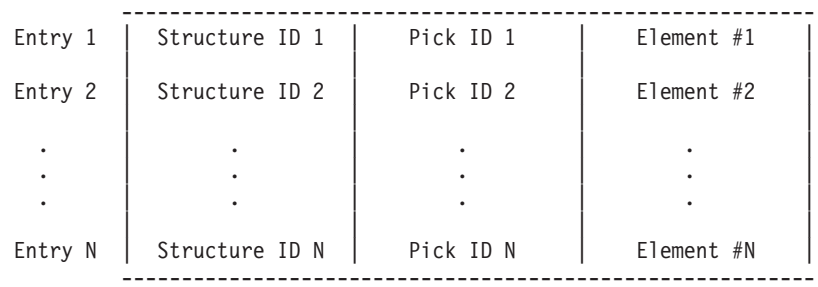
*depth* — **returned by the graPHIGS API, fullword integer**
Depth of the pick path returned in the *pickpath* parameter.

A depth of the actual pick path equal to zero indicates no pick.

*pickpath* — **returned by the graPHIGS API, array of fullword integers**
Pick path to picked primitive. List of pick path triplets where each triplet represents the structure identifier, the pick identifier and the element number of the pick path.

The pick path is returned in the order specified in the initialize pick subroutine, that is, `1=TOP_FIRST` or `2=BOTTOM_FIRST`.

If the actual pick information is larger than the *pickpath* array provided by the application, the overage is truncated.




### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**505** LENGTH OF RETURN ARRAY < ZERO

### Related Subroutines

**GPADCN**
Add Class Name to Set

**GPPKF**
Set Pick Filter

**GPPKID**
Set Pick Identifier

**GPPKMO**
Set Pick Mode

**GPQLI**
Inquire List of Logical Input Devices

**GPRCN**
Remove Class Name from Set

**GPSMXP**
Sample Extended Pick

**RCP code**

201335557 (X'0C002305')

## GPSMSK - Sample Stroke

**GPSMSK (*wsid*, *device*, *length*, *view*, *npoint*, *pointarray*)**

**Purpose**

Use **GPSMSK** to immediately retrieve the current measureof the specified stroke device.

This measure consists of a sequence of stroke positions (not exceeding the current input buffer size) in World Coordinates (WC), and the index of the view table entry whose matrix was used to convert the stroke locations to World Coordinates.

Stroke input is returned from the view active for input with the highest input priority which contains all the points. View 0 is the highest priority view, unless modified by your application.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Stroke device number (>=1).

*length* **— specified by user, fullword integer**
Length of *pointarray* provided by the application for the graPHIGS API to return stroke information (>=0) This value is specified as the size of the *pointarray* in point entries.

*view* **— returned by the graPHIGS API, fullword integer**
View table index.

*npoint* **— returned by the graPHIGS API, fullword integer**
Number of points in the stroke measure.

*pointarray* **— returned by the graPHIGS API, array of short floating-point numbers (WC)**
Coordinates of points in the stroke returned as a pointlist of width=3.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**505** LENGTH OF RETURN ARRAY < ZERO

**Related Subroutines**

**GPQLI**
Inquire List of Logical Input Devices

**GPSKMO**
Set Stroke Mode

**RCP code**

201335554 (X'0C002302')

## GPSMST - Sample String

**GPSMST (*wsid*, *device*, *length*, *number*, *string*)**

**Purpose**

Use **GPSMST** to retrieve the current measure of the specified string device.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
String device number (>=1).

*length* **— specified by user, fullword integer**
Length, in bytes, of string array provided by the application in which the graPHIGS API returns the string data (>=0).

*number* **— returned by the graPHIGS API, fullword integer**
Number of bytes returned.

*string* **— returned by the graPHIGS API, variable length character string**
Character string.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**505** LENGTH OF RETURN ARRAY < ZERO

**Related Subroutines**

**GPQLI**
Inquire List of Logical Input Devices

**GPSTMO**
Set String Mode

**RCP code**

201335558 (X'0C002306')

## GPSMVL - Sample Valuator

**GPSMVL (*wsid*, *device*, *value*)**

**Purpose**

Use **GPSMVL** to retrieve the current measure of the specified valuator device.

The delivered value is in the range specified for this device through the Initialize Valuator (**GPINVL**) subroutine.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Valuator device number (>=1).

*value* **— returned by the graPHIGS API, short floating-point number**
Value.

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

### Related Subroutines

**GPQLI**
Inquire List of Logical Input Devices

**GPVLMO**
Set Valuator Mode

### RCP code

201335555 (X'0C002303')

## GPSMXP - Sample Extended Pick

**GPSMXP (*wsid*, *device*, *maxdepth*, *view*, *point*, *modelling*, *depth*, *pickpath*)**

### Purpose

Use **GPXMSP** to immediately retrieve the current measure of the specified pick device. This subroutine can be used only for a pick device which provides the extended pick information. For a pick device which does not return the extended information, the Sample Pick (**GPSMPK**) subroutine should be used. The Inquire Pick Measure Type (**GPQPKT**) subroutine can be used to determine if the pick device provides the extended pick information.

The pick path information is returned in the order specified in the Initialize Pick (**GPINPK**) subroutine, that is, `1=TOP_FIRST` or `2=BOTTOM_FIRST`. If **GPINPK** has not been called, the default value is `1=TOP_FIRST`.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Pick device number (>=1).

*maxdepth* **— specified by user, fullword integer**
Length of pick path array provided by the application in which the graPHIGS API returns the pick information (>=0).

This value is specified as the size of the pick path array in terms of pick path entries.

*view* **— returned by the graPHIGS API, fullword integer**
Index of the view which was used to render the instance of the primitive that was picked.

*point* **— returned by the graPHIGS API, 3 short floating-point numbers**
Position of the center of the pick aperture in NPC when the pick was detected. For the typical physical input device that is used to drive the logical pick device (for example, tablet), only the*x* and *y* values are meaningful. The *z* value will always contain a constant (0).

*modelling* **— returned by the graPHIGS API, 16 short floating-point numbers**
Composite modelling transformation that was used to transform the instance of the primitive that was picked.

*depth* **— returned by the graPHIGS API, fullword integer**
Depth of the pick path returned in the *pickpath* parameter.

*pickpath* **— returned by the graPHIGS API, array of fullword integers**
Pick path for picked primitive. List of pick path quadruples. Each quadraple represents a structure identifier, pick identifier, label and element number of the picked primitive or an execute type structure element of the *pickpath* Pick path data is returned in the order (`1=TOP_FIRST`, `2=BOTTOM_FIRST`) that was last set through an Initialize Pick (**GPINPK**) subroutine.

If the actual pick information is longer than the pick path array provided by the application, the overage is truncated. Pick path data is returned in the *pickpath* parameter in the following order: Entry 1, Entry 2, ...Entry N.

| | | | | |
|---|---|---|---|---|
| Entry 1 | Structure ID 1 | Pick ID 1 | Label #1 | Element #1 |
| Entry 2 | Structure ID 2 | Pick ID 2 | Label #2 | Element #2 |
| . | . | . | . | . |
| . | . | . | . | . |
| . | . | . | . | . |
| Entry N | Structure ID N | Pick ID N | Label #N | Element #N |

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**164** PICK DEVICE DOES NOT PROVIDE EXTENDED INFORMATION

**505** LENGTH OF RETURN ARRAY < ZERO

**Related Subroutines**

**GPADCN**
Add Class Name to Set

**GPPKF**
Set Pick Filter

**GPPKID**
Set Pick Identifier

**GPPKMO**
Set Pick Mode

**GPQLI**
Inquire List of Logical Input Devices

**GPQPKT**
Inquire Pick Measure Type

**GPRCN**
Remove Class Name from Set

**RCP code**

201335565 (X'0C00230D')

## GPSTMO - Set String Mode

**GPSTMO (*wsid*, *device*, *mode*, *echosw*)**

**Purpose**

Use **GPSTMO** to set the operating mode of the specified string device.

After the string mode is set, its echoing state is set to 1=NOECHO or 2=ECHO based on the *echosw* parameter. Depending on the specified operating mode, 1=REQUEST, 2=SAMPLE or 3=EVENT, an interaction with the given device may either begin or end.

> **Note:** The input device is reset with the initialization values when the **GPSTMO** subroutine is called with *mode* parameters set to SAMPLE or EVENT.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
String device number (>=1).

*mode* **— specified by user, fullword integer**
Operating mode (1=REQUEST, 2=SAMPLE, 3=EVENT).

*echosw* **— specified by user, fullword integer**
Echo switch (1=NOECHO, 2=ECHO).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**326** OPERATING MODE IS INVALID

**327** ECHO SWITCH VALUE IS INVALID

**Related Subroutines**

**GPQLI**
Inquire List of Logical Input Devices

**RCP code**

201335814 (X'0C002406')

# GPVLMO - Set Valuator Mode

**GPVLMO (*wsid*, *device*, *mode*, *echosw*)**

### Purpose

Use **GPVLMO** to set the operating mode of the specified valuator device.

After the valuator mode is set, its echoing state is set to 1=NOECHO or 2=ECHO based on the *echosw* parameter. Depending on the specified operating mode, 1=REQUEST, 2=SAMPLE or 3=EVENT, an interaction within the given device may begin or end.

> **Note:** The input device is reset with the initialization values when the **GPVLMO** subroutine is called with *mode* parameters set to SAMPLE or EVENT.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Valuator device number (>=1).

*mode* **— specified by user, fullword integer**
Operating mode (1=REQUEST, 2=SAMPLE, 3=EVENT).

*echosw* **— specified by user, fullword integer**
Echo switch (1=NOECHO, 2=ECHO).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**168** INPUT DEVICE IS CURRENTLY OWNED BY ANOTHER CONNECTION

**326** OPERATING MODE IS INVALID

**327** ECHO SWITCH VALUE IS INVALID

### Related Subroutines

**GPQLI**
Inquire List of Logical Input Devices

### RCP code

201335811 (X'0C002403')

# Chapter 12. Font Subroutines

Font subroutines let the application program manage a workstation's font resources.

Through the use of Font directories, fonts may be loaded to the directory and shared by workstation resources.

## GPACFO - Activate Font

**GPACFO(*wsid*, *csid*, *font*)**

### Purpose

Use **GPACFO** to activate a geometric text font to the specified workstation. If the specified character set/font pair is already active, this subroutine is ignored.

The character set/font pair is searched for first in the font directory associated with the workstation (if one exists) and then in the nucleus font disk system.

Activation of a character set/font readies it for use by text primitives. Font 1 of the primary character set is always active.

The maximum number of simultaneously active fonts on a workstation is determined by the workstation's font pool size. For more information on available fonts, see *The graPHIGS Programming Interface: Technical Reference*.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*csid* **— specified by user, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

*font* **— specified by user, fullword integer**
Font number (>=1)

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **75** | TEXT FONT VALUE IS INVALID |
| **542** | CHARACTER SET IDENTIFIER IS INVALID |
| **559** | FONT POOL SIZE EXCEEDED ON WORKSTATION |
| **563** | CHARACTER SET/FONT COMBINATION IS NOT AVAILABLE FOR GEOMETRIC TEXT |
| **647** | UNICODE IS NOT SUPPORTED ON THE SPECIFIED WORKSTATION |

### Related Subroutines

| | |
|---|---|
| **GPDAFO** | Deactivate Font |
| **GPLDFO** | Load Font |

| | |
|---|---|
| **GPQFO** | Inquire Active Fonts |
| **GPQFP** | Inquire Font Pool Size |

### RCP code

201337346 (X'0C002A02')

## GPAFDW - Associate Font Directory with Workstation

**GPAFDW(*wsid*, *fdid*)**

### Purpose

Use **GPAFDW** to associate a font directory with a workstation. If there is a font directory associated with the workstation, it is replaced by the new one. Only one font directory can be associated to a workstation at any one time.

Associating a font directory to a workstation allows any character set/font loaded in the directory to be activated to the workstation.

Fonts that reside in the directory take precedence over fonts that exist on the nucleus disk system when they are activated to the workstation.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*fdid* — **specified by user, fullword integer**
Font directory identifier.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **215** | SPECIFIED RESOURCES DO NOT EXIST ON THE SAME NUCLEUS |
| **242** | SPECIFIED FONT DIRECTORY DOES NOT EXIST |

### Related Subroutines

| | |
|---|---|
| **GPACFO** | Activate Font |
| **GPCRFD** | Create Font Directory |
| **GPDAFO** | Deactivate Font |
| **GPLDFO** | Load Font |

### RCP code

201337351(X'0C002A07')

## GPDAFO - Deactivate Font

**GPDAFO (*wsid*, *csid*, *font*)**

### Purpose

Use **GPDAFO** to deactivate a geometric text font from the specified workstation. If the specified character set/font pair is not active, this subroutine has no effect.

The character set and font can still be active on another workstation.

Deactivating a font from a workstation has no effect on its existence in a font directory (i.e., if a font currently exists in a font directory it will remain in the font directory even though it has been deactivated from the workstation).

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*csid* **— specified by user, fullword integer**
Character set identifier.

See Appendix A, Character Set and Font Identifiers for more information.

*font* **— specified by user, fullword integer**
Font number (>=1)

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **75** | TEXT FONT VALUE IS INVALID |
| **542** | CHARACTER SET IDENTIFIER IS INVALID |
| **553** | PRIMARY CHARACTER SET FONT ONE CANNOT BE DEACTIVATED |

### Related Subroutines

| | |
|---|---|
| **GPACFO** | Activate Font |
| **GPQFO** | Inquire Active Fonts |

### RCP code

201337347 (X'0C002A03')

## GPDLFO - Delete Font

**GPDLFO(*fdid*,*csid*,*font*)**

### Purpose

Use **GPDLFO** to delete a font definition from the specified font directory. If the specified font definition does not exist in the font directory, this subroutine has no effect.

If the specified character set/font is currently active to any workstation, it remains active to the workstation until the application deactivates it. However, it is still deleted from the font directory.

### Parameters

*fdid* — **specified by user, fullword integer**
Font directory identifier.

*csid* — **specified by user, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

*font* — **specified by user, fullword integer**
Font number (>=1).

### Error Codes

| | |
|---|---|
| **75** | TEXT FONT VALUE IS INVALID |
| **242** | SPECIFIED FONT DIRECTORY DOES NOT EXIST |
| **542** | CHARACTER SET IDENTIFIER IS INVALID |

### Related Subroutines

| | |
|---|---|
| **GPCRFD** | Create Font Directory |
| **GPLDFO** | Load Font |

### RCP code

201345282 (X'0C004902')

## GPLDFO - Load Font

**GPLDFO (*fdid*, *csid*, *font*, *option*)**

### Purpose

Use **GPLDFO** to load a font definition of a geometric text font to the specified font directory. The specified character set/font pair is taken from a disk accessible by the graPHIGS API shell and sent to the specified font directory in the nucleus. If the specified font definition already exists in the font directory, it is replaced by the new one.

Once a font definition of a geometric text font is loaded in the font directory, and the font directory has been associated to a workstation, the character set/font pair may be activated to a workstation. Font definitions that reside in a font directory take precedence over font definitions that are accessible by the nucleus when they are activated to the workstation. Font directories are primarily intended for networked environments.

### Parameters

*fdid* — **specified by user, fullword integer**
Font directory identifier.

*csid* — **specified by user, fullword integer**
Character set identifier.

See Appendix A, "Character Set and Font Identifiers," for more information.

*font* — **specified by user, fullword integer**
Font number (>=1).

*option* **— specified by user, fullword integer**
Font option (2=IBM_DEFINED_RANGE, 3=USER_DEFINED_RANGE). This parameter specifies what portion of the font definition should be loaded and has its meaning only for the IBM defined double byte character sets. For other character sets, this parameter is ignored.

**Error Codes**

**75** TEXT FONT VALUE IS INVALID

**242** SPECIFIED FONT DIRECTORY DOES NOT EXIST

**245** FONT OPTION IS INVALID

**542** CHARACTER SET IDENTIFIER IS INVALID

**Related Subroutines**

**GPACFO**
Activate Font

**GPAFDW**
Associate Font Directory with Workstation

**GPCRFD**
Create Font Directory

**GPDLFO**
Delete Font

**RCP code**

201345281 (X'0C004901')

# Chapter 13. Image Subroutines

This section describes the image subroutines that can be used to manipulate image data. The subroutines discussed in this section allow the following operations:

- Filling a position of an image board by a constant value.
- Moving data between an application's image storage to an image board.
- Defining an image on a workstation for subsequent display.

For each subroutine in this section, each target rectangle must have the same horizontal and vertical size as the source rectangle, and the source rectangle must be entirely within the source image data. See *The graPHIGS Programming Interface: Understanding Concepts* for more information about source and target rectangles.

## GPCAI - Cancel Image

**GPCAI (*wsid*, *index*)**

### Purpose

Use **GPCAI** to cancel an image definition on the specified workstation. If the specified image is used for any image mapping, the image mapping is removed from the view.

If the specified image is not defined, no action is performed.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*index* — **specified by user, fullword integer**
Defined image index (>= 1).

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **288** | IMAGE INDEX NOT WITHIN WORKSTATION TABLE RANGE |

### Related Subroutines

| | |
|---|---|
| **GPCIM2** | Create Image Mapping 2 |
| **GPCIM3** | Create Image Mapping 3 |
| **GPDFI** | Define Image |
| **GPQIW** | Inquire List of Images on the Workstation |

### RCP code

201346306 (X'0C004D02')

# GPDFI - Define Image

**GPDFI (*wsid*, *index*, *conn*, *ctid*, *nibid*, *libid*)**

### Purpose

Use **GPDFI** to define an image on a workstation. Image boards included in the image definition become ready to be displayed on the specified workstation. The specified image boards for the image definition must have been created with the same horizontal and vertical sizes or an error is generated and the image is not defined. If the specified image is already defined, it is canceled and redefined.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Defined image index (>=1).

*conn* **— specified by user, fullword integer**
Connection type (-1=FRAME_BUFFER_COMPATIBLE, 1=COMPONENT, 2=INDEXED).

*ctid* **— specified by user, fullword integer**
Color table identifier. For connection type (*conn*) -1,this parameter is ignored.

*nibid* **— specified by user, fullword integer**
Number of image boards.

*libid* **— specified by user, array of fullword integers**
List of image board identifiers. For each of the image connection types, the following number of image boards must be specified:

**If*conn*=- 1 (FRAME_BUFFER_COMPATIBLE)**
the number of image board identifiers specified in the list equals the number of frame buffer components for the specified workstation.

**If*conn*=1 (COMPONENT)**
the image board identifiers must be specified in the list.

**If*conn*=2 (INDEXED)**
1 image board identifier must be specified in the list.

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **215** | SPECIFIED RESOURCES DO NOT EXIST ON THE SAME NUCLEUS |
| **232** | SPECIFIED IMAGE BOARD DOES NOT EXIST |
| **284** | COLOR TABLE IDENTIFIER DOES NOT EXIST |
| **288** | IMAGE INDEX NOT WITHIN WORKSTATION TABLE RANGE |
| **291** | SPECIFIED IMAGE CONNECTION TYPE IS NOT SUPPORTED |
| **292** | NUMBER OF IMAGE BOARDS DOES NOT MATCH THE CONNECTION TYPE |
| **293** | CHARACTERISTICS OF THE SPECIFIED IMAGE BOARDS DO NOT MATCH |

### Related Subroutines

| | |
|---|---|
| **GPCAI** | Cancel Image |
| **GPCIM2** | Create Image Mapping 2 |
| **GPCIM3** | Create Image Mapping 3 |
| **GPCRIB** | Create Image Board |
| **GPQCID** | Inquire List of Color Table Identifiers |
| **GPQICH** | Inquire Image Characteristics |
| **GPQIDF** | Inquire Image Definition Facilities |
| **GPQIW** | Inquire List of Images on the Workstation |
| **GPQXCF** | Inquire Extended Color Facilities |

### RCP code

201346305 (X'0C004D01')

## GPFRCT - Fill Rectangle

**GPFRCT (*ibid*, *origin*, *size*, *value*)**

### Purpose

Use **GPFRCT** to fill a rectangular area in the specified image board with a specific value. The specified fill value's least significant part, corresponding to the image board's bit depth, is used to fill the specified rectangle.

See *The graPHIGS Programming Interface: Understanding Concepts* for a discussion on image processing.

### Parameters

*ibid* **— specified by user, fullword integer**
Image board identifier.

*origin* **— specified by user, two fullword integers**
Rectangle origin (*x*, *y*) (>=0).

*size* **— specified by user, two fullword integers**
Rectangle size (*SX*, *SY*) (>=1).

*value* **— specified by user, fullword integer**
Fill value.

### Error Codes

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

**236** RECTANGLE DEFINITION IS INVALID

### Related Subroutines

**GPCRIB**
Create Image Board

**GPQIBC**
Inquire Image Board Characteristics

**GPQIBF**
Inquire Image Board Facilities

**GPQIMC**
Inquire Image Mapping Characteristics

**GPQIMF**
Inquire Image Mapping Facilities

**RCP code**

201346310 (X'0C004D06')

## GPRRCT - Read Rectangle

**GPRRCT (*sibid*, *sorigin*, *size*, *format*, *parm*, *torigin*, *data*)**

### Purpose

Use **GPRRCT** to read pixel data from an image board.

The specified target rectangle need not be inside of the target application image; however, the specified source rectangle must be inside of the source image board. When the target rectangle exceeds the target application image boundaries, pixel data outside the boundaries is discarded. If source and target storage have different bit depth, then source pixels are adjusted to the target bit length by removing most significant bits or adding 0-bit to the most significant part.

See *The graPHIGS Programming Interface: Understanding Concepts* for a discussion on image processing.

### Parameters

*sibid* **— specified by user, fullword integer**
Source image board identifier.

*sorigin* **— specified by user, two fullword integers**
Source rectangle origin (*x*, *y*) (>=0).

*size* **— specified by user, two fullword integers**
Rectangle size (*SX*, *SY*) (>=1).

*format* **— specified by user, fullword integer**
Application image format `1=PIXEL_ARRAY`).

*parm* **— specified by user, array of fullword quantities**
Format dependent parameters. The image format 1 requires the following parameters:

- bit depth — fullword integer 1, 2, 4, 8, or 16
- *x* size — fullword integer (>=1)
- *y* size — fullword integer (>=1)
- pixel order — fullword integer (`1=LEFT_RIGHT_BOTTOM_TOP`, `2=LEFT_RIGHT_TOP_BOTTOM`)

The product of the bit depth and *x* size must be a multiple of 8. This is to ensure that each row of the application's image data starts on a byte boundary.

*torigin* **— specified by user, two fullword integers**
Target rectangle origin (*x*, *y*) (>=0).

*data* **— returned by the graPHIGS API, array of pixels**
Target application image data. For the purpose of byte swapping between different shell/nucleus

environments, the source application image data with bit depth of 16 will be handled as 16-bit halfwords. For all other bit depths, the application image data will be treated as 8-bit unsigned characters.

**Error Codes**

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

**236** RECTANGLE DEFINITION IS INVALID

**237** SPECIFIED APPLICATION IMAGE FORMAT IS NOT SUPPORTED

**240** APPLICATION IMAGE DESCRIPTION IS INVALID

**Related Subroutines**

None

**RCP code**

201346316 (X'0C004D0C')

## GPTRCT - Transfer Rectangle

**GPTRCT (*tibid*, *torigin*, *size*, *sibid*, *sorigin*)**

**Purpose**

Use **GPTRCT** to transfer pixel data from an image board to another image board. Both source and target image boards must reside on the same nucleus.

The specified target rectangle need not be inside of the target image board; however, the specified source rectangle must be inside of the source image board. When the target rectangle exceeds the target image board boundaries, pixel data outside the boundaries is discarded. If source and target storage have different bit depth, source pixels are adjusted to the target bit length by removing most significant bits or adding 0-bit to the most significant part.

**Parameters**

*tibid* **— specified by user, fullword integer**
Target image board identifier.

*torigin* **— specified by user, two fullword integers**
Target rectangle origin (*x*, *y*) (>=0).

*size* **— specified by user, two fullword integers**
Rectangle size (*SX*, *SY*) (>=1).

*sibid* **— specified by user, fullword integer**
Source image board identifier.

*sorigin* **— specified by user, two fullword integers**
Source rectangle origin (*x*, *y*) (>=0).

**Error Codes**

**215** SPECIFIED RESOURCES DO NOT EXIST ON THE SAME NUCLEUS

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

**236** RECTANGLE DEFINITION IS INVALID

**Related Subroutines**

None

**RCP code**

20134613 (X'0C004D09')

# GPTHPO - Three Operand Pixel Operation

**GPTHPO (*tibid*, *torigin*, *size*, *sibid1*, *sorigin1*, *sibid2*, *sorigin2*, *op*, *opparm*)**

**Purpose**

Use **GPTHPO** to produce pixel data by combining two pixels on two image boards. The operation performed to the pixel data is specified by an operation type and accompanied operation dependent parameters.

All specified image boards must reside on the same nucleus. The specified target rectangle need not be inside of the target image board; however, two specified source rectangles must be inside of each image board. When the target rectangle exceeds the target image board boundaries, pixel data outside the boundaries is discarded. If source and target storage have different bit depth, source pixels are adjusted to the target bit length by removing most significant bits or adding 0-bit to the most significant part.

**Parameters**

*tibid* **— specified by user, fullword integer**
Target image board identifier.

*torigin* **— specified by user, two fullword integers**
Target rectangle origin (*x*, *y*) (>=0).

*size* **— specified by user, two fullword integers**
Rectangle size (*SX*, *SY*) (>=1).

*sibid1* **— specified by user, fullword integer**
First source image board identifier.

*sorigin1* **— specified by user, two fullword integers**
First source rectangle origin (*x*, *y*) (>=0).

*sibid2* **— specified by user, fullword integer**
Second source image board identifier.

*sorigin2* **— specified by user, two fullword integers**
Second source rectangle origin (*x*, *y*) (>=0).

*op* **— specified by user, fullword integer**
Operation type (`1=LOGICAL, 2=ARITHMETIC`).

*opparm* **— specified by user, array of fullword quantities**
Operation-dependent parameters.

The following operation types are defined and each of them requires specified operation dependent parameters.

**1 (`LOGICAL_OPERATION`)**
Perform a binary logical operation. This operation requires a fullword integer specifying one of the following logical operations:

- 1: 0

- 2: s1 AND s2
- 3: s1 AND (NOT s2)
- 4: s1
- 5: (NOT s1) AND s2
- 6: s2
- 7: s1 XOR s2
- 8: s1 OR s2
- 9: NOT (s1 OR s2)
- 10: NOT (s1 XOR s2)
- 11: NOT s2
- 12: s1 OR (NOT s2)
- 13: NOT s1
- 14: (NOT s1) OR s2
- 15: NOT (s1 AND s2)
- 16: 1

**2 (ARITHMETIC_OPERATION)**
Performs a binary arithmetic operation. This operation requires a fullword integer specifying one of the following arithmetic operations:

- 1: Add with saturation to 2** *n*, where *n* is a bit depth of the target image board.
- 2: Subtract with saturation to 0
- 3: Minimum of two pixels
- 4: Maximum of two pixels

### Error Codes

**215** SPECIFIED RESOURCES DO NOT EXIST ON THE SAME NUCLEUS

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

**236** RECTANGLE DEFINITION IS INVALID

**239** SPECIFIED THREE OPERAND OPERATION IS NOT SUPPORTED

### Related Subroutines

None

### RCP code

201346315 (X'0C004D0B')

## GPTWPO - Two Operand Pixel Operation

**GPTWPO (*tibid*, *torigin*, *size*, *sibid*, *sorigin*, *op*, *opparm*)**

### Purpose

Use **GPTWPO** to transfer pixel data from an image board to another image board through a pixel by pixel operation. The operation performed to the pixel data is specified by an operation type and accompanied operation dependent parameters.

Both source and target image boards must reside on the same nucleus. The specified target rectangle need not be inside of the target image board; however, the specified source rectangle must be inside of the source image board. When the target rectangle exceeds the target image board boundaries, pixel data outside the boundaries is discarded. If source and target storage have different bit depth, source pixels are adjusted to the target bit length by removing most significant bits or adding 0-bit to the most significant part.

**Parameters**

*tibid* **— specified by user, fullword integer**
Target image board identifier.

*torigin* **— specified by user, two fullword integers**
Target rectangle origin (*x*, *y*) (>=0).

*size* **— specified by user, two fullword integers**
Rectangle size (*SX*, *SY*) (>=1).

*sibid* **— specified by user, fullword integer**
Source image board identifier.

*sorigin* **— specified by user, two fullword integers**
Source rectangle origin (*x*, *y*) (>=0).

*op* **— specified by user, fullword integer**
Operation type (`1=REFLECTION`).

*opparm* **— specified by user, array of fullword quantities**
Operation dependent parameters

The following operation is defined and requires operation dependent parameters.

**1 (`REFLECTION`)**
Pixels in the source rectangle are placed into the target rectangle in the reverse order. This operation requires a fullword integer specifying an axis about which the pixels are reflected as follows:

- 1: reflect about x-axis (reverse y-direction)
- 2: reflect about y-axis (reverse x-direction)

**Error Codes**

**215** SPECIFIED RESOURCES DO NOT EXIST ON THE SAME NUCLEUS

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

**236** RECTANGLE DEFINITION IS INVALID

**238** SPECIFIED TWO OPERAND OPERATION IS NOT SUPPORTED

**Related Subroutines**

None

**RCP code**

201346314 (X'0C004D0A')

## GPWRCT - Write Rectangle

**GPWRCT (*tibid*, *torigin*, *size*, *format*, *parm*, *sorigin*, *data*)**

### Purpose

Use **GPWRCT** to write pixel data into an image board. Pixel data for the specified source rectangle will be extracted from the source application image data and written to the target rectangle within the specified image board.

The specified target rectangle need not be inside of the target image board but the specified source rectangle must be inside of the source application image. When the target rectangle exceeds the target image board boundaries, pixel data outside the boundaries is discarded. If source and target data have different bit depth, source pixels are adjusted to the target bit length by removing most significant bits or adding 0-bit to the most significant part.

See *The graPHIGS Programming Interface: Understanding Concepts* for a discussion on image processing.

### Parameters

*tibid* **— specified by user, fullword integer**
Target image board identifier.

*torigin* **— specified by user, two fullword integers**
Target rectangle origin (*x*, *y*) (>=0).

*size* **— specified by user, two fullword integers**
Rectangle size (*SX*, *SY*) (>=1).

*format* **— specified by user, fullword integer**
Source application image format (`1=PIXEL_ARRAY`).

*parm* **— specified by user, variable data**
Format dependent parameters.

The image format 1 requires the following parameters:

- bit depth — fullword integer (1, 2, 4, 8, 16)
- *x* size — fullword integer (>=1)
- *y* size — fullword integer (>=1)
- pixel order — fullword integer (`1=LEFT_RIGHT_BOTTOM_TOP`, `2=LEFT_RIGHT_TOP_BOTTOM`).

  The product of *x* size and bit depth must be a multiple of 8. This is to ensure that each row of the application's image data starts on a byte boundary.

*sorigin* **— specified by user, two fullword integers**
Source rectangle origin (*x*, *y*) (>=0).

*data* **— specified by user, array of pixels**
Source application image data. For the purpose of byte swapping between different shell/nucleus environments, the source application image data with bit depth of 16 will be handled as 16-bit halfwords. Bit depths 1, 2, 4, 8 should be packed respectively as 8, 4, 2 and 1 pixels per byte. For all other bit depths, the application image data will be treated as 8-bit unsigned characters.

### Error Codes

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

**236** RECTANGLE DEFINITION IS INVALID

**237** SPECIFIED APPLICATION IMAGE FORMAT IS NOT SUPPORTED

**240** APPLICATION IMAGE DESCRIPTION IS INVALID

### Related Subroutines

**GPCRIB**
Create Image Board

**GPQAI**
Inquire List of Available Application Image Formats

**GPQIBC**
Inquire Image Board Characteristics

**GPQIBF**
Inquire Image Board Facilities

**GPQIMC**
Inquire Image Mapping Characteristics

**GPQIMF**
Inquire Image Mapping Facilities

### RCP code

201346311 (X'0C004D07')

# Chapter 14. Utility Subroutines

The subroutines in this category provide convenient mechanisms for modifying data or performing calculations.

Some subroutines perform transformations on matrixes; others let your application alter the viewing characteristics in the Viewing Coordinate (VC) system. In addition, the Pack Data Record utility provides a convenient mechanism for the construction of data records used by input device initialization subroutines.

The Convert Data utility allows you to convert data to a form known by a target application when two or more application processes are communicating.

## GPCCV - Convert Coordinate Values

**GPCCV (*wsid*, *ctype*, *ptype*, *number*, *ilist*, *errind*, *olist*)**

### Purpose

Use **GPCCV** to convert coordinate values in a window system from one coordinate system to another. Only workstations which use the facilities of a window system (e.g., X-Windows) support this procedure.

This subroutine converts Normalized Projection Coordinates (NPC), Device Coordinates (DC) or Window Units (WU).

The conversion type (*ctype*) parameter specifies the type of conversion to be performed. The point type (*ptype*) parameter specifies the organization of the input values, either as an array of two-dimensional points (`1=POINT_2D`) or as an array of three-dimensional points (`2=POINT_3D`). The *number* parameter specifies the number of two-dimensional or three-dimensional points in the input list (*ilist*) parameter to convert. The array of the output list (*olist*) is the same size as the *ilist*.

This subroutine is assigned escape identifier 1015.

> **Note:** This subroutine is an escape subroutine, and therefore, may not be available on all workstations. Use the Inquire List of Available Escape Subroutines (**GPQES**) to determine if this subroutine is supported by a specified workstation.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ctype* **— specified by user, fullword integer**
Conversion type (`1=NPC_TO_WU, 2=WU_TO_NPC, 3=DC_TO_WU, 4=WU_TO_DC`).

*ptype* **— specified by user, fullword integer**
Point type (`1=POINT_2D, 2=POINT_3D`)

*number* **— specified by user, fullword integer**
Number of two-dimensional or three-dimensional points to be converted (>=0).

*ilist* **— specified by user, array of short floating-point numbers**
Input list of coordinate values. Depending on the value of the *ptype* parameter, this parameter is treated as an array of two-dimensional or three-dimensional points.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **527** | ESCAPE FUNCTION NOT AVAILABLE |
| **539** | REQUESTED NUMBER < ZERO |
| **624** | CONVERSION TYPE IS INVALID |
| **625** | POINT TYPE IS INVALID |

*olist* **— returned by the graPHIGS API, array of short floating-point numbers**
Output list of converted coordinate values. This list is the same size as *ilist*.

### Error Codes

None

### Related Subroutines

| | |
|---|---|
| **GPDCMM** | Set Device Coordinate Mapping Method |
| **GPGWIN** | Get Window |

### RCP code

201336855 (X'0C002817')

## GPCMT2 - Compose Matrix 2

**GPCMT2 (*matra*, *matrb*, *matrix*)**

### Purpose

Use **GPCMT2** to perform a 3x3 matrix multiplication and return the result.

Transformation Matrix A (*matra*) x Transformation Matrix B (*matrb*) is computed and returned in *matrix*.

For this task, the elements of the transformation matrixes are passed in the following order:

```
|m11 m12 m13|
|m21 m22 m23|---> (m11,m12,m13,m21....m33)
|m31 m32 m33|
```

### Parameters

*matra* **— specified by user, 9 short floating-point numbers**
Transformation matrix A.

*matrb* **— specified by user, 9 short floating-point numbers**
Transformation matrix B.

*matrix* **— returned by the graPHIGS API, 9 short floating-point numbers**
Composed transformation matrix.

### Error Codes

None

**Related Subroutines**

None

**RCP code**

201330955 (X'0C00110B')

## GPCMT3 - Compose Matrix 3

**GPCMT3 (*matra*, *matrb*, *matrix*)**

**Purpose**

Use **GPCMT3** to perform a 4x4 matrix multiplication and return the results.

Transformation Matrix A (*matra*) x Transformation Matrix B (*matrb*) is computed and returned in *matrix*.

For this task, the elements of the transformation matrixes are passed in the following order:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24|---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

**Parameters**

*matra* **— specified by user, 16 short floating-point numbers**
Transformation matrix A.

*matrb* **— specified by user, 16 short floating-point numbers**
Transformation matrix B.

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Composed transformation matrix.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201330954 (X'0C00110A')

## GPCVD - Convert Data

**GPCVD (*datatype*, *env*, *origin*, *datalen*, *idata*, *odata*)**

**Purpose**

Use **GPCVD** to convert data that is valid in one environment to data that is valid in another environment.

The differences in data in different environments could be:

- character encoding can be EBCDIC or ASCII
- floating-point format can be IBM single precision or IEEE single precision
- the byte order of data can be swapped

You need to convert data when:

- An application process is communicating with another application process and the data the one application process receives from the other application process is not in a form that it recognizes. Likewise, you need to convert data when an application process wants to send data to another application process in a form that the receiving application process recognizes.
- An application is directing workstation-dependent output (WDO) to a workstation existing in a different environment.

When using **GPCVD** for converting communication data between application processes:

- The environment descriptor is the descriptor of the environment of the application process communicating with your application process. Your application process has either received data from or is sending data to that application process. An application process can inquire its environment descriptor by issuing the Inquire Shell Identifier (**GPQSH**) subroutine. An application process can pass its environment descriptor to another application process by issuing the Send Broadcast Message (**GPSBMS**) subroutine or the Send Private Message (**GPSPMS**) subroutine.
- The origin parameter identifies whether the data to be converted originated in your application process doing the conversion (`1=LOCAL_DATA`) or in the application process you are communicating with (`2=EXTERNAL_DATA`).

When using **GPCVD** for converting an application's workstation-dependent output (WDO) data to a form recognized by the environment of the workstation to which it is directed:

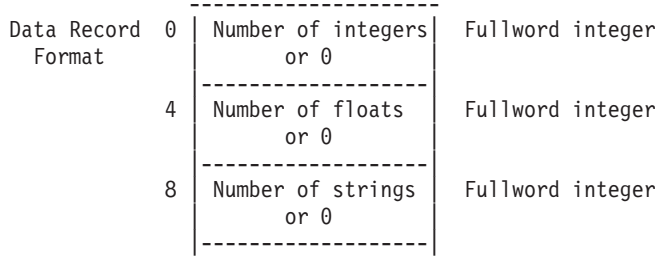
- The environment descriptor is the descriptor of the environment of the workstation to which the WDO is directed. You can inquire the environment descriptor of the workstation by issuing the Inquire Nucleus Environment (**GPQNCE**) subroutine. Specify a *type* parameter of `2=ENVIRONMENT_DESCRIPTOR` and a nucleus identifier (*ncid*) of the nucleus where the workstation was created. If the workstation was opened using the Open Workstation (**GPOPWS**) subroutine, then use a nucleus identifier of one.
- You should set the *origin* parameter to `1=LOCAL_DATA`.

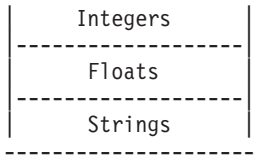
The conversion of string data is done using the current text character set defined in the graPHIGS API state list (see the Set Text Character Set [**GPTXCS**] subroutine [page GPTXCS - Set Text Character Set] ). The one exception to this is if your application is running locally within the 6090, the graPHIGS API uses the IBM defined character set 1 for the conversion.

### Parameters

*datatype* **— specified by user, fullword integer**

Type of data that is being converted (`1=CHARACTER_STRING`, `2=FLOATS`, `3=INTEGERS`, `4=DATA_RECORD`).






**Note:** The first byte of a data record string is the length of the string.

*env* **— specified by user, 4-byte character string**
Environment descriptor.

*origin* **— specified by user, fullword integer**
Origin of the data to be converted (1=LOCAL_DATA, 2=EXTERNAL_DATA).

*datalen* **— specified by user, fullword integer (>0)**
Length in bytes of the data being converted. This also equals the length of the area to return the converted data.

*idata* **— specified by user, variable data**
Data to be converted.

*odata* **— returned by the graPHIGS API, variable data**
The converted data.

**Note:** The input area (*idata*) and the output area (*odata*) may be the same area.

### Error Codes

| | |
|---|---|
| **146** | FIELD IN INPUT DEVICE DATA RECORD IN ERROR |
| **177** | ORIGIN PARAMETER IS INVALID |
| **178** | DATATYPE PARAMETER IS INVALID |
| **179** | ENVIRONMENT DESCRIPTOR IS INVALID |
| **509** | DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH |

### Related Subroutines

| | |
|---|---|
| **GPES** | Escape |
| **GPQNCE** | Inquire Nucleus Environment |
| **GPQSH** | Inquire Shell Identifier |
| **GPSBMS** | Send Broadcast Message |
| **GPSPMS** | Send Private Message |
| **GPWDO** | Workstation-Dependent Output |

### RCP code

201330959 (X'0C00110F')

## GPCVMT - Compute View Matrix

**GPCVMT (*matrix*)**

### Purpose

Use **GPCVMT** to calculate a view matrix based upon the values present in the Utility State List (USL). This view matrix is used as the view matrix parameter on the Set Extended View Representation (**GPXVR**) subroutine.

The matrix returned will perform a change of coordinate system from the World Coordinate (WC) system to a Viewing Coordinate (VC) system in which the origin is the view reference point, the n-axis is the view plane normal, and the v-axis lies in the half plane designated by the View Up vector. This matrix represents the change of coordinate systems by a translation operation followed by a rotation operation.

If the matrix cannot be calculated, the graPHIGS API issues an error.

### Parameters

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

### Error Codes

| | |
|---|---|
| **58** | UP AND PLANE NORMAL VECTORS ARE PARALLEL |

### Related Subroutines

None

### RCP code

201331204 (X'0C001204')

## GPDFCO - Define Coordinate System

**GPDFCO (*origin*, *zplane*, *up*, *matrix*)**

### Purpose

Use **GPDFCO** to compute a three-dimensional, homogeneous transformation matrix.

This matrix mathematically represents a change of coordinates from the default World Coordinate (WC) system to the coordinate system described by the specified input parameters (*origin*, *zplane*, *up*). This matrix represents the change of coordinate systems by a rotation operation followed by a translation operation.

### Parameters

*origin* **— specified by user, 3 short floating-point numbers (WC)**
Point of origin of new coordinate system.

*z* **plane — specified by user, 3 short floating-point numbers (WC)**
*z* plane normal vector of new coordinate system.

*up* **— specified by user, 3 short floating-point numbers (WC)**
Up vector of new coordinate system.

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Transformation matrix. The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

### Error Codes

**58** UP AND PLANE NORMAL VECTORS ARE PARALLEL

### Related Subroutines

None

### RCP code

201330953 (X'0C001109')

## GPEVM2 - Evaluate View Mapping Matrix 2

**GPEVM2 (*window*, *viewpt*, *errind*, *matrix*)**

### Purpose

Use **GPEVM2** to create a view mapping matrix. The matrix can be used as input to the Set Extended View Representation (**GPXVR**) subroutine. This subroutine returns a two-dimensional matrix.

The calculation of the view mapping matrix is as follows:

- The *z* extents for the viewport are set to the *z* extents of the NPC range.
- The projection type is set to `1=PARALLEL`.
- The projection reference point is placed on a line perpendicular to the center of the specified window. The *z* value of the projection reference point is set to one-half of the maximum of the Umax-Umin and Vmax-Vmin.
- The view plane distance is set to zero.
- The far clipping plane is set to the negative of one-half of the maximum of the Umax-Umin and Vmax-Vmin.
- The near clipping plane is set to one-half of the maximum of the Umax-Umin and Vmax-Vmin.

If the view mapping matrix can be computed, the error indicator is set to zero, and the matrix is returned. If the matrix cannot be computed, the error indicator contains an error number indicating the reason. In this case, the value of the output parameter is unpredictable.

### Parameters

*window* **— specified by user, 4 short floating-point numbers (VC)**
Window limits (Umin, Umax, Vmin, Vmax)

*viewpt* **— specified by user, 4 short floating-point numbers (NPC)**
Viewport limits (Xmin, Xmax, Ymin, Ymax)

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**44** INVALID WINDOW DEFINITION

*matrix* **— returned by the graPHIGS API, 9 short floating-point numbers**
View mapping matrix. For the output view matrix, the elements are in the following order:

```
|m11  m12  m13|
|m21  m22  m23| --->(m11,m12,m13,m21.....m43)
|m41  m42  m43|
```

**Error Codes**

None

**Related Subroutines**

**GPXVR**
Set Extended View Representation

**RCP code**

201331208 (X'0C001208')

## GPEVM3 - Evaluate View Mapping Matrix 3

**GPEVM3 (*window*, *viewpt*, *type*, *point*, *dist*, *near*, *far*, *errind*, *matrix*)**

**Purpose**

Use **GPEVM3** to create a view mapping matrix. The matrix can be used as input to the Set Extended View Representation (**GPXVR**) subroutine.

If the view mapping matrix can be computed, the error indicator is set to zero, and the matrix is returned. If the matrix cannot be computed, the error indicator contains an error number indicating the reason. In this case, the value of the output parameter is unpredictable.

**Parameters**

*window* **— specified by user, 4 short floating-point numbers (VC)**
Window limits (Umin, Umax, Vmin, Vmax).

*viewpt* **— specified by user, 6 short floating-point numbers (NPC)**
Viewport limits (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*type* **— specified by user, fullword integer**
Projection type (`1=PARALLEL, 2=PERSPECTIVE`).

*point* **— specified by user, 3 short floating point numbers (VC)**
Projection reference point (*U*, *V*, *N*).

*dist* **— specified by user, short floating-point number (VC)**
Distance of view plane from view reference point along n-axis.

*near* **— specified by user, short floating-point number (VC)**
Distance of near plane from view reference point along n-axis.

*far* **— specified by user, short floating-point number (VC)**
Distance of far plane from view reference point along n-axis.

*errind* **— returned by the graPHIGS API, fullword integer**
If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**44** INVALID WINDOW DEFINITION

**55** PRP IS POSITIONED ON THE VIEW PLANE

**330** INVALID VIEWPORT

**331** PROJECTION TYPE IS INVALID

**336** FAR CLIPPING PLANE IN FRONT OF NEAR CLIPPING PLANE

**608** FRONT PLANE DISTANCE = BACK PLANE DISTANCE WHEN Z-EXTENT NON-ZERO

**610** PROJECTION REFERENCE POINT BETWEEN NEAR AND FAR PLANES

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
View mapping matrix. The elements are returned in the following order for the view mapping matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

**Error Codes**

None

**Related Subroutines**

**GPXVR**
Set Extended View Representation

**RCP code**

201331207 (X'0C001207')

# GPPREC - Pack Data Record

**GPPREC (*numi*, *iary*, *numr*, *rary*, *nums*, *swidth*, *lens*, *sary*, *mlodr*, *errind*, *lodr*, *datarec*)**

**Purpose**

Use **GPPREC** to construct a data record for passing to input device initialization routines. The data record constructed by **GPPREC** consists of a header identifying the number of integers, reals, and character strings in the data record, followed by the actual data. The *lodr* and *datarec* output parameters can be passed to the desired input device initialization subroutine.

**GPPREC** accepts as input a list of integers, a list of reals, and a list of character strings. The list of character strings is passed in a format similar to a pointlist. That is, a width parameter is specified (*swidth*) which tells the graPHIGS API how many bytes are placed between the starting location of subsequent character strings. The length of each character string is specified in a separate array of lengths (*lens*).

If the area passed is not large enough to contain the entire data record, the error indicator is set to 509 and no data is placed in the output area.

**Parameters**

*numi* **— specified by user, fullword integer**
Number of integers in the list of integers which follows (>=0).

*iary* **— specified by user, array of fullword integers**
Array of integers to be placed in data record.

*numr* **— specified by user, fullword integer**
Number of real values in the list of real values which follows (>=0).

*rary* **— specified by user, array of short floating-point numbers**
Array of reals to be placed in the data record.

*nums* **— specified by user, fullword integer**
Number of strings in the list of strings which follows (>=0).

*swidth* **— specified by user, fullword integer**
Width of string array. The number of bytes between subsequent entries in the list of character strings which follows (>=0)

*lens* **— specified by user, array of fullword integers**
List of lengths, in bytes, of character strings in the array which follows (>=0).

*sary* **— specified by user, variable length character string**
Array of character strings to be placed in the data record. The portion of this array which is used is defined by the *swidth* and *lens* parameters. The *lens* parameter specifies the length of each string in the array and the *swidth* parameter specifies the spacing between subsequent entries in the array of strings.

*mlodr* **— specified by user, fullword integer**
Maximum length, in bytes, of data record to be constructed by the graPHIGS API (>=12).

The application provided area for construction of the data record must be large enough to accommodate the data provided. This consists of three fullwords (12 bytes) of header information and one byte length field per string. Therefore, the length of the data record in bytes can be computed using the following formula:

```
mlodr >= 12 + (4*numi)
+ (4*numr) + nums +
(lens(1) + (lens(2) + ... +lens(nums))
```

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If zero, the request has been completed. Otherwise, one of the following errors has been encountered:

**108** NUMBER OF CHARACTERS IN TEXT STRING < ZERO

**505** LENGTH OF RETURN ARRAY < ZERO

**506** NUMBER OF INITIAL VALUES < ZERO

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**557** WIDTH PARAMETER < MINIMUM ALLOWED

*lodr* **— returned by the graPHIGS API, fullword integer**
Actual length, in bytes, of data record constructed by the graPHIGS API

*datarec* **— returned by the graPHIGS API, variable length character string**
Data record as specified by the input parameters. The returned data record is in the proper format for passing to input device initialization routines.

**Error Codes**

None

**Related Subroutines**

**GPINCH**
Initialize Choice

**GPINLC**
Initialize Locator

**GPINPK**
Initialize Pick

**GPINSK**
Initialize Stroke

**GPINST**
Initialize String

**GPINVL**
Initialize Valuator

**RCP code**

201330958 (X'0C00110E')

## GPRNBS - Reevaluate Non-Uniform B-Spline Surface

**GPRNBS (*uorder*, *vorder*, *unum*, *vnum*, *uknots*, *vknots*, *tflag*, *utdata*, *vtdata*, *cflags*, *cwidth*, *ctlpts*, *umin*, *umax*, *vmin*, *vmax*, *option*, *nelem*)**

**Purpose**

Use **GPRNBS** to reevaluate a large surface into smaller surfaces. This utilty creates *x* number of structure elements to define all the data and returns the number of structure elements created. If an error occurs, processing stops and the actual number of created structure elements is returned.

The reevaluation of the surface is based on the specified option. If the option is specified as `1=CONTROL_POINTS`, then it reevaluates the surface by looking at the number of control points in the *u* and *v* dimension. If the option is specified as `2=ELEMENT_SIZE`, then it reevaluates the surface by looking at the size of the surface. If the element size exceeds 64K bytes, the surface is partitioned such that the resulting surfaces will not exceed 64K bytes.

If the option parameter is not a valid value, it defaults to `1=CONTROL_POINTS`.

**Parameters**

*uorder* **— specified by user, fullword integer**
Order of the basis functions for the *u* parameter (>=2).

*vorder* **— specified by user, fullword integer**
Order of the basis functions for the *v* parameter (>=2).

*unum* **— specified by user, fullword integer**
Number of surface control points for the *u* direction (>= *uorder*).

*vnum* **— specified by user, fullword integer**
Number of surface control points for the *v* direction (>= *vorder*).

*uknots* **— specified by user, array of floating-point numbers**
Knot values for the *u* parameter. The length of this array must be *uorder*+ *unum*. This parameter must be a non-decreasing knot value sequence.

*vknots* **— specified by user, array of floating-point numbers**
Knot values for the *v* parameter. The length of this array must be *vorder*+ *vnum* This parameter must be a non-decreasing knot value sequence.

*tflag* **— specified by user, fullword integer**
Surface tessellation quality value flag. This parameter indicates whether the tessellation quality values are specified or not.

| Value | Meaning |
|---|---|
| **0** | Not specified. |
| **1** | Specified. |

*utdata* **— specified by user, array of short floating-point numbers**
Tesselation quality values for the *u* direction. When the *tflag* parameter is 1 (specified), this parameter must contain *unum*- *uorder*+1 quality values. These values are used in conjunction with Surface Approximation Criteria method 8 to control the number of subdivisions made in the *u* direction. The number of subdivisions performed for a patch is approximately: *value of utdata* [default] *u* (the Surface Approximation Criteria control value in the traversal state list).

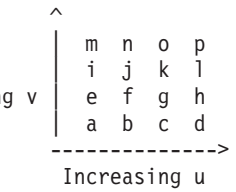
*vdata* **— specified by user, array of short floating-point numbers**
Tessellation quality values for the *v* direction. When the *tflag* parameter is 1 (specified), this parameter must contain *vnum*- *vorder*+1 quality values. These values are used in conjunction with Surface Approximation Criteria method 8 to control the number of subdivisions made in the *v* direction. The number of subdivisions performed for a patch is approximately: *value of vtdata* [default] *v* (the Surface Approximation Criteria control value in the traversal state list).

*cflags* **— specified by user, fullword integer**
Control point option data flags. This parameter indicates what data is specified for each control point. The value specified should be the sum of the following values based on the fields that are specified by the *ctlpts* parameter.

| Value | Meaning |
|---|---|
| **0** | Control point coordinates. |
| **1** | Weights are specified with each control point.This produces the rational form of the Non-Uniform B-Spline Surface. |

*cwidth* **— specified by user, fullword integer**
Number of words between subsequent entries of control points array *ctlpts*.

*ctlpts* **— specified by user, array of short floating-point numbers**
Grid of control points. The control points are stored by row where a row is considered to be the direction associated with the *u* parameter. For example, the set of control points




is stored in the order a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p. The *cwidth* parameter must be at least 3. If *cflags* specifies that weights are included with each control point, the *cwidth* parameter must be at least 4. Each weight W must be greater than zero when specified.

**Note:** When W is specified, the control points are not in homogeneous form (i.e., XW, YW, ZW, W). They are specified after division by W or (X,Y,Z,W)

*umin* **— specified by user, short floating-point number**
The minimum parameter value in the *u* direction at which the surface is evaluated. This value must be greater than or equal to the value of knot *uorder* in the parameter *uknots*.

*umax* **— specified by user, short floating-point number**
The maximum parameter value in the *u* direction at which the surface is evaluated. This value must be less than or equal to the value of knot *unum*+1 in the parameter *uknots*.

*vmin* **— specified by user, short floating-point number**
The minimum parameter value in the *v* direction at which the surface is evaluated. This value must be greater than or equal to the value of *vorder* in the parameter *vknots*.

*vmax* **— specified by user, short floating-point number**
The maximum parameter value in the *v* direction at which the surface is evaluated. This value must be less than or equal to the value of knot *vnum*+1 in the parameter *vknots*.

*option* **— specified by user, fullword integer**
Option to use when reevaluating the surface (`1=CONTROL_POINTS`, `2=ELEMENT_SIZE`).

*nelem* **— returned by the graPHIGS API, fullword integer**
The number of structure elements created by the utility or 0 if none were created.

**Error Codes**

**4** FUNCTION REQUIRES STATE STOP

**341** ORDER OF BASIS FUNCTION < TWO

**342** ORDER IS GREATER THAN NUMBER OF CONTROL POINTS

**343** KNOT VECTOR IS INVALID

**345** WEIGHT IN CONTROL POINT IS <= ZERO

**347** PARAMETER LIMITS ARE OUTSIDE VALID PARAMETER RANGE

**348** MINIMUM PARAMETER LIMIT > MAXIMUM

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**362** TESSELLATION CONTROL VALUE IS INVALID

**557** WIDTH PARAMETER < MINIMUM ALLOWED

**Related Subroutines**

**GPNBS**
Non-Uniform B-Spline Surface

**RCP code**

201348097 (X'0C005401')

## GPROTX - Rotate X

**GPROTX (*angle*, *matrix*)**

**Purpose**

Use **GPROTX** to calculate a rotation matrix around the x-axis using a given angle of rotation.

### Parameters

*angle* **— specified by user, short floating-point number**
Rotation angle in radians.

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

### Error Codes

None

### Related Subroutines

None

### RCP code

201330949 (X'0C001105')

## GPROTY - Rotate Y

**GPROTY (*angle*, *matrix*)**

### Purpose

Use **GPROTY** to calculate a rotation matrix around the y-axis using a given angle of rotation.

### Parameters

*angle* **— specified by user, short floating-point number**
Rotation angle in radians.

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

### Error Codes

None

### Related Subroutines

None

### RCP code

201330950 (X'0C001106')

## GPROTZ - Rotate Z

**GPROTZ (*angle*, *matrix*)**

### Purpose

Use **GPROTZ** to calculate a rotation matrix around the z-axis using a given angle of rotation.

### Parameters

*angle* **— specified by user, short floating-point number**
Rotation angle in radians.

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

### Error Codes

None

### Related Subroutines

None

### RCP code

201330951 (X'0C001107')

## GPROT2 - Rotate 2

**GPROT2 (*angle*, *matrix*)**

### Purpose

Use **GPROT2** to calculate a two-dimensional rotation matrix using a given angle of rotation.

### Parameters

*angle* **— specified by user, short floating-point number**
Rotation angle in radians.

*matrix* **— returned by the graPHIGS API, 9 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13|
|m21 m22 m23| ---> (m11,m12,m13,m21....m33)
|m31 m32 m33|
```

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201330952 (X'0C001108')

## GPRTNS - Reevaluate Trimmed Non-Uniform B-Spline Surface

**GPRTNS (*uorder*, *vorder*, *unum*, *vnum*, *uknots*, *vknots*, *tflag*, *utess*, *vtess*, *cflags*, *cwidth*, *ctlpts*, *ncontour*, *ncurve*, *curveinfo*, *tknot*, *ttess*, *cdwidth*, *cddata*, *option*, *nelem*)**

**Purpose**

Use **GPRTNS** to reevaluate a large trimmed surface into smaller trimmed surfaces. This utilty creates *x* number of structure elements to define all the data and returns the number of structure elements created. If an error occurs, processing stops and the actual number of created structure elements is returned.

The reevaluation of the surface is based on the specified option. If the option is specified as `1=CONTROL_POINTS`, then it reevaluates the surface by looking at the number of control points in the *u* and *v* dimension. If the option is specified as `2=ELEMENT_SIZE`, then it reevaluates the surface by looking at the size of the surface. If either number exceeds 64K bytes, the surface is partitioned such that the resulting surfaces will not exceed 64K bytes.

If the option parameter is not a valid value, it defaults to `1=CONTROL_POINTS`.

**Parameters**

*uorder* **— specified by user, fullword integer**
Order of the basis functions for the *u* parameter (>=2).

*vorder* **— specified by user, fullword integer**
Order of the basis functions for the *v* parameter (>=2).

*unum* **— specified by user, fullword integer**
Number of surface control points for the *u* direction (>= *uorder*).

*vnum* **— specified by user, fullword integer**
Number of surface control points for the *v* direction (>= *vorder*).

*uknots* **— specified by user, array of floating-point numbers**
Knot values for the *u* parameter. The length of this array must be *uorder*+ *unum*. This parameter must be a non-decreasing knot value sequence.

*vknots* **— specified by user, array of floating-point numbers**
Knot values for the *v* parameter. The length of this array must be *vorder*+ *vnum*. This parameter must be a non-decreasing knot value sequence.

*tflag* — **specified by user, fullword integer**
Surface tessellation quality value flag. This parameter indicates whether the tessellation quality values are specified or not.

| Value | Meaning |
|---|---|
| **0** | Not specified. |
| **1** | Specified. |

*utess* — **specified by user, array of short floating-point numbers**
Tesselation quality values for the *u* direction. When the *tflag* parameter is 1 (specified), this parameter must contain *unum*- *uorder*+1 quality values. These values are used in conjunction with Surface Approximation Criteria method 8 to control the number of subdivisions made in the *u* direction. The number of subdivisions performed for a patch is approximately: *value of utess* [default] *u* (the Surface Approximation Criteria control value in the traversal state list).

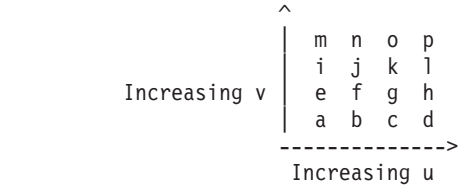
*vtess* — **specified by user, array of short floating-point numbers**
Tessellation quality values for the *v* direction. When the *tflag* parameter is 1 (specified), this parameter must contain *vnum*- *vorder*+1 quality values. These values are used in conjunction with Surface Approximation Criteria method 8 to control the number of subdivisions made in the *v* direction. The number of subdivisions performed for a patch is approximately: *value of vtess* [default] *v* (the Surface Approximation Criteria control value in the traversal state list).

*cflags* — **specified by user, fullword integer**
Control point option data flags. This parameter indicates what data is specified for each control point. The value specified should be the sum of the following values based on the fields that are specified by the *ctlpts* parameter.

| Value | Meaning |
|---|---|
| **0** | Control point coordinates. |
| **1** | Weights are specified with each control point. This produces the rational form of the Non-Uniform B-Spline Surface. |

*cwidth* — **specified by user, fullword integer**
Number of words between subsequent entries of control points array *ctlpts*.

*ctlpts* — **specified by user, array of short floating-point numbers**
Grid of control points. The control points are stored by row where a row is considered to be the direction associated with the *u* parameter. For example, the set of control points




is stored in the order a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p. The *cwidth* parameter must be at least 3. If *cflags* specifies that weights are included with each control point, the *cwidth* parameter must be at least 4. Each weight W must be greater than zero when specified.

**Note:** When W is specified, the control points are not in homogeneous form (i.e., XW, YW, ZW, W). They are specified after division by W or (X,Y,Z,W).

*ncontour* — **specified by user, fullword integer**
Number of contours to be generated (>=0).

*ncurve* — **specified by user, fullword integer**
Number of curves in each contour. Each entry must be greater than or equal to 1. The length of this array is defined by the value of *ncontour* parameters.

*curveinfo* **— specified by user, array of 6 fullword data**
Array containing information about each curve. Each entry of this parameter must have the following fields in this order:

**type of curve**
— fullword integer

This parameter specifies various options of the curve. Each option is specified by a bit in this word and the following bits are currently defined:

| Bit | Meaning |
|---|---|
| **0-28** | Reserved. Must be set at 0. |
| **29** | Tessellation quality flag. If set, a tessellation quality value for each span of this curve is specified in the *ttess* parametr. |
| **30** | Weight flag. If set, the curve is rational and the weight is specified for each control point in the *cddata* parameter. |
| **31** | Boundary flag. If set, the curve is treated as an edge of the composite fill area. |

**order** — fullword integer

Order of the curve (>=2)

**number of data**
— fullword integer

Number of entries of the *cddata* parameter used to define the curve. This parameter corresponds to the *npoint* parameter of the Non-Uniform B-Spline Curve 2. The specified entries of numbers of the *cddata* parameter are used as its *ctlpts* parameter.

**start** — short floating-point number

The parameter value representing the start point of the curve.

**end** — short floating-point number

The parameter value representing the end point of the curve.

*tknot* **— specified by user, array of short floating-point numbers**
Array of knot values for the *t* direction of the curve. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo*. The length of each list is equal to the *order*+ *number of data* for the curve.

*ttess* **— specified by user, array of short floating-point numbers**
Array of tessellation quality values. This array must contain one list of each Non-Uniform B-Spline Curve with a tessellation quality flag set to ON. For other curves, this array is not referenced. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo* The length of each list is equal to the *number of data*- *order*+1 of the curve.

*cdwidth* **— specified by user, fullword integer**
Specifies the number of fullwords between each entry of the array in *cddata* If there is any rational curve in the *curveinfo* parameter, this parameter must be at least 3. Otherwise, it must be larger than or equal to 2.

*cddata* **— specified by user, array of short floating-point numbers**
This array must contain one list for each curve. The sequence of each list in this array is assumed to match the order of the curve definitions in *curveinfo*. The length of each list is equal to the number of definition data specified by the *curveinfo* parameter.

For each entry, the following fields are defined and the fields must be specified in this order without any gap.

***u, v* components**
— two short floating-point numbers

**weight**
— short floating-point number

*option* **— specified by user, fullword integer**
Option to use when reevaluating the surface (`1=CONTROL_POINTS, 2=ELEMENT_SIZE`).

*nelem* **— returned by the graPHIGS API, fullword integer**
The number of structure elements created by the utility, or zero if none were created.

### Error Codes

**4** FUNCTION REQUIRES STATE STOP

**341** ORDER OF BASIS FUNCTION < TWO

**342** ORDER IS GREATER THAN NUMBER OF CONTROL POINTS

**343** KNOT VECTOR IS INVALID

**345** WEIGHT IN CONTROL POINT IS <= ZERO

**347** PARAMETER LIMITS ARE OUTSIDE VALID PARAMETER RANGE

**348** MINIMUM PARAMETER LIMIT > MAXIMUM

**351** OPTIONAL DATA AVAILABILITY FLAG IS INVALID

**353** NUMBER OF CONTOURS < ZERO

**354** NUMBER OF CURVES PER CONTOUR < ONE

**361** CURVE OPTIONS FIELD IS INVALID

**362** TESSELLATION CONTROL VALUE IS INVALID

**557** WIDTH PARAMETER < MINIMUM ALLOWED

### Related Subroutines

**GPTNBS**
Trimmed Non-Uniform B-Spline Surface

### RCP code

201348098 (X'0C005402')

## GPSC2 - Scale 2

**GPSC2 (*scale*, *matrix*)**

### Purpose

Use **GPSC2** to calculate a two-dimensional scaling matrix using a given scaling vector containing *x* and *y* scale factors.

### Parameters

*scale* **— specified by user, 2 short floating-point numbers**
Scale factors in *x* and *y* directions.

*matrix* **— returned by the graPHIGS API, 9 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13|
|m21 m22 m23| ---> (m11,m12,m13,m21.....m33)
|m31 m32 m33|
```

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201330948 (X'0C001104')

## GPSC3 - Scale 3

**GPSC3 (*scale*, *matrix*)**

**Purpose**

Use **GPSC3** to calculate a three-dimensional scaling matrix using a given three-dimensional vector containing *x*, *y*, *z* scale factors.

**Parameters**

*scale* **— specified by user, 3 short floating-point numbers**
Scale factors for *x*, *y*, *z* directions.

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201330947 (X'0C001103')

# GPTRL2 - Translate 2

**GPTRL2 (*vector*, *matrix*)**

### Purpose

Use **GPTRL2** to calculate a two-dimensional translation matrix using a two-dimensional, translation vector containing *x*, *y* translation components.

### Parameters

*vector* **— specified by user, 2 short floating-point numbers**
Translation vector containing *x*, *y* translation components.

*matrix* **— returned by the graPHIGS API, 9 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13|
|m21 m22 m23| ---> (m11,m12,m13,m21....m33)
|m31 m32 m33|
```

### Error Codes

None

### Related Subroutines

None

### RCP code

201330946 (X'0C001102')

# GPTRL3 - Translate 3

**GPTRL3 (*vector*, *matrix*)**

### Purpose

Use **GPTRL3** to calculate a three-dimensional translation matrix using a given three-dimensional, translation vector containing *x*, *y*, *z* translation components.

### Parameters

*vector* **— specified by user, 3 short floating-point numbers**
Translation vector containing *x*, *y*, *z* translation components.

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Transformation matrix.

The elements are returned in the following order for the transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201330945 (X'0C001101')

## GPVPLN - Set View Plane Normal

**GPVPLN (*normal*)**

**Purpose**

Use **GPVPLN** to set the View Plane Normal in the USL (Utility State List) to the value specified.

When the Compute View Matrix (**GPCVMT**) subroutine is called, this vector serves as the n-axis of the Viewing Coordinate (VC) system. For more information on the Viewing Coordinate system, see *The graPHIGS Programming Interface: Understanding Concepts*.

**Parameters**

*normal* — **specified by user, 3 short floating-point numbers (WC)**
Directional components of view plane normal in World Coordinates (WC).

**Error Codes**

**56** SPECIFIED VECTOR HAS LENGTH ZERO

**Related Subroutines**

None

**RCP code**

201331202 (X'0C001202')

## GPVR - Set View Reference Point

**GPVR (*point*)**

**Purpose**

Use **GPVR** to set the View Reference Point in the USL (Utility State List) to the value specified.

When the Compute View Matrix (**GPCVMT**) subroutine is called, this point serves as the origin of the Viewing Coordinate (VC) system and is specified in World Coordinates (WC). For more information on the Viewing Coordinate system, see *The graPHIGS Programming Interface: Understanding Concepts*.

**Parameters**

*point* **— specified by user, 3 short floating-point numbers (WC)**
View reference point.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201331201 (X'0C001201')

## GPVUP - Set View Up

**GPVUP (*vector*)**

**Purpose**

Use **GPVUP** to set the View Up vector in the USL (Utility State List) to the value specified.

When the Compute View Matrix (**GPCVMT**) subroutine is called, this vector identifies the positive side of the v-axis of the Viewing Coordinate (VC) system. For more information on the Viewing Coordinate system, see *The graPHIGS Programming Interface: Understanding Concepts*.

**Parameters**

*vector* **— specified by user, 3 short floating-point numbers (WC)**
Directional components of view up vector in World Coordinates (WC).

**Error Codes**

**56** SPECIFIED VECTOR HAS LENGTH ZERO

**Related Subroutines**

**GPCVMT**
Compute View Matrix

**GPVPLN**
Set View Plane Normal

**GPVR** Set View Reference Point

**RCP code**

201331203 (X'0C001203')

## GPXF2 - Transform Point 2

**GPXF2 (*point*, *matrix*, *result*)**

### Purpose

Use **GPXF2** to transform a point using a specified transformation matrix.

The graPHIGS API returns the result of multiplying the given point by the transformation.

### Parameters

*point* **— specified by user, 2 short floating-point numbers**
Point to be transformed.

*matrix* **— specified by user, 9 short floating-point numbers**
Transformation matrix.

The elements must be passed in the following order for the transformation matrix:

```
|m11 m12 m13|
|m21 m22 m23| ---> (m11,m12,m13,m21....m33)
|m31 m32 m33|
```

*result* **— returned by the graPHIGS API, 2 short floating-point numbers**
Transformed point.

### Error Codes

None

### Related Subroutines

None

### RCP code

201330957 (X'0C00110D')

## GPXF3 - Transform Point 3

**GPXF3 (*point*, *matrix*, *result*)**

### Purpose

Use **GPXF3** to transform a point using a specified transformation matrix.

The graPHIGS API returns the result of multiplying the given point by the transformation.

### Parameters

*point* **— specified by user, 3 short floating-point numbers**
Point.

*matrix* **— specified by user, 16 short floating-point numbers**
Transformation matrix.

The elements must be passed in the following order for the transformation matrix:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24| ---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

*result* **— returned by the graPHIGS API, 3 short floating-point numbers**
Transformed point.

### Error Codes

None

### Related Subroutines

None

### RCP code

201330956 (X'0C00110C')

# Chapter 15. Error Handling Subroutines

The subroutines in this category allow your application to modify the error handling characteristics of the graPHIGS API system. By default, the error handler provided by the graPHIGS API is given control when an error is detected. However, the application can provide an alternative error handler which supersedes the default and receives control when an error is detected.

If desired, the Set Error Handling Mode (**GPEMO**) subroutine can be used to set error handling `1=OFF`. When error handling is `1=OFF`, processing continues until an ABEND condition is reached. Generally, error handling should remain `2=ON` in a program development environment. Error handling is `2=ON` by default.

See *The graPHIGS Programming Interface: Understanding Concepts* for details on error handlers and their operation.

See *The graPHIGS Programming Interface: Writing Applications* for information on the techniques used to code application error handlers in specific languages.

See *The graPHIGS Programming Interface: Messages and Codes* for information on specific error conditions and system responses - severity levels, etc.

## GPEHND - Define Error Handling Subroutine

**GPEHND (*error-handler*)**

### Purpose

Use **GPEHND** to specify the address of an application-defined first-level error handling routine.

The graPHIGS API gives control to this routine when an error is detected. If an application error handler is not defined, the graPHIGS API uses the default, which logs an error in the file specified by the first parameter of the Open graPHIGS (**GPOPPH**) subroutine. A first-level error handler can only invoke Inquiry subroutines and the Error Logging (**GPELOG**) subroutine.

The application error handler receives six parameters, when invoked.

*number* **— returned by the graPHIGS API, fullword integer**
Error number.

*name* **— returned by the graPHIGS API, 8-byte character string**
Identification of the graPHIGS API procedure causing the error.

*file* **— returned by the graPHIGS API, 8-byte character string**
Error file identifier.

*wsid* **— returned by the graPHIGS API, fullword integer**
Workstation identifier associated with the error.

*flag* **— returned by the graPHIGS API, fullword integer**

**Bits 0-29** Reserved.

**Bit 30** Specifies if the error was generated by a subroutine performed at a specific workstation. If set, the *wsid* parameter contains the workstation identifier. Otherwise the *wsid* parameter is not valid.

**Bit 31** If set, it indicates that there are more errors waiting for the current graPHIGS API subroutine.

*severity* **— set by error handler, fullword integer**
Error severity (initialized to zero by the graPHIGS API).

**Parameters**

*error-handler* **— specified by user, fullword integer**
The address of the routine receiving control when an error is encountered.

The default error handler can be reset by specifying an error handler address of a fullword binary zero.

A user supplied error handler receives the parameter list shown above *only* if the application is using the non-reentrant interface. If it is using the reentrant interface, the error handler receives the Application Anchor Block (AAB) as its first parameter, followed by the other parameters. If the application is using the System Programmer's Interface (SPI), it receives the following data: the AAB; a Request Control Parameter code (RCP), which it should ignore *and must not corrupt* (this code provides the only way to return to the program once error processing is completed[default] and the remaining parameters. The error handler has one special consideration. It is possible for an application to mix SPI and re-entrant subroutines. If it does, the error handler receives varying length parameter lists. The type of list passed by the application depends on the type of the current graPHIGS API subroutine. The only way the error handler can differentiate between interfaces is by checking the number of parameters and by searching for the End of List marker. See *The graPHIGS Programming Interface: Writing Applications*.

In effect, mixing the two types of interface subroutines is practical only if the language in which the error handler is written allows the number of parameters to be determined.

**Error Codes**

None

**Related Subroutines**

| | |
|---|---|
| **GPELOG** | Error Logging |
| **GPEXIT** | Specify an Error Exit and Error Threshold |
| **GPOPPH** | Open graPHIGS |

**RCP code**

201337857 (X'0C002C01')

# GPELOG - Error Logging

**GPELOG (*file*)**

**Purpose**

Use **GPELOG** to print an error message to the specified error file. When invoked, the error which caused the first-level error handler to be called is logged to the specified file. If the specified file is blank or cannot be opened, then the error is logged to the console from which the application was started.

This subroutine is available only to a first-level error handler.

**Parameters**

*file* **— specified by user, 8-byte character string**
Error file identification.

### Error Codes

**521** NOT IN ERROR STATE

### Related Subroutines

**GPEHND** Define Error Handling Subroutine

### RCP code

201337858 (X'0C002C02')

## GPEMO - Set Error Handling Mode

**GPEMO (*mode*)**

### Purpose

Use **GPEMO** to enable or disable graPHIGS API error handling.

The error handling mode in the Error State List (ESL) is set to the value specified. If the error handling mode is `1=OFF`, then the errors detected by the graPHIGS API are ignored. The default error handling mode is `2=ON`.

### Parameters

*mode* **— specified by user, fullword integer**
Error handling mode (`1=OFF`, `2=ON`).

### Error Codes

**94** ERROR HANDLING MODE IS INVALID

### Related Subroutines

**GPQEMO** Inquire Error Handling Mode

### RCP code

201337860 (X'0C002C04')

## GPEXIT - Specify an Error Exit and Error Threshold

**GPEXIT (*error-routine*, *severity*)**

### Purpose

Use **GPEXIT** to specify a second-level error handler to receive control at the end of a graPHIGS API subroutine call if an error of at least the specified severity is encountered.

The severity of an error can be assigned by the first-level error handler (see Define Error Handling [**GPEHND**] [page GPEHND - Define Error Handling Subroutine]) subroutine, which receives control before the second-level error handler.

If a value of zero or less is specified for the *severity* parameter of this subroutine, the error exit is invoked after every call to the graPHIGS API If a severity level of 17 or more is specified, the error exit is never invoked.

When the error severity threshold is zero, your second-level error exit routine should perform the following steps to prevent infinite loops:

1. Set the error severity threshold to 17 using **GPEXIT** to prevent the graPHIGS API from calling your second-level error exit.
2. Call any graPHIGS API subroutine. Note that because the severity threshold is 17, your second-level error exit will not be called again.
3. Reset the error severity threshold to zero before returning control to the graPHIGS API. Note that the graPHIGS API will not transfer control to your second-level error exit after a call to **GPEXIT.**

The default error exit can be reset by specifying an error exit address consisting of a fullword binary zero.

Control is returned to the application statement following the one which invoked the graPHIGS API A second-level error handler can call any graPHIGS API subroutine except the Error Logging (**GPELOG**) subroutine.

### Parameters

*error-routine* **— specified by user, fullword integer**
: The address of the routine receiving control when an error of the specified severity or greater is encountered.

*severity* **— specified by user, fullword integer**
: A fullword integer indicating the minimum level of severity (severity threshold) at which the error routine is called.

### Error Codes

None

### Related Subroutines

**GPEHND**
: Define Error Handling Subroutine

### RCP code

201337862 (X'0C002C06')

# Chapter 16. Miscellaneous Subroutines

This section contains the definition of two miscellaneous subroutines. The first is an escape routine used by the graPHIGS API for special functions such as setting plot sizes. The second subroutine is used to read the contents of a workstation's frame buffer.

## GPES - Escape

**GPES (*funcid*, *lidr*, *idr*, *mlodr*, *lodr*, *odr*)**

### Purpose

Use **GPES** to perform an escape function. The specific escape subroutine is identified by way of the identifier parameter. In general, an escape subroutine accepts both an input data record and an output data record to place any output generated by the escape subroutine.

If the specified escape identifier is not supported by the graPHIGS API, then the subroutine may be ignored or produce unexpected results.

### Parameters

*funcid* **— specified by user, fullword integer**
Identifier of the escape to be performed.

**1001 - Sound Alarm:** This escape subroutine causes the specified workstation to sound its alarm. If the workstation does not have an alarm or does not support this escape subroutine, then the escape is ignored. The input data record for this escape code contains the workstation identifier at which to perform this function. There is no output data record generated by this function.

**1002 - Enable/Disable Link Switch Notification:** When the link switch is modified on a 5080 workstation, this escape subroutine enables or disables the generation of events. The 5080 operator has the capability to switch between a RT PC and a System/370. This escape subroutine may be used to enable application notification of this operator action. Notification is disabled by default. Notification occurs by way of the graPHIGS API event queue. When the switch is changed, an event is placed on the event queue. The event type identifies the direction of the switch. Event type 101 indicates that the 5085 link has been switched away from the application program. Event type 102 indicates that the 5085 link has been switched to the application program. The input data record for this escape code contains the workstation identifier at which to perform this function and a flag which indicates whether this capability is being enabled or disabled. This escape subroutine is only valid for 5080 workstations. When specified with a workstation type other than a 5080, this escape is ignored. There is no output data record generated by this function.

**1003 - GDF/CGM Plot Size:** This escape subroutine allows the user to specify the width of the plotted output in meters. Zero and negative sizes are not valid. The input data record for this escape code contains the identifier for the workstation at which to perform this function and the value, in floating-point representation, for the size. This escape subroutine is only valid for workstations of type 'GDF' or 'CGM'. When specified with a workstation type other than 'GDF' or 'CGM', this escape is ignored. No output data record is generated by this function.

CGM plot scaling is implemented by adjusting the scale factor metric to be equal to 960 * plot size/1000. This is the distance that corresponds to one Virtual device coordinate (VDC) unit. When you change the plot size, values stored in the workstation description table are not changed to reflect the new display size. The nominal line width, nominal edge width, and marker size are not scaled.

GDF scaling is accomplished via a comment order which is understood by the IBM GDF interpreters which are supplied by the graPHIGS API program product. Other interpreters will most likely ignore this information but will not generate an error. If you are designing your own interpreter and you wish to utilize the scaling comment order, note that it consists of the following 18 bytes. (For more detailed information, see the *GDDM Base Programming Reference, Volume 2*).

| Field Length | Content | Meaning |
|---|---|---|
| 1 | X'01' | Comment order. |
| 1 | X'10' | Length of following data. |
| 2 | 0000 | Reserved. |
| 8 | 'graPHIGS' | Application identifier (*char* data). |
| 1 | X'01' | Indicates exact scale comment. |
| 1 | X'04' | Length of following data. |
| 4 | 4 byte floating-point | Size in meters of the *x* dimension of the resulting GDF plot. |

**1009 - Window Resize Notification Control:** This escape subroutine allows your application to be notified when the window containing a mapped display surface changes size. This escape subroutine also allows your application to specify whether it wants the graPHIGS API to redraw the contents of the window when the window is resized. This escape subroutine is meaningful only on a workstation which uses the facilities of a window system (e.g., X Windows).

The mapped display surface is the subarea of the window that the workstation uses as the workstation's display surface for graphical output and input. The mapped display surface size may change if the user changes the size of the window that contains the mapped display surface. Your application can receive notification of such size changes by enabling this Window Resize Notification function. Notification occurs by way of the graPHIGS API event queue. The new size of the mapped display surface can then be obtained using the Inquire Mapped Display Surface Size (**GPQMDS**) subroutine. By default, notification is disabled and the window contents will always be redrawn when a resize occurs. The input data record for this escape code contains the workstation identifier at which to perform this function. There is no output data record generated by this function.

**1011 - Window Exposure Notification Control:** This escape subroutine allows your application to be notified when the window containing a mapped display surface is exposed. This escape subroutine also allows your application to specify whether it wants the graPHIGS API to redraw the contents of the screen when the exposure occurs. This escape subroutine is meaningful only on a workstation which uses the facilities of a window system (e.g., X Window System).

Your application can receive notification each time the window is exposed by enabling this Window Exposure Notification function. Notification occurs by way of the graPHIGS API event queue. By default, notification is disabled and redraw will occur. This function causes generation of event class 106 (Window Exposure Event), retrieved by the Get Window (**GPGWIN**) subroutine. The input data record for this subroutine code contains the workstation identifier at which to perform this function. There is no output data record generated by this function.

**1012 - Window Deletion Notification Control:** This escape subroutine allows your application to be notified when the window is deleted (closed) via the Window Manager. (See *The graPHIGS Programming Interface: Technical Reference* for more details on this function).

**1014 - Workstation-Dependent Output:** This escape subroutine allows your application to generate workstation-dependent output. The graPHIGS API can send this data directly to the workstation. This escape subroutine is valid only for workstations of type 'CGM'. When specified with a workstation type other than 'CGM', this escape is ignored.

The input data record contains the workstation identifier, which specifies where the graPHIGS API performs the subroutine, and the output data, which is *not* verified by the workstation. The graPHIGS API does *not* guarantee the validity of the data. It is the application's responsibility to ensure that the data is valid (e.g., proper length(s), identifiers, padding, etc.). When character encoding, floating-point, and/or byte ordering differences exist between the target workstation and the application environment, the application must ensure the validity of this data also. For information on how the application can ensure the validity of this data, see the Convert Data (**GPCVD**) subroutine. For information on CGM data, see the "CGM Workstation" in *The graPHIGS Programming Interface: Technical Reference*. There is no output data record generated by this function.

*lidr* **— specified by user, fullword integer**
Length of the input data record in bytes (0<=*lidr*<=32K).

*idr* **— specified by user, variable length character string**
Input data record. Escape subroutine identifier 1001 (sound alarm) requires the following input data record:

```
-----------------------------
|            1              |  number of integers  (fullword integer)
|-----------------------------|
|            0              |  number of reals     (fullword integer)
|-----------------------------|
|            0              |  number of strings   (fullword integer)
|-----------------------------|
|  workstation identifier    |                      (fullword integer)
-----------------------------
```

Escape subroutine identifier 1002 (link switch notification) requires the following data record:

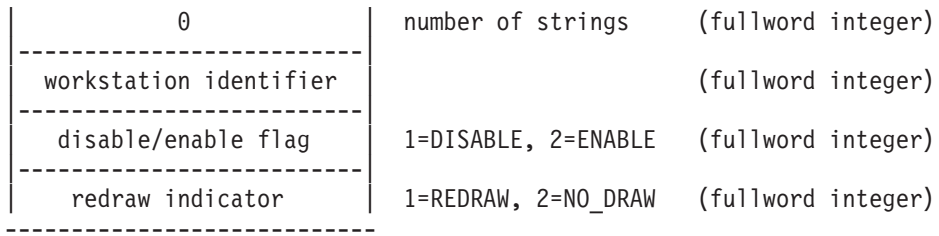
```
    -----------------------------
 0  |            2              |  number of integers  (fullword integer)
    |-----------------------------|
 4  |            0              |  number of reals     (fullword integer)
    |-----------------------------|
 8  |            0              |  number of strings   (fullword integer)
    |-----------------------------|
12  |  workstation identifier    |                      (fullword integer)
    |-----------------------------|
16  |   disable/enable flag      |  1=DISABLE, 2=ENABLE (fullword integer)
    -----------------------------
```

Escape subroutine identifier 1003 (GDF/CGM plot size) requires the following data record:

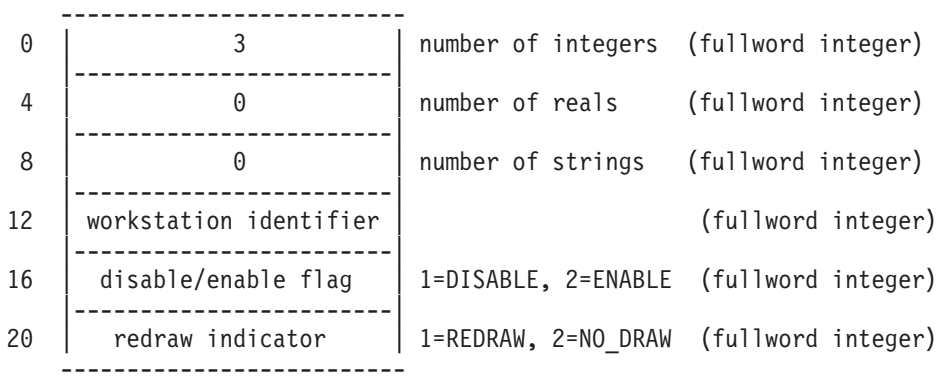
```
    -----------------------------
 0  |            1              |  number of integers    (fullword integer)
    |-----------------------------|
 4  |            1              |  number of reals       (fullword integer)
    |-----------------------------|
 8  |            0              |  number of strings     (fullword integer)
    |-----------------------------|
12  |  workstation identifier    |                        (fullword integer)
    |-----------------------------|
16  |     plot size value        |                        (real)
    -----------------------------
```

Escape subroutine identifier 1009 (Window Resize Notification Control) requires the following data record:

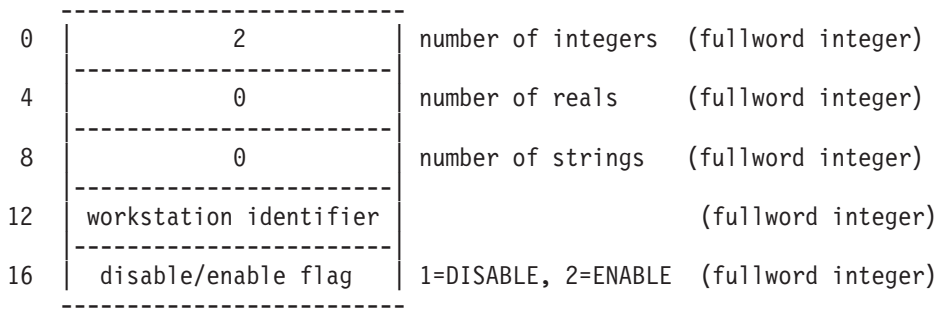
```
    -----------------------------
 0  |            3              |  number of integers    (fullword integer)
    |-----------------------------|
 4  |            0              |  number of reals       (fullword integer)
    |-----------------------------|
```



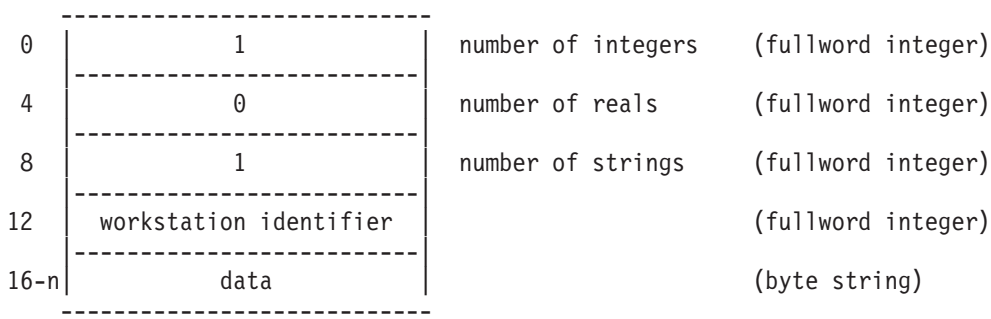

Escape subroutine identifier 1011 (Window Exposure Notification Control) requires the following data record:




Escape subroutine identifier 1012 (Window Deletion Notification Control) requires the following data record:




Escape subroutine identifier 1014 (Workstation-Dependent Output) requires the following data record:




*mlodr* **— specified by user, fullword integer**
Maximum length of the output data record to be returned by the graPHIGS API in bytes.

*lodr* **— returned by the graPHIGS API, fullword integer**
Length of the output data record returned by the graPHIGS API in bytes.

*odr* **— returned by the graPHIGS API, variable length character string**
Output data record.

### Error Codes

**146** FIELD IN INPUT DEVICE DATA RECORD IN ERROR

**501** DATA RECORD WAS NOT SPECIFIED BUT IS REQUIRED

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**527** ESCAPE FUNCTION NOT AVAILABLE

**2050** INSUFFICIENT DATA LEN *n1* FOR CGM WDO

**2051** DATA LEN *n1* > 32771 FOR CGM WDO

**2052** DATA LEN *n1* <> ENCODED LEN *n2* + HDRSZ *n3* IN CGM WDO - USING ENCODED LEN

### Related Subroutines

**GPGWIN**
Get Window

**GPQES**
Inquire List of Available Escape Subroutines

**GPQMDS**
Inquire Mapped Display Surface Size

### RCP code

201340673 (X'0C003701')

## GPRDEV - Redrive Events

**GPRDEV**

### Purpose

Use **GPRDEV** to have the graPHIGS API handle any available events for its X workstations and other nucleus resources. This subroutine is intended to be used in conjunction with the graPHIGS SYNCPROC default. See *The graPHIGS Programming Interface: Technical Reference* for further information on the SYNCPROC default and the ramifications of its use.

### Parameters

None

### Error Codes

None

### Related Subroutines

**GPQSID**
Inquire List of Socket Identifiers

### RCP code

201338118 (X'0C002D06')

# GPRDFB - Read Frame Buffer

**GPRDFB (*wsid*, *frame*, *sorigin*, *size*, *format*, *parm*, *torigin*, *data*)**

### Purpose

Use **GPRDFB** to read pixel data from one of the workstation's frame buffer components into your target application image data. The specified bit depth of your target application image data must match the bit depth of the workstation's frame buffer component that contains the pixel data.

Use the Inquire Frame Buffer Characteristics (**GPQFBC**) inquiry subroutine to determine your workstation's frame buffer characteristics, in particular the bit depth of the workstation's frame buffer components.

The specified target rectangle need not be inside the target application image, but any portion of the target rectangle outside the image will be clipped. The pixel data returned is workstation dependent when all or part of the source rectangle is not visible. It is the application's responsibility to serialize access to the workstation during the read frame buffer operation. This means that other application processes should not attempt to modify the workstation while the read frame buffer is being performed. To guarantee the state of the frame buffer that is returned by this subroutine call (i.e., to ensure that all pending updates are in the returned data), the application should issue an Update Workstation (**GPUPWS**) subroutine before issuing this function.

This subroutine is assigned escape identifier 1007.

> **Note:** This subroutine is an escape subroutine and therefore may not be available on all workstations. Use the Inquire List of Available Escape Subroutines (**GPQES**) subroutine to determine if this subroutine is supported by an open workstation. If not supported, an error will be generated.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*frame* **— specified by user, fullword integer**
Frame buffer component number. This parameter must be a valid frame buffer component of the workstation.

*sorigin* **— specified by user, two fullword integers**
Source rectangle origin (*x*, *y*) (>=0).

*size* **— specified by user, two fullword integers**
Rectangle size (*SX*, *SY*) (>=1).

*format* **— specified by user, fullword integer**
Application image format (`1=PIXEL_ARRAY`).

*parm* **— specified by user, variable data**
Format dependent parameters. The image format 1 requires the following parameters:

- bit depth — fullword integer (1, 2, 4, 8, or 16). This bit depth must match the frame buffer component bit depth.
- *x* size — fullword integer (>=1)
- *y* size — fullword integer (>=1)
- pixel order — fullword integer (`1=LEFT_RIGHT_BOTTOM_TOP`, `2=LEFT_RIGHT_TOP_BOTTOM`)

The product of the *x* size and the bit depth must to a multiple of 8. This is to ensure that each row of the application's image data starts on a byte boundary.

*torigin* **— specified by user, two fullword integers**
Target rectangle origin (*x*, *y*) (>=0).

*data* **— returned by the graPHIGS API, array of pixels**
Target application image data. The graPHIGS API returns the pixels within the intersection of the target application image data and the target rectangle.

The target application image data with bit depth of 16 will be handled as 16-bit halfwords. For all other bit depths, the application image data will be treated as 8-bit unsigned characters.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**119** FRAME BUFFER COMPONENT NUMBER IS INVALID

**236** RECTANGLE DEFINITION IS INVALID

**237** SPECIFIED APPLICATION IMAGE FORMAT IS NOT SUPPORTED

**240** APPLICATION IMAGE DESCRIPTION IS INVALID

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

**527** ESCAPE FUNCTION NOT AVAILABLE

**Related Subroutines**

**GPQAI**
Inquire List of Available Application Image Formats

**GPQFBC**
Inquire Frame Buffer Characteristics

**GPQIBC**
Inquire Image Board Characteristics

**GPQIBF**
Inquire Image Board Facilities

**GPQIMC**
Inquire Image Mapping Characteristics

**GPQIMF**
Inquire Image Mapping Facilities

**RCP code**

201346312 (X'0C004D08')

# Chapter 17. Inquire Subroutines

Inquiry programming subroutines allow application programs to obtain the following information:

- Default system characteristics
- Current state of the system
- Default workstation characteristics
- Current state of a workstation
- Configuration of a workstation
- Structure existence and relationships
- Structure content
- Nucleus characteristics
- Error state and message content

For all inquire subroutines, the parameters listed *prior to* the *errind* parameter are specified by the user; *errind* and the parameters which follow are returned by the graPHIGS API.

## WSL Inquiries

The subroutines in this section have a first parameter of workstation identifier (*wsid*). These subroutines inquire information from the Workstation State List (WSL) corresponding to the specified workstation identifier. The WSL contains the current state of the workstation. To inquire the general characteristics and capabilities of the workstation type, see "WDT Inquiries."

**GPQAR**
Inquire Set of Associated Roots.

**GPQBKS**
Inquire Break Action State.

**GPQCCH**
Inquire Color Table Characteristics.

**GPQCH**
Inquire Choice Device State.

**GPQCID**
Inquire List of Color Table Identifiers.

**GPQCML**
Inquire Color Model.

**GPQCPR**
Inquire Color Processing Representation.

**GPQCSR**
Inquire Cull Size Representation.

**GPQCVE**
Inquire Current View Table Entries Input.

**GPQCVO**
Inquire Current View Table Entries Output.

**GPQCVR**
Inquire Current View Representation.

**GPQDMR**
Inquire Data Mapping Representation.

**GPQDCR**
Inquire Depth Cue Representation.

**GPQDV**
Inquire Deferral and Update State Values.

**GPQFO**
Inquire Active Fonts.

**GPQGFC**
Inquire Geometric Font Characteristics.

**GPQHLF**
Inquire Highlighting Filter.

**GPQHR**
Inquire Hatch Representation.

**GPQICH**
Inquire Image Characteristics.

**GPQICS**
Inquire Input Character Set.

**GPQID**
Inquire Input Device State.

**GPQIMC**
Inquire Image Mapping Characteristics.

**GPQIMI**
Inquire Image Mapping of Image.

**GPQIMV**
Inquire Image Mapping on View.

**GPQIMW**
Inquire Image Mapping on Workstation.

**GPQITS**
Inquire Input Device Trigger State.

**GPQIVF**
Inquire Invisibility Filter.

**GPQIW**
Inquire List of Images on the Workstation.

**GPQLC**
Inquire Locator Device State.

**GPQLSR**
Inquire Light Source Representation.

**GPQLTR**
Inquire Linetype Representation.

**GPQMDS**
Inquire Mapped Display Surface Size.

**GPQMTR**
Inquire Marker Type Representation.

**GPQPAR**
Inquire Pattern Representation.

**GPQPK**
Inquire Pick Device State.

**GPQPKA**
Inquire Pick Aperture.

**GPQRCT**
Inquire Realized Connection Type.

**GPQRV**
Inquire Set of Roots in View.

**GPQRVE**
Inquire Requested View Table Entries Input.

**GPQRVO**
Inquire Requested View Table Entries Output.

**GPQRVR**
Inquire Requested View Representation.

**GPQSK**
Inquire Stroke Device State.

**GPQST**
Inquire String Device State.

**GPQVL**
Inquire Valuator Device State.

**GPQVR**
Inquire Set of Views Which Contain Root.

**GPQWSU**
Inquire Workstation Storage Utilization.

**GPQWSX**
Inquire Workstation Transformation.

**GPQXAF**
Inquire Extended Annotation Font Characteristics.

**GPQXCR**
Inquire Extended Color Representation.

**GPQXER**
Inquire Extended Edge Representation.

**GPQXIR**
Inquire Extended Interior Representation.

**GPQXLR**
Inquire Extended Polyline Representation.

**GPQXMR**
Inquire Extended Polymarker Representation.

**GPQXTR**
Inquire Extended Text Representation.

## WDT Inquiries

Subroutines with a first parameter of workstation type (*wstype*) inquire information from the Workstation Description Table (WDT) corresponding to the specified workstation type. For each workstation you may inquire information from its generic or actual descriptor table. The generic descriptor table contains the maximum capabilities of the workstation, whereas the actual descriptor table contains the realized capabilities of the workstation after it is opened. To be able to inquire the realized capabilities of a workstation, you must supply the realized workstation type on the inquiry. Use the Inquire Realized Connection and Type (**GPQRCT**) subroutine to obtain the realized workstation type.

**GPQAAF**
Inquire Advanced Attribute Facilities.

**GPQAMO**
Inquire Available Antialiasing Modes.

**GPQANF**
Inquire Annotation Facilities.

**GPQART**
Inquire Rendering Targets.

**GPQBK**
Inquire Break Capabilities.

**GPQCDF**
Inquire Curve Display Facilities.

**GPQCF**
Inquire Color Facilities.

**GPQCPF**
Inquire Color Processing Facilities.

**GPQCQM**
Inquire Available Color Quantization Methods.

**GPQCSF**
Inquire Cull Size Facilities.

**GPQCUF**
Inquire Cursor Facilities.

**GPQDBK**
Inquire Default Break Action.

**GPQDCF**
Inquire Depth Cue Facilities.

**GPQDCH**
Inquire Default Choice Device Data.

**GPQDDV**
Inquire Default Deferral State Values.

**GPQDIT**
Inquire Default Input Device Triggers.

**GPQDLC**
Inquire Default Locator Device Data.

**GPQDPK**
Inquire Default Pick Device Data.

**GPQDS**
Inquire Maximum Display Surface Size.

**GPQDSK**
Inquire Default Stroke Device Data.

**GPQDST**
Inquire Default String Device Data.

**GPQDVL**
Inquire Default Valuator Device Data.

**GPQEF**
Inquire Edge Facilities.

**GPQES**
Inquire List of Available Escape Subroutines.

**GPQFBC**
Inquire Frame Buffer Characteristics.

**GPQFP**
Inquire Font Pool Size.

**GPQGD**
Inquire List of Generalized Drawing Primitives.

**GPQGDP**
Inquire Generalized Drawing Primitive.

**GPQGSE**
Inquire List of Available GSEs.

**GPQHD**
Inquire Maximum Hierarchy Depth.

**GPQHF**
Inquire Hatch Facilities.

**GPQHMO**
Inquire Available HLHSR Modes.

**GPQIDD**
Inquire Input Device Description.

**GPQIDF**
Inquire Image Definition Facilities.

**GPQIF**
Inquire Interior Facilities.

**GPQIMF**
Inquire Image Mapping Facilities.

**GPQISF**
Inquire Input Character Set Facilities.

**GPQIT**
Inquire Input Trigger Capabilities.

**GPQLCF**
Inquire List of Color Facilities.

**GPQLI**
Inquire List of Logical Input Devices.

**GPQLNR**
Inquire List of Line Rendering Styles.

**GPQLSF**
Inquire Light Source Facilities.

**GPQLTF**
Inquire Linetype Facilities.

**GPQLW**
Inquire Length of Workstation State Tables.

**GPQMTF**
Inquire Marker Type Facilities.

**GPQNCN**
Inquire Number of Available Class Names.

**GPQNSP**
Inquire Number of Structure Priorities Supported.

**GPQNST**
Inquire Number of Secondary Triggers.

**GPQNV**
Inquire Number of Definable View Table Entries.

**GPQPAF**
Inquire Pattern Facilities.

**GPQPCR**
Inquire Predefined Color Representation.

**GPQPCS**
Inquire Primary Character Set.

**GPQPDC**
Inquire Physical Device Characteristics.

**GPQPER**
Inquire Predefined Edge Representation.

**GPQPIR**
Inquire Predefined Interior Representation.

**GPQPKT**
Inquire Pick Measure Type.

**GPQPLF**
Inquire Polyline Facilities.

**GPQPLR**
Inquire Predefined Polyline Representation.

**GPQPMF**
Inquire Polymarker Facilities.

**GPQPMR**
Inquire Predefined Polymarker Representation.

**GPQPPR**
Inquire Predefined Pattern Representation.

**GPQPTR**
Inquire Predefined Text Representation.

**GPQRCM**
Inquire Available Rendering Color Models.

**GPQSDF**
Inquire Surface Display Facilities.

**GPQSPD**
Inquire Source Physical Device.

**GPQTDF**
Inquire Trimming Curve Display Facilities.

**GPQTMO**
Inquire Available Transparency Modes.

**GPQVF**
Inquire View Facilities.

**GPQWC**
Inquire Workstation Category.

**GPQWD**
Inquire Workstation Display Classification.

**GPQWDT**
Inquire Workstation Description.

**GPQWTO**
Inquire Workstation Type and Options.

**GPQXCF**
Inquire Extended Color Facilities.

**GPQXTX**
Inquire Extended Text Facilities.

## PDT Inquiries

The subroutines in this section inquire information from the graPHIGS API Shell Description Table (PDT).

The PDT contains information describing the general capabilities of the graPHIGS API.

**GPQAI**
Inquire List of Available Application Image Formats.

**GPQCMM**
Inquire List of Available Connection Methods.

## PSL Inquiries

The subroutines in this section inquire information about the state of the graPHIGS API shell from the graPHIGS API State List (PSL).

The PSL contains the dynamic (variable) workstation-independent state of the graPHIGS API.

**GPQASV**
Inquire Archive State Value.

**GPQATR**
Inquire List of Attached Resources.

**GPQCEV**
Inquire Current Event.

**GPQCNC**
Inquire List of Connected Nuclei.

**GPQCS**
Inquire Character Set Identifier.

**GPQDCM**
Inquire Direct Color Model.

**GPQEDM**
Inquire Edit Mode.

**GPQEMO**
Inquire Error Handling Mode.

**GPQEMS**
Inquire Error Message.

**GPQFAR**
Inquire Font Aspect Ratios.

**GPQFCH**
Inquire Font Characteristics.

**GPQIBC**
Inquire Image Board Characteristics.

**GPQIQO**
Inquire Input Queue Overflow.

**GPQNCC**
Inquire Nucleus Connection State.

**GPQNCR**
Inquire Nucleus Resource Identifier.

**GPQOPW**
Inquire Set of Open Workstations.

**GPQSEV**
Inquire More Simultaneous Events.

**GPQSH**
Inquire Shell Identifier.

**GPQSHD**
Inquire Shell Deferral State.

**GPQSPL**
Inquire Shell Product Level.

**GPQSSS**
Inquire Selected Structure Store.

**GPQSTV**
Inquire Structure State Value.

**GPQSYV**
Inquire System State Value.

**GPQWSV**
Inquire Workstation State Value.

## NDT Inquiries

The subroutines in this section inquire information from the Nucleus Description Table (NDT).

The NDT contains information describing the general capabilities of the nucleus connected to the graPHIGS API shell.

**GPQIBF**
Inquire Image Board Facilities.

**GPQNCE**
Inquire Nucleus Environment.

**GPQNS**
Inquire Nucleus Specification.

**GPQPO**
Inquire Available Pixel Operations.

**GPQWTN**
Inquire List of Available Workstation Types on Nucleus.

## NSL Inquiries

The subroutines in this section inquire information from the Nucleus State List (NSL).

The NSL contains the dynamic (variable) states of the nucleus connected to the graPHIGS API shell.

**GPQNCS**
Inquire Available Nucleus Storage Size.

## SSL Inquiries

The subroutines in this section inquire information from the Structure State List (SSL) about the state of the structure store that exists in the nucleus connected to the graPHIGS API shell.

The following information can be inquired:

- current structure element pointers
- the currently open structure
- if a structure is in the SSL
- all the structures in the SSL
- structures executed by a structure
- the structure elements in a structure
- contents of a structure element
- workstations associated with a structure
- list of structures that conflict between different structure stores
- ancestor/descendant path data

**GPELS**
Element Search.

**GPQACS**
Inquire All Conflicting Structures in Structure Store.

**GPQCSN**
Inquire Conflicting Structures in Network in Structure Store.

**GPQED**
Inquire List of Element Data.

**GPQEDA**
Inquire List of Element Data for any Structure.

**GPQEHA**
Inquire List of Element Headers for any Structure.

**GPQEHD**
Inquire List of Element Headers.

**GPQEP**
Inquire Element Pointer.

**GPQEXS**
Inquire Executed Structures.

**GPQISN**
Inquire Identifiers of Structures in Network.

**GPQOPS**
Inquire Open Structure.

**GPQPAS**
Inquire Ancestors of Structure.

**GPQPDS**
Inquire Descendents of Structure.

**GPQRST**
Inquire Referencing Structures.

**GPQSTI**
Inquire Structure Identifiers.

**GPQSTS**
Inquire Structure Status.

**GPQWSA**
Inquire Set of Workstations to Which Associated.

## Archive Inquiries

The subroutines in this section inquire archive information from the structure store and the archive files.

With these subroutines, you can retrive the following information:

- list of structures in the archive file
- list of structures that conflict between the specified archive file and the currently selected structure store
- list of open archive files
- the conflict resolution flags
- ancestor/descendant path data

**GPQACA**
Inquire All Conflicting Structures in Archive.

**GPQARF**
Inquire Archive Files.

**GPQCNA**
Inquire Conflicting Structures in Network in Archive.

**GPQCNR**
Inquire Conflict Resolution.

**GPRAS**
Retrieve Ancestors to Structures.

**GPRDS**
Retrieve Descendants to Structures.

**GPRISN**
Retrieve Identifiers of Structures in Network.

**GPRSTI**
Retrieve Structure Identifiers.

## GPELS - Element Search

**GPELS (*strid*, *start*, *direction*, *inclnum*, *lincl*, *exclnum*, *lexcl*, *errind*, *status*, *position*, *header*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPELS** to search through a specified structure for an element that matches a given criteria. The search starts at a specified element position and searches in a designated direction until either an element is found that matches the criteria or until the limits of the structure are reached.

**Search criteria**
An element is selected if the element code is in the element inclusion list and not in the element exclusion list.

**Element exclusion**
An element is excluded if the element code is either not in the inclusion list or it is in the exclusion list.

**Starting search position**
The search starts at element position of zero if the specified starting position is less than zero. The search starts with the last element in the structure, if the specified starting position is larger than the number of elements in the structure.

If the search is successful, the application sets the status indicator to `2=SUCCESS`, the element position is returned in the *position* parameter, and the element type and size is returned in the *header* parameter. Otherwise, the application sets the status indicator to `1=FAILURE` and the values returned in the *position* and *header* parameters are unpredictable.

If the information is available, the error indicator is set to zero and the values are put into the output parameters. If the information is unavailable, the error indicator contains an error number indicating the reason. In this case, the values returned in the output parameters are unpredictable.

For the valid structure element codes used by the graPHIGS API see *The graPHIGS Programming Interface: Technical Reference*. You can use the following two additional element codes in the inclusion and exclusion lists for this subroutine:

| Decimal | Hex | Description |
|---|---|---|
| 0 | 0000 | Represents element position zero. |
| 65535 | FFFF | Represents all elements. |

### Parameters

*strid* **— specified by user, fullword integer**
Structure identifier.

*start* **— specified by user, fullword integer**
The position of the element to begin the search.

*direction* **— specified by user, fullword integer**
Search direction(1=BACKWARD, 2=FORWARD)

*inclnum* **— specified by user, fullword integer**
The number of element codes in the inclusion list (>=0)

*lincl* **— specified by user, array of fullword integers**
A list of element codes to be included in the inclusion list.

*exclnum* **— specified by user, fullword integer**
The number of element codes in the exclusion list (>=0)

*lexcl* **— specified by user, array of fullword integers**
A list of element codes to be included in the exclusion list.

*errind* **— specified by user, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL.

**122** STRUCTURE IDENTIFIER DOES NOT EXIST.

**528** DIRECTION VALUE IS INVALID.

**529** NUMBER OF ENTRIES IN INCLUSION OR EXCLUSION LIST < ZERO.

*status* **— returned by the graPHIGS API, fullword integer**
Status indicator which indicates the results of the search (1=FAILURE, 2=SUCCESS)

*position* **— returned by the graPHIGS API, fullword integer**
Position of the element found in the search.

*header* **— returned by the graPHIGS API, fullword integer**
Element header of the element found in the search. The first halfword contains the length of the element and the second halfword contains the element code. The information that corresponds to each element header and the list of valid structure element codes used by graPHIGS API are found in *The graPHIGS Programming Interface: Technical Reference*.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

# GPQAAF - Inquire Advanced Attribute Facilities

**GPQAAF (*wstype*, *attrib*, *start*, *number*, *errind*, *totnum*, *enum*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQAAF** to inquire the list of enumerations of an attribute that are supported by a specific workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*attrid* **— specified by user, fullword integer**
Attribute identifier being inquired (`1=EDGE_FLAG, 2=FACE_DISTINGUISH_MODE, 3=LIGHTING_CALCULATION_MODE, 4=POLYGON_CULLING, 5=POLYHEDRON_EDGE_CULLING, 6=POLYLINE_END_TYPE, 8=INTERIOR_SHADING_METHODS, 10=REFLECTANCE_MODES`).

*start* **— specified by user, fullword integer**
Starting member of the list of supported enumerations for the specified attribute (>=1).

*number* **— specified by user, fullword integer**
Number of supported enumerations requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**314** AN ATTRIBUTE IDENTIFIER IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of supported enumerations for the specified attribute.

*enum* **— returned by the graPHIGS API, array of fullword integers**
List of supported enumerations for the specified attribute.

- 1 - Edge Flag (`1=OFF, 2=ON, 3=GEOMETRY_ONLY`)

- 2 - Face Distinguish Mode (1=NONE, 2=COLOR_SURFACE_PROPERTIES)
- 3 - Lighting Calculation Mode (1=NONE, 2=PER_AREA, 3=PER_VERTEX)
- 4 - Polygon Culling (1=NONE, 2=BACK, 3=FRONT)
- 5 - Polyhedron Edge Culling (1=NONE, 2=BOTH_BACK, 3=BOTH_FRONT, 4=BOTH_BACK_BOTH_FRONT, 5=BACK_AND_FRONT, 6=LEAST_ONE_BACK, 7=LEAST_ONE_FRONT)
- 6 - Polyline End Type (1=FLAT, 2=ROUND, 3=SQUARE)
- 8 - Interior Shading Methods (1=SHADING_NONE, 2=SHADING_COLOR, 3=SHADING_DATA)
- 10 - Reflectance Mode (1=REFLECTANCE_NONE, 2=AMB, 3=AMB_DIFF, 4=AMB_DIFF_SPEC)

**Error Codes**

None

**Related Subroutines**

**GPPLET**
Set Polyline End Type

**GPEF** Set Edge Flag

**GPFDMO**
Set Face Distinguish Mode

**GPLMO**
Set Lighting Calculation Mode

**GPPGC**
Set Polygon Culling

**GPPHEC**
Set Polyhedron Edge Culling

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339411 (X'0C003213')

## GPQACA - Inquire All Conflicting Structures in Archive

**GPQACA (*arid*, *start*, *number*, *errind*, *totnum*, *idstrid*)**

**Note:** This subroutine is an Archive inquiry. For an overview, see "Archive Inquiries."

**Purpose**

Use **GPQACA** to inquire a list of structure identifiers that exist in both the currently selected structure store and the specified open archive file.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*arid* **— specified by user, fullword integer**
Archive file identifier.

*start* **— specified by user, fullword integer**
The starting member in the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**7** FUNCTION REQUIRES STATE AROP

**12** FUNCTION REQUIRES STATE SSSL

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of conflicting structures.

*idstrid* **— returned by the graPHIGS API, array of fullword integers**
List of conflicting structure identifiers.

**Error Codes**

None

**Related Subroutines**

**GPQCNA**
Inquire Conflicting Structures in Network in Archive

**RCP code**

201347592(X'0C005208')

## GPQACS - Inquire All Conflicting Structures in Structure Store

**GPQACS (*ssid*, *start*, *number*, *errind*, *totnum*, *istrid*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

**Purpose**

Use **GPQACS** to inquire a list of the structure identifiers that exist in both the currently selected structure store.

If the specified structure store identifier is the same as the currently selected structure store identifier, then this subroutine is functionally equivalent to the Inquire Structure Identifiers (**GPQSTI**) subroutine.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*ssid* **— specified by user, fullword integer**
Structure store identifier.

*start* **— specified by user, fullword integer**
The starting member of the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**222** SPECIFIED STRUCTURE STORE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of conflicting structures.

*istrid* **— returned by the graPHIGS API, array of fullword integers**
List of conflicting structure identifiers.

**Error Codes**

None

**Related Subroutines**

**GPQCSN**
Inquire Conflicting Structures in Network in Structure Store

**GPQSTI**
Inquire Structure Identifiers

**RCP code**

201347590 (X'0C005206')

## GPQAI - Inquire List of Available Application Image Formats

**GPQAI (*start*, *number*, *errind*, *totnum*, *format*)**

**Note:** This subroutine is a graPHIGS API Description Table (PDT) inquiry. For an overview, see "PDT Inquiries."

### Purpose

Use **GPQAI** to inquire the list of available application image formats.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*start* **— specified by user, fullword integer**
Starting member in the list of available application image formats (>=1).

*number* **— specified by user, fullword integer**
Number of image formats requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available application image formats.

*format* **— returned by the graPHIGS API, array of fullword integers**
List of application image formats (`1=PIXEL_ARRAY`).

### Error Codes

None

### Related Subroutines

None

### RCP code

201345803 (X'0C004B0B')

## GPQAMO - Inquire Available Antialiasing Modes

**GPQAMO (*wstype*, *start*, *number*, *errind*, *totnum*, *mode*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQAMO** to inquire the antialiasing facilities that are supported by the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available antialiasing modes (>=1).

*number* **— specified by user, fullword integer**
Number of antialiasing modes requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available antialiasing modes supported by a workstation.

*mode* **- returned by the graPHIGS API, array of fullword integers**
List of available antialiasing modes (`1=OFF, 2=SUBPIXEL_ON_THE_FLY, 3=NON_SUBPIXEL_ON_THE_FLY`).

**Error Codes**

None

**Related Subroutines**

**GPAID**
Set Antialiasing Identifier

**GPXVR**
Set Extended View Representation

**RCP code**

201339414 (X'0C003216')

## GPQANF - Inquire Annotation Facilities

**GPQANF (*wstype*, *start*, *number*, *errind*, *totnum*, *styles*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQANF** to inquire the list of annotation styles that are supported by the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting number of annotation styles (>=1).

*number* **— specified by user, fullword integer**
Number of annotation styles requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available annotation styles.

*styles* **— returned by the graPHIGS API, array of fullword integers**
List of annotation styles (`1=UNCONNECTED, 2=LEAD_LINE`).

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201346060 (X'0C004C0C')

# GPQAR - Inquire Set of Associated Roots

**GPQAR (*wsid*, *start*, *number*, *errind*, *totnum*, *strid*)**

> **Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQAR** to inquire a list of structure identifiers which belong to the currently selected structure store and are associated with the workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of structure identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **215** | SPECIFIED RESOURCES DO NOT EXIST ON THE SAME NUCLEUS |
| **227** | STRUCTURE STORE IS NOT SELECTED |
| **538** | START VALUE < ONE |
| **539** | REQUESTED NUMBER < ZERO |
| **543** | START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED |

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of roots associated to the workstation.

*strid* **— returned by the graPHIGS API, array of fullword integers**
List of root structure identifiers.

**Error Codes**

None

**Related Subroutines**

**GPARV**
Associate Root with View

**GPARW**
Associate Root with Workstation

**RCP code**

201337089 (X'0C002901')

## GPQARF - Inquire Archive Files

**GPQARF (*start*, *number*, *buflen*, *errind*, *actnum*, *totnum*, *arlist*, *termcond*)**

**Note:** This subroutine is an Archive inquiry. For an overview, see "Archive Inquiries."

**Purpose**

Use **GPQARF** to inquire a list of open archive files.

If the current archive state is archive closed (ARCL), then the graPHIGS API returns a value of zero for the total number of open archive files (*totnum*).

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*start* **— specified by user, fullword integer**
The starting member in the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*buflen* **— specified by user, fullword integer**
Length, in bytes, of the data area specified by the *arlist* parameter into which the graPHIGS API returns the archive file names (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

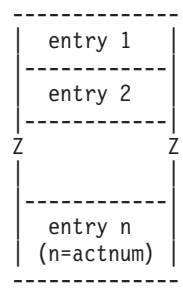
**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**577** BUFFER LENGTH IS < ZERO

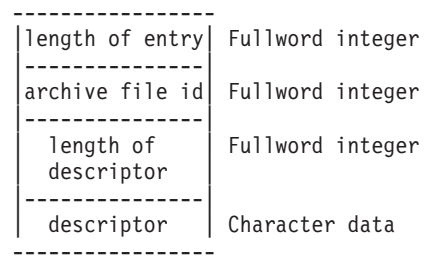
*actnum* **— returned by the graPHIGS API, fullword integer**
Total number of archive file names returned.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of open archive files.

*arlist* **— returned by the graPHIGS API, variable data**
List of entries for the requested archive files. The value of each field is expressed in the following data format:




where each entry has the following format:




The graPHIGS API pads each archive file descriptor with blanks to align the next length field on a fullword boundary.

*termcond* **— returned by the graPHIGS API, fullword integer**
Termination condition. The graPHIGS API terminated the list of archive file names due to one of the following reasons:

**1- Count Exhausted**
The graPHIGS API returned the requested number of archive file names.

**2- Buffer Overflow**
The graPHIGS API count not return the requested number of elements because there was not enough room in the area provided. *actnum* contains the actual number returned.

**3- End of List of Archive Files**
The graPHIGS API reached the last archive file in the list. This condition supersedes the Count Exhausted condition (if that condition was in effect). The total number of archive files returned may not be equal to the number of returned archive files you requested. Check *actnum* to find the actual number of archive files returned.

### Error Codes

None

### Related Subroutines

None

### RCP code

201336338 (X'0C002612')

## GPQART - Inquire Rendering Targets

**GPQART (*wstype*, *errind*, *totnum*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQART** to inquire the number of rendering targets that are available on the specified workstation for explicit control by an application. A rendering target is a repository for the 'picture' produced by rendering primitives and their attributes. Rendering targets may be frame buffers or disk files. Implicit in the workstation type is the type of rendering target that is available on the workstation. A workstation does not support multiple types of rendering targets.

Each available rendering target on a workstation has an associated identifier. The identifiers are from one to *totnum* (which is the total number of rendering targets available on the workstation). When you open a workstation, the current displayed rendering target and the current selected rendering target are the same.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— specified by user, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

| | |
|---|---|
| **23** | SPECIFIED WORKSTATION TYPE DOES NOT EXIST |
| **548** | SPECIFIED WORKSTATION TYPE CANNOT BE LOADED |

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of rendering targets available for explicit control by the application. (Zero indicates that there are no rendering targets are available for explicit control by the application.)

### Error Codes

None

### Related Subroutines

None

### RCP code

201339415 (X'0C003217')

## GPQASV - Inquire Archive State Value

**GPQASV (*state*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQASV** to inquire the archive state of the graPHIGS API. The archive state is either Archive Open (2=AROP) or Archive Closed (1=ARCL). If the state is AROP, then at least one archive file is open. If the state is ARCL, then no archive files are open.

### Parameters

*state* **— specified by user, fullword integer**
Archive state value (1=ARCL, 2=AROP).

### Error Codes

None

### Related Subroutines

**GPCLAR**
Close Archive File

**GPOPAR**
Open Archive File

### RCP code

201336336 (X'0C002610')

## GPQATR - Inquire List of Attached Resources

**GPQATR (*ncid*, *type*, *start*, *number*, *errind*, *totnum*, *id*)**

> **Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQATR** to inquire the list of resources of the specified type in the specified nucleus that are attached to the shell.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

> **Note:** The Inquire Set of Open Workstations (**GPQOPW**) subroutine is treated as a special form of this generic subroutine. It inquires the set of workstation resources in the nucleus with an identifier of 1.

### Parameters

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*type* **— specified by user, fullword integer**
Resource type (1=WORKSTATION, 2=STRUCTURE_STORE, 3=IMAGE_BOARD, 4=FONT_DIRECTORY, 5=ARCHIVE_FILE).

*start* **— specified by user, fullword integer**
Starting member in the list of attached resources (>=1).

*number* **— specified by user, fullword integer**
Number of shell resource identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**211** RESOURCE TYPE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of attached resources of the specified resource type.

*id* **— returned by the graPHIGS API, array of fullword integers**
List of identifiers belonging to the specified resource type.

**Error Codes**

None

**Related Subroutines**

**GPATR**
Attach Resource

**GPDTR**
Detach Resource

**GPQOPW**
Inquire Set of Open Workstations

**RCP code**

201336334 (X'0C00260E')

## GPQBK - Inquire Break Capabilities

**GPQBK (*wstype*, *start*, *number*, *errind*, *ntrigs*, *ltrigs*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQBK** to inquire the break action capabilities for the specified workstation type. If the break action is programmable, the available triggers are returned.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member in the list of available triggers (>=1).

*number* **— specified by user, fullword integer**
Number of trigger list entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

**572** WORKSTATION DOES NOT SUPPORT PROGRAMMABLE BREAK ACTION

*ntrigs* **— returned by the graPHIGS API, fullword integer**
Number of entries in the available triggers list.

*ltrigs* **— returned by the graPHIGS API, array of fullword integers**
List of available triggers (positive integers are button device numbers). The list is an array of trigger descriptors in which a descriptor consists of a triplet (3 fullword integers) containing the trigger type, low trigger qualifier, and high trigger qualifier. Positive integers as trigger types are button device numbers. The trigger qualifier for a choice device is the choice number. The parameter *ntrigs* identifies the total number of triplets in the available trigger list. The actual number returned depends on the setting of the *start* and *number* parameters.

**Error Codes**

None

**Related Subroutines**

**GPBKAC**
Set Break Action

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339664 (X'0C003310')

# GPQBKS - Inquire Break Action State

**GPQBKS (*wsid*, *errind*, *trigger*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQBKS** to inquire the current break action trigger on the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

*trigger* **— returned by the graPHIGS API, 2 fullword integers**
Trigger to be used on the specified workstation for the break action. The trigger consists of a trigger type followed by a trigger qualifier. Positive integers as trigger types are button device numbers. The trigger qualifier for a button device is the choice alternative.

### Error Codes

None

### Related Subroutines

**GPBKAC**
Set Break Action

**GPQBK**
Inquire Break Capabilities

**GPQDBK**
Inquire Default Break Action

### RCP code

201336843 (X'0C00280B')

# GPQCCH - Inquire Color Table Characteristics

**GPQCCH (*wsid*, *ctid*, *errind*, *model*, *length*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQCCH** to inquire the current characteristics of the specified color table on the workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ctid* **— specified by user, fullword integer**
Color table identifier (`-1=DISPLAY_COLOR_TABLE_MODIFIABLE, 0=RENDERING_COLOR_TABLE_MODIFIABLE`).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**284** COLOR TABLE IDENTIFIER DOES NOT EXIST

*model* **— returned by the graPHIGS API, fullword integer**
Color model (`1=RGB, 2=HSV, 3=CMY, 4=CIELUV`).

*length* **— returned by the graPHIGS API, fullword integer**
Length of the color table in $\log_2$.

### Error Codes

None

### Related Subroutines

**GPQCID**
Inquire List of Color Table Identifiers

**GPXCR**
Set Extended Color Representation

### RCP code

201339144 (X'0C003108')

## GPQCDF - Inquire Curve Display Facilities

**GPQCDF (*wstype*, *start*, *number*, *errind*, *order*, *totnum*, *criteria*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQCDF** to inquire the curve facilities of the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available curve approximation criteria (>=1).

*number* **— specified by user, fullword integer**
Number of curve approximation criteria requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*order* **— returned by the graPHIGS API, fullword integer**
Maximum curve order supported.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available curve approximation criteria.

*criteria* **— returned by the graPHIGS API, array of fullword integers**
List of curve approximation criteria (`1=WORKSTATION_DEPENDENT`, `3=CONSTANT_SUBDIVISION_BETWEEN_KNOTS`, `8=VARIABLE_SUBDIVISION_BETWEEN_KNOTS`).

### Error Codes

None

### Related Subroutines

**GPCAC**
Set Curve Approximation Criteria

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201346057 (X'0C004C09')

## GPQCEV - Inquire Current Event

**GPQCEV (*major*, *class*, *minor*)**

> **Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQCEV** to inquire information about the current event report.

When the current event report is empty, the graPHIGS API returns a zero for the event class parameter and major/minor code parameters are not set. Otherwise, the graPHIGS API returns an event class of the event in the current event report to the event class parameter.

The details of the possible event classes and meanings of their major and minor codes are shown in *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*major* **— returned by the graPHIGS API, fullword integer**
Major event code.

*class* **— returned by the graPHIGS API, fullword integer**
Event class.

*minor* **— returned by the graPHIGS API, fullword integer**
Minor event code.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201336329 (X'0C002609')

## GPQCF - Inquire Color Facilities

**GPQCF (*wstype*, *errind*, *model*, *ncolor*, *avcolor*, *npred*)**

> **Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQCF** to inquire the color facilities of the specified workstation type.

The graPHIGS API returns the default color model of the workstation (*model*[default], the total quantity of available colors (*ncolor*[default], the color status (*avcolor*) where 1=MONOCHROME, 2=COLOR, and the quantity of predefined color table entries in the workstation's default color table (*npred*).

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*model* **— returned by the graPHIGS API, fullword integer**
Color model of the workstation (1=RGB, 2=HSV, 3=CMY, 4=CIELUV).

*ncolor* **— returned by the graPHIGS API, fullword integer**
Number of available colors (the total color palette size).

*avcolor* **— returned by the graPHIGS API, fullword integer**
Color available (1=MONOCHROME, 2=COLOR).

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined default color table entries.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**GPXCR**
Set Extended Color Representation

**RCP code**

201339659 (X'0C00330B')

## GPQCH - Inquire Choice Device State

**GPQCH (*wsid*, *device*, *type*, *length*, *errind*, *mode*, *echosw*, *choice*, *echo*, *area*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQCH** to inquire the current state of the specified choice device on the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Choice device number (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*length* **— specified by user, fullword integer**
Length of choice data record array requested in bytes.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**534** TYPE VALUE IS INVALID

*mode* **— returned by the graPHIGS API, fullword integer**
Current operating mode (1=REQUEST, 2=SAMPLE, 3=EVENT, 4=APPLICATION_DEFINED). The graPHIGS API only returns a mode of 4=APPLICATION_DEFINED if the application set the device mode using the Set Input Device State (**GPIDMO**) subroutine and the mode does not emulate Request, Sample or Event mode.

*echosw* **— returned by the graPHIGS API, fullword integer**
Current echo switch (1=NOECHO, 2=ECHO).

*choice* **— returned by the graPHIGS API, fullword integer**
Current initial choice number.

*echo* **— returned by the graPHIGS API, fullword integer**
Current prompt/echo type.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Current echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*datalen* **— returned by the graPHIGS API, fullword integer**
Current choice data record length in bytes.

*data* **— returned by the graPHIGS API, variable length data**
Current choice data record.

### Error Codes

None

### Related Subroutines

**GPIDMO**
Set Input Device Mode

**GPINCH**
Initialize Choice

### RCP code

201338881 (X'0C003001')

## GPQCID - Inquire List of Color Table Identifiers

**GPQCID (*wsid*, *start*, *number*, *errind*, *totnum*, *ctid*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQCID** to inquire the list of color table identifiers existing on the specified workstation.

Color table identifiers that may exist on a workstation include: `-1=DISPLAY_COLOR_TABLE`, `0=RENDERING_COLOR_TABLE`.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of existing color tables (>=1).

*number* **— specified by user, fullword integer**
Number of color table identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* — **returned by the graPHIGS API, fullword integer**
Total number of color tables on the workstation.

*ctid* — **returned by the graPHIGS API, array of fullword integers**
List of color table identifiers.

**Error Codes**

None

**Related Subroutines**

**GPDFI**
Define Image

**GPXCR**
Set Extended Color Representation

**RCP code**

201339145 (X'0C003109')

## GPQCML - Inquire Color Model

**GPQCML (*wsid*, *errind*, *model*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQCML** to inquire the current color model for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*model* — **returned by the graPHIGS API, fullword integer**
Current color model (`1=RGB, 2=HSV, 3=CMY, 4=CIELUV`).

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201336833 (X'0C002801')

## GPQCMM - Inquire List of Available Connection Methods

**GPQCMM (*start*, *number*, *errind*, *totnum*, *conn*)**

**Note:** This subroutine is a graPHIGS API Description Table (PDT) inquiry. For an overview, see "PDT Inquiries."

**Purpose**

Use **GPQCMM** to determine the list of connection methods available to the application process when connecting to a nucleus.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*start* **— specified by user, fullword integer**
Starting member in the list of available connection methods (>=1).

*number* **— specified by user, fullword integer**
Number of connection methods requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available connection methods.

*conn* **— returned by the graPHIGS API, array of fullword integers**
List of available connection methods (`1=CALL, 2=GAM, 3=SOCKETS`) GAM=Graphics Access Method and is the connection method used with the IBM 6090 Graphics System Access.

**Error Codes**

None

**Related Subroutines**

**GPCNC**
Connect Nucleus

**RCP code**

201345793 (X'0C004B01')

## GPQCNA - Inquire Conflicting Structures in Network in Archive

**GPQCNA (*arid*, *strid*, *source*, *start*, *number*, *errind*, *totnum*, *istrid*)**

**Note:** This subroutine is an Archive inquiry. For an overview, see "Archive Inquiries."

**Purpose**

Use **GPQCNA** to inquire a list of structure identifiers from a specified structure network that exists in both the currently selected structure store and the specified open archive file.

The value for the source determines whether the structure network originates from the currently selected structure store or from the archive file.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*arid* **— specified by user, fullword integer**
Archive file identifier.

*strid* **— specified by user, fullword integer**
Structure identifier of the root structure.

*source* **— specified by user, fullword integer**
The source of the structure network to be searched:

| Value | Meaning |
|---|---|
| **1** | Currently selected structure store |
| **2** | Archive file |

*start* **— specified by user, fullword integer**
The starting member of the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**7** FUNCTION REQUIRES STATE AROP

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**135** VALUE OF SOURCE IS INVALID

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of conflicting structures in the network.

*istrid* **— returned by the graPHIGS API, array of fullword integers**
List of conflicting structure identifiers.

**Error Codes**

None

**Related Subroutines**

**GPQACA**
Inquire All Conflicting Structures in Archive

**RCP code**

201347593 (X'0C005209')

## GPQCNC - Inquire List of Connected Nuclei

**GPQCNC (*start*, *number*, *errind*, *totnum*, *ncid*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQCNC** to inquire the nuclei that are connected to the application process (shell).

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*start* **— specified by user, fullword integer**
Starting member in the list of connected nuclei (>=1).

*number* **— specified by user, fullword integer**
Number of nucleus identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of connected nuclei.

*ncid* **— returned by the graPHIGS API, array of fullword integers**
List of nucleus identifiers connected to the application process (shell).

**Error Codes**

None

**Related Subroutines**

**GPCNC**
Connect Nucleus

**GPDNC**
Disconnect Nucleus

**RCP code**

201345794 (X'0C004B02')

# GPQCNR - Inquire Conflict Resolution

**GPQCNR (*aflag*, *rflag*)**

**Note:** This subroutine is an Archive inquiry. For an overview, see "Archive Inquiries."

**Purpose**

Use **GPQCNR** to inquire the archive file conflict resolution values of the graPHIGS API.

There are two conflict resolution flags that return values: *aflag* is used when the graPHIGS API moves structure data *to* an archive file from a structure store and *rflag* is used when the graPHIGS API moves archive data *from* an archive file to a structure store. The possible return values are: `1=MAINTAIN`, `2=ABANDON`, and `3=UPDATE`.

- If the value returned is `1=MAINTAIN`, then the graPHIGS API only transfers the structures that do not conflict.
- If the value returned is `2=ABANDON`, then the graPHIGS API does not transfer any structures when a conflict occurs.
- If the value returned is `3=UPDATE`, then the graPHIGS API transfers all the structures and replaces any conflicting structures with the new ones.

**Parameters**

*aflag* **— returned by the graPHIGS API, fullword integer**
Archive conflict resolution(`1=MAINTAIN, 2=ABANDON, 3=UPDATE`).

*rflag* **— returned by the graPHIGS API, fullword integer**
Retrieve conflict resolution (`1=MAINTAIN, 2=ABANDON, 3=UPDATE`).

**Error Codes**

None

**Related Subroutines**

**GPCNRS**
Set Conflict Resolution

**RCP code**

201336337 (X'0C002611')

## GPQCPF - Inquire Color Processing Facilities

**GPQCPF (*wstype*, *errind*, *number*, *npred*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQCPF** to inquire the color processing mode table facilities supported by the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*number* **— returned by the graPHIGS API, fullword integer**
Number of definable color processing mode table entries. Entry 0 of the color processing table may not be changed.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined color processing mode table entries.

**Error Codes**

None

**Related Subroutines**

**GPCPI**
Set Color Processing Index

**GPCPR**
Set Color Processing Representation

**GPQCPR**
Inquire Color Processing Representation

**GPQCQM**
Inquire Available Color Quantization Methods

**GPQRCT**
Inquire Realized Connection Type

**GPXVR**
Set Extended View Representation

**RCP code**

201346054 (X'0C004C06')

## GPQCPR - Inquire Color Processing Representation

**GPQCPR (*wsid*, *index*, *errind*, *model*, *quant*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQCPR** to inquire the current attribute values in the specified entry in the color processing table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Color processing table index (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**265** COLOR PROCESSING INDEX < ZERO

**266** COLOR PROCESSING INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY

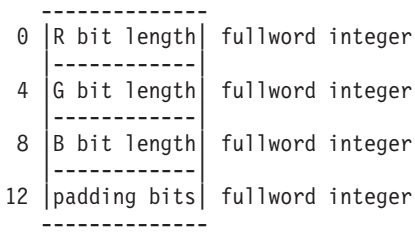
*model* **— returned by the graPHIGS API, fullword integer**
Rendering color model (`1=RGB_NORMAL, 2=RGB_B_ONLY`).

*quant* **— returned by the graPHIGS API, fullword integer**
Quantization method (1=WORKSTATION_DEPENDENT, 2=BITWISE).

*data* **— returned by the graPHIGS API, variable data**
Quantization parameters. Values returned to this parameter depend on the quantization method. The application must supply storage for this parameter that is large enough to contain the maximum data listed below.

**If *quant*=1 (WORKSTATION_DEPENDENT)**
The *data* parameter should be ignored.

**If *quant*=2 (BITWISE)**
The *data* parameter returns the following format:




**Note:** The least significant (right most) bits of the padding data will be used.

### Error Codes

None

### Related Subroutines

**GPCPI**
Set Color Processing Index

**GPCPR**
Set Color Processing Representation

**GPQCPF**
Inquire Color Processing Facilities

**GPQCQM**
Inquire Available Color Quantization Methods

**GPQRCM**
Inquire Available Rendering Color Models

**GPXVR**
Set Extended View Representation

### RCP code

201339146 (X'0C00310A')

## GPQCQM - Inquire Available Color Quantization Methods

**GPQCQM (*wstype*, *start*, *number*, *errind*, *totnum*, *method*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQCQM** to inquire a list of available color quantization methods supported by the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available color quantization methods (>=1).

*number* **— specified by user, fullword integer**
Number of color quantization methods requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available color quantization methods.

*method* **— returned by the graPHIGS API, array of fullword integers**
List of color quantization methods (`1=WORKSTATION_DEPENDENT, 2=BITWISE`).

### Error Codes

None

### Related Subroutines

**GPCPI**
Set Color Processing Index

**GPCPR**
Set Color Processing Representation

**GPQCPF**
Inquire Color Processing Facilities

**GPQCPR**
Inquire Color Processing Representation

**GPQRCT**
Inquire Realized Connection Type

**GPXVR**
Set Extended View Representation

**RCP code**

201339405 (X'0C00320D')

## GPQCS - Inquire Character Set Identifier

**GPQCS (*csid*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQCS** to inquire the current character set identifier from the graPHIGS API state list.

**Parameters**

*csid* **— returned by the graPHIGS API, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

**Error Codes**

None

**Related Subroutines**

**GPICS**
Set Input Character Set

**RCP code**

201336333 (X'0C00260D')

## GPQCSF - Inquire Cull Size Facilities

**GPQCSF (*wstype*, *errind*, *number*, *npred*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQCSF** to inquire the cull size facilities supported by the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*number* **— returned by the graPHIGS API, fullword integer**
Number of definable cull size table entries. All entries in the cull size table may be changed.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined cull size table entries.

**Error Codes**

None

**Related Subroutines**

**GPCSR**
Set Cull Size Representation

**GPQCSR**
Inquire Cull Size Representation

**GPQRCT**
Inquire Realized Connection Type

**GPTEX2**
Test Extent 2

**GPTEX3**
Test Extent 3

**RCP code**

201346055 (X'0C004C07')

## GPQCSN - Inquire All Conflicting Structures in Network in Structure Store

**GPQCSN (*ssid*, *strid*, *source*, *start*, *number*, *errind*, *totnum*, *istrid*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

**Purpose**

Use **GPQCSN** to inquire a list of the structure identifiers from a specified structure network that exists in both the currently selected structure store and the specified structure store.

The value of the *source* parameter determines whether the structure network originates from the currently selected structure store (1) or from the specified structure store (2).

If the specified structure store identifier is the same as the currently selected structure store identifier, then this subroutine is functionally equivalent to the Inquire Identifiers of Structures in Network (**GPQISN**) subroutine.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*ssid* **— specified by user, fullword integer**
Structure store identifier.

*strid* **— specified by user, fullword integer**
Structure identifier of the root structure.

*source* **— specified by user, fullword integer**
The source of the structure network to be searched:

| Value | Meaning |
|---|---|
| **1** | Currently selected structure store |
| **2** | Specified structure store |

*start* **— specified by user, fullword integer**
The starting member of the list of structure identifiers (>=1)

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**135** VALUE OF SOURCE IS INVALID

**222** SPECIFIED STRUCTURE STORE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of conflicting structures in network.

*istrid* **— returned by the graPHIGS API, array of fullword integers**
List of conflicting structure identifiers.

### Error Codes

None

### Related Subroutines

**GPQACS**
Inquire All Conflicting Structures in Structure Store

**GPQISN**
Inquire Identifiers of Structures in Network

### RCP code

201347591 (X'0C005207')

## GPQCSR - Inquire Cull Size Representation

**GPQCSR (*wsid*, *index*, *type*, *errind*, *size*)**

> **Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQCSR** to inquire the current cull size in the specified entry in the cull size table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Cull size table index (>=1)

*type* **— specified by user, fullword integer**
Type of returned value (`1=SET`).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**278** CULL SIZE INDEX < ONE

**279** CULL SIZE INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY

**534** TYPE VALUE IS INVALID

*size* **— returned by the graPHIGS API, short floating-point number (DC)**
Cull size.

**Error Codes**

None

**Related Subroutines**

**GPCSR**
Set Cull Size Representation

**GPTEX2**
Test Extent 2

**GPTEX3**
Test Extent 3

**GPQCSF**
Inquire Cull Size Facilities

**RCP code**

201339147 (X'0C00310B')

## GPQCUF - Inquire Cursor Facilities

**GPQCUF (*wstype*, *start1*, *num1*, *start2*, *num2*, *errind*, *maxent*, *maxsize*, *totnum1*, *lformat*, *totnum2*, *lcursor*, *npred*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQCUF** to inquire the cursor definition facilities for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start1* **— specified by user, fullword integer**
Starting member of the list of available cursor definition formats (>=1).

*num1* **— specified by user, fullword integer**
Number of entries from the list of available cursor definition formats that are requested (>=0).

*start2* **— specified by user, fullword integer**
Starting member of the list of available fixed cursor types (>=1).

*num2* **— specified by user, fullword integer**
Number of entries from the list of available fixed cursor types that are requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**37** WORKSTATION IS NOT OF CATEGORY OUTIN

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*maxent* **— returned by the graPHIGS API, fullword integer**
Maximum number of cursor shape table entries.

*maxsize* **— returned by the graPHIGS API, fullword integer**
Maximum number of bytes that a cursor definition may occupy.

*totnum1* **— returned by the graPHIGS API, fullword integer**
Total number of available cursor definition formats.

*lformat* **— returned by the graPHIGS API, array of fullword integers**
List of available cursor definition formats. Each entry of this array consists of three integers. The first one will always be a format identifier. The meaning of the other two words is dependent on the value of the first word.

**Format 1 -**
Words 2 and 3 are the required *x* size and *y* size respectively of the pixel array which defines the cursor shape.

*totnum2* **— returned by the graPHIGS API, fullword integer**
Total number of available fixed cursor types.

*lcursor* **— returned by the graPHIGS API, array of fullword integers**
List of available fixed cursor types.

Possible available fixed cursor types are:

- -1=Cross hair cursor.

*npred* **— returned by the graPHIGS API, array of fullword integers**
Number of predefined cursor shape table entries.

**Error Codes**

None

**Related Subroutines**

**GPCUR**
Set Cursor Representation

**GPCUS**
Set Cursor Shape

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201346061 (X'0C004C0D')

# GPQCVE - Inquire Current View Table Entries Input

**GPQCVE (*wsid*, *start*, *number*, *errind*, *nview*, *view*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQCVE** to inquire the current view table indexes in input priority order for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of view table entries (>=1).

*number* **— specified by user, fullword integer**
Number of view table entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*nview* **— returned by the graPHIGS API, fullword integer**
Total number of view table entries.

*view* **— returned by the graPHIGS API, array of fullword integers**
List of view table indexes, in decreasing view input priority order. The output array must be large enough to contain the requested data.

### Error Codes

None

### Related Subroutines

**GPVIP**
Set View Input Priority

### RCP code

# GPQCVO - Inquire Current View Table Entries Output

**GPQCVO (*wsid*, *start*, *number*, *errind*, *nview*, *view*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQCVO** to inquire the current view table indexes in output priority order for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of view table entries (>=1).

*number* **— specified by user, fullword integer**
Number of view table entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*nview* **— returned by the graPHIGS API, fullword integer**
Total number of view table entries.

*view* **— returned by the graPHIGS API, array of fullword integers**
List of view table indexes, in decreasing view output priority order. The output array must be large enough to contain the requested data.

### Error Codes

None

### Related Subroutines

**GPVOP**
Set View Output Priority

**RCP code**

201336848 (X'0C002810')

## GPQCVR - Inquire Current View Representation

**GPQCVR (*wsid*, *view*, *number*, *ids*, *errind*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQCVR** to inquire one or more fields from the specified current view table entry.

Each field in the view table entry is identified by a group identifier.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter in the requested order. The output parameter must be large enough to store all requested data.

If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*number* **— specified by user, fullword integer**
Number of groups requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers requested.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**571** INQUIRED INFORMATION IS NOT AVAILABLE

*data* — **returned by the graPHIGS API, variable data**
Data array containing the values in the requested groups. The value of each field is expressed in the data format listed below:

**Group Identifier 1 - Window clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP)

**Group Identifier 2 - Near clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP)

**Group Identifier 3 - Far clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP)

**Group Identifier 4 - Shielding indicator**
A fullword integer (1=OFF, 2=ON)

**Group Identifier 5 - Shielding color**
Four fullwords of data with either of the following two formats:

```
      indexed format                        direct format
      -------------                         -------------
    0 |           1| fullword integer      0 |           2| fullword integer
      |------------|                         |------------|
    4 |color index | fullword integer      4 |component 1 | short floating-point
number
      |------------|                         |------------|
    8 |     ignored| fullword integer      8 |component 2 | short floating-point
number
      |------------|                         |------------|
   12 |     ignored| fullword integer     12 |component 3 | short floating-point
number
      -------------                         -------------
```

**Group Identifier 6 - Border indicator**
A fullword integer (1=OFF, 2=ON)

**Group Identifier 7 - Border color**
Four fullwords of data with either of the following two formats:

```
      indexed format                        direct format
      -------------                         -------------
    0 |           1| fullword integer      0 |           2| fullword integer
      |------------|                         |------------|
    4 |color index | fullword integer      4 |component 1 | short floating-point
number
      |------------|                         |------------|
    8 |     ignored| fullword integer      8 |component 2 | short floating-point
number
      |------------|                         |------------|
   12 |     ignored| fullword integer     12 |component 3 | short floating-point
number
      -------------                         -------------
```

**Group Identifier 8 - Reserved.**
This field is reserved.

**Group Identifier 9 - Temporary view indicator**
A fullword integer (1=OFF, 2=ON)

**Group Identifier 10 - HLHSR mode**
A fullword integer (1=OFF, 2=ON_THE_FLY)

**Group Identifier 11 - Transparency processing mode**
A fullword integer (1=OFF, 2=PARTIAL_TRANSPARENT, 3=BLEND, 4=BLEND_ALL)

**Group Identifier 12 - Initial color processing mode index**
A fullword integer (>=0)

**Group Identifier 13 - Initial frame buffer write protect mask**
A 32-bit bit string.

**Group Identifier 14 - Viewport, 2D form**
4 short floating-point numbers (including only Xmin, Xmax, Ymin, Ymax). For the set subroutine, Zmin and Zmax are set to their default values.

**Group Identifier 15 - Viewport, 3D form**
6 short floating-point numbers (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax)

**Group Identifier 16 - View volume, 2D form**
4 short floating-point numbers specifying the view window (Umin, Umax, Vmin, Vmax). For the set subroutine, other fields of the view volume group are set to their default values.

**Group Identifier 17 - View volume, 3D form**
10 short floating-point numbers and a fullword integer specifying a view window (Umin, Umax, Vmin, Vmax), near plane distance, far plane distance, projection reference point (*u*, *v*, *n*), view plane distance and a projection type (1=PARALLEL, 2=PERSPECTIVE)

| | |
|---|---|
| Umin | short floating-point number |
| Umax | short floating-point number |
| Vmin | short floating-point number |
| Vmax | short floating-point number |
| near plane distance | short floating-point number |
| far plane distance | short floating-point number |
| U projection-reference point | short floating-point number |
| V projection-reference point | short floating-point number |
| N projection-reference point | short floating-point number |
| view plane distance | short floating-point number |
| projection type | fullword integer |

**Group Identifier 18 - View matrix, 2D form**
9 short floating-point numbers. For the output view matrix, the elements are in the following order:

```
|m11  m12  m14|
|m21  m22  m24| --->(m11,m12,m14,m21.....m44)
|m41  m42  m44|
```

The output 3x3 matrix is extracted by the graPHIGS API from the 4[default]4 matrix of the three-dimensional form:

```
|m11  m12  m13  m14|
|m21  m22  m23  m24| --->(m11,m12,m13,m14,m21,m22.....m44)
|m31  m32  m33  m34|
|m41  m42  m43  m44|
```

**Group Identifier 19 - View matrix, 3D form**
16 short floating-point numbers, M11, M12, M13. For the output view matrix, the elements are in the following order:

```
|m11  m12  m13  m14|
|m21  m22  m23  m24| --->(m11,m12,m13,m14,m21,m22.....m44)
|m31  m32  m33  m34|
|m41  m42  m43  m44|
```

**Group Identifier 20 - View input active flag**
A fullword integer (1=INACTIVE, 2=ACTIVE)

**Group Identifier 21 - View output active flag**
A fullword integer (1=INACTIVE, 2=ACTIVE)

**Group Identifier 22 - View mapping matrix, 2D form**
9 short floating-point numbers:

```
|m11  m12  m14|
|m21  m22  m24| --->(m11,m12,m14,m21.....m44)
|m41  m42  m44|
```

**Group Identifier 23 - View mapping matrix, 3D form**
16 short floating-point numbers:

```
|m11  m12  m13  m14|
|m21  m22  m23  m24| --->(m11,m12,m13,m14,m21,m22.....m44)
|m31  m32  m33  m34|
|m41  m42  m43  m44|
```

**Group Identifier 24 - Antialiasing mode**
A fullword integer (1=OFF, 2=ON)

**Group Identifier 25 - Shield alpha value**
A fullword integer (0<=*alpha*<=255)

### Error Codes

None

### Related Subroutines

**GPQAMO**
Inquire Available Antialiasing Modes

**GPQHMO**
Inquire Available HLHSR Modes

**GPQRVR**
Inquire Requested View Representation

**GPQTMO**
Inquire Available Transparency Modes

**GPXVR**
Set Extended View Representation

**RCP code**

201336846 (X'0C00280E')

# GPQDBK - Inquire Default Break Action

**GPQDBK (*wstype*, *errind*, *trigger*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQDBK** to inquire the default break action for the specified workstation type.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the graPHIGS API sets the error indicator (*errind*) to one of the error numbers to indicate the reason for the non-availability, and the values returned in the output parameters are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*trigger* **— returned by the graPHIGS API, 2 fullword integers**
Trigger to be used on the specified workstation for the break action. The trigger consists of a trigger type followed by a trigger qualifier. Positive integers as trigger types are button device numbers. The trigger qualifier for a button device is the choice alternative.

**Error Codes**

None

**Related Subroutines**

**GPBKAC**
Set Break Action

**GPQBK**
Inquire Break Capabilities

**GPQBKS**
Inquire Break Action State

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339665 (X'0C003311')

# GPQDCF - Inquire Depth Cue Facilities

**GPQDCF (*wstype*, *errind*, *number*, *npred*)**

> **Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQDCF** to inquire the depth cue facilities for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— specified by user, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*number* **— returned by the graPHIGS API, fullword integer**
Number of definable depth cue table entries. Entry 0 of the depth cue table may not be changed.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined depth cue indexes.

**Error Codes**

None

**Related Subroutines**

**GPDCI**
Set Depth Cue Index

**GPDCR**
Set Depth Cue Representation

**GPQDCR**
Inquire Depth Cue Representation

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201346053 (X'0C004C05')

## GPQDCH - Inquire Default Choice Device Data

**GPQDCH (*wstype*, *device*, *start*, *number*, *length*, *errind*, *choice*, *necho*, *echo*, *area*, *datalen*, *data*)**

> **Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQDCH** to inquire the default values of a requested choice device for the specified workstation type.

The graPHIGS API returns the default values for the requested choice device. For more information on the defaults, see *The graPHIGS Programming Interface: Technical Reference*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*device* **— specified by user, fullword integer**
Choice device number.

*start* **— specified by user, fullword integer**
Starting member of the list of prompt/echo types (>=1).

*number* **— specified by user, fullword integer**
Number of prompt/echo types requested (>=0).

*length* **— specified by user, fullword integer**
This refers to the length, in bytes, of the data array provided by the application for the graPHIGS API to return the data record.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*choice* **— returned by the graPHIGS API, fullword integer**
Maximum number of choice alternatives.

*necho* **— returned by the graPHIGS API, fullword integer**
Total number of available prompt/echo types.

*echo* **— returned by the graPHIGS API, array of fullword integers**
List of prompt/echo types. The output array must be large enough to contain the requested data.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Default echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax)

*datalen* **— returned by the graPHIGS API, fullword integer**
Default choice data record length, in bytes.

*data* **— returned by the graPHIGS API, variable length data**
Default choice data record for the default prompt/echo type (`1=TYPE`).

### Error Codes

None

### Related Subroutines

**GPINCH**
Initialize Choice

**GPQCH**
Inquire Choice Device State

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201340166 (X'0C003506')

## GPQDCM - Inquire Direct Color Model

**GPQDCM (*model*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQDCM** to inquire the current direct color model.

### Parameters

*model* **— returned by the graPHIGS API, fullword integer**
Current direct color model (`1=RGB, 2=HSV, 3=CMY, 4=CIELUV`).

### Error Codes

None

**Related Subroutines**

None

**RCP code**

201345798 (X'0C004B06')

## GPQDCR - Inquire Depth Cue Representation

**GPQDCR (*wsid*, *index*, *type*, *number*, *ids*, *errind*, *data*)**

> **Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQDCR** to inquire the current value of one or more fields in the specified depth cue table entry of the specified workstation's depth cue table. Each field is identified by a group identifier.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter in the requested order. The output parameter must be large enough to store all requested data.

If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Depth cue table index (>=0).

*type* **— specified by user, fullword integer**
Type of returned value (1=SET).

*number* **— specified by user, fullword integer**
Number of group identifiers requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**261** DEPTH CUE INDEX < ZERO

**262** DEPTH CUE INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**534** TYPE VALUE IS INVALID

*data* — **returned by the graPHIGS API, variable data.**
Data array containing the values in the requested groups. The value of each field is expressed in the data format listed below:

**Group identifier 1 - Depth cue mode**
A fullword integer (1=SUPPRESSED, 2=ALLOWED).

**Group identifier 2 - Depth cue reference planes**
Two short floating-point numbers specifying the far and near depth cue reference plane distance.

**Group identifier 3 - Depth cue scale factors**
Two short floating-point numbers specifying two scale factors corresponding to the far and near reference planes.

**Group identifier 4 - Depth cue color**
Four fullwords of data with either of the following two formats:

```
      indexed format                        direct format
      -------------                         -------------
   0 |           1| fullword integer      0 |           2| fullword integer
     |------------|                         |------------|
   4 | color index| fullword integer      4 | component 1| short floating-point number
     |------------|                         |------------|
   8 |     ignored| fullword integer      8 | component 2| short floating-point number
     |------------|                         |------------|
  12 |     ignored| fullword integer     12 | component 3| short floating-point number
     -------------                         -------------
```

### Error Codes

None

### Related Subroutines

**GPDCI**
Set Depth Cue Index

**GPDCR**
Set Depth Cue Representation

**GPQDCF**
Inquire Depth Cue Facilities

### RCP code

201339148 (X'0C00310C')

## GPQDDV - Inquire Default Deferral State Values

**GPQDDV (*wstype*, *errind*, *defer*, *modif*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQDDV** to inquire the default values of the deferral state and modification mode for the specified workstation type.

For information about deferral states and modification modes, see *The graPHIGS Programming Interface: Understanding Concepts*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*defer* **— returned by the graPHIGS API, fullword integer**
Default value for deferral mode (`1=ASAP, 2=BNIG, 3=BNIL, 4=ASTI, 5=WAIT`).

*modif* **— returned by the graPHIGS API, fullword integer**
Default value for modification mode (`1=NO_IMMEDIATE_VISUAL_EFFECT, 2=UPDATE_WITHOUT_REGENERATION, 3=QUICK_UPDATE`).

**Error Codes**

None

**Related Subroutines**

**GPDF** Set Deferral State

**GPQDV**
Inquire Deferral and Update State Values

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339393 (X'0C003201')

## GPQDIT - Inquire Default Input Device Triggers

**GPQDIT (*wstype*, *class*, *devnum*, *listid*, *start*, *number*, *errind*, *ndtrigs*, *dtriglist*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQDIT** to inquire the default trigger list of a specified input device for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*class* **— specified by user, fullword integer**
Input device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*devnum* **— specified by user, fullword integer**
Input device number (>=1).

*listid* **— specified by user, fullword integer**
Trigger list identifier for return of default trigger list (>=0). Trigger list identifier zero is always present and is called the primary trigger. The primary trigger causes the input to be returned to the application.

Secondary triggers may have different intermediate subroutines used in the processing of the input. They are identified with trigger list identifiers beginning with the value one.

*start* **— specified by user, fullword integer**
Starting member in the list of default triggers (>=1).

*number* **— specified by user, fullword integer**
Number of triggers requested from the trigger list (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

**570** SPECIFIED TRIGGER LIST IDENTIFIER DOES NOT EXIST

*ndtrigs* **— returned by the graPHIGS API, fullword integer**
Total number of triggers in the default trigger list.

*dtriglist* **— returned by the graPHIGS API, array of fullword integers**
List of default trigger descriptor triplets. The list is an array of trigger descriptors in which a descriptor consists of a triplet (3 fullword integers) containing the trigger type, low trigger qualifier, and high trigger qualifier. Positive integers as trigger types are choice device numbers. The trigger

qualifier for a choice device is the choice number. The parameter *ntrigs* identifies the total number of triplets in the available trigger list. The actual number returned will depend on the setting of the *start* and *number* parameters.

### Error Codes

None

### Related Subroutines

**GPIT** Set Input Device Trigger

**GPQITS**
Inquire Input Device Trigger State

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201339401 (X'0C003209')

## GPQDLC - Inquire Default Locator Device Data

**GPQDLC (*wstype*, *device*, *start*, *number*, *length*, *errind*, *dimen*, *pos*, *necho*, *echo*, *area*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQDLC** to inquire the default value of the requested locator device for the specified workstation type.

The graPHIGS API returns the default values for the requested locator device. The default initial locator position is in view zero, which is the only active view by default. For more information on the defaults, see *The graPHIGS Programming Interface: Technical Reference*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*device* **— specified by user, fullword integer**
Locator device number.

*start* **— specified by user, fullword integer**
Starting member of the list of prompt/echo types (>=1).

*number* **— specified by user, fullword integer**
Number of prompt/echo types requested (>=0)

*length* **— specified by user, fullword integer**
Length of locator data record array, in bytes, provided by the application for the graPHIGS API to return data record (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*dimen* **— returned by the graPHIGS API, fullword integer**
Type of locator device (`1=2D, 2=3D`).

*pos* **— returned by the graPHIGS API, 3 short floating-point numbers (WC)**
Default initial locator position.

*necho* **— returned by the graPHIGS API, fullword integer**
Total number of available prompt/echo types.

*echo* **— returned by the graPHIGS API, array of fullword integers**
List of available prompt/echo types requested. The output array must be large enough to contain the requested data.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Default echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*datalen* **— returned by the graPHIGS API, fullword integer**
Default locator data record length, in bytes.

*data* **— returned by the graPHIGS API, variable length data**
Default locator data record for the default prompt/echo type (`1=TYPE`).

**Error Codes**

None

**Related Subroutines**

**GPINLC**
Initialize Locator

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201340163 (X'0C003503')

# GPQDMR - Inquire Data Mapping Representation

**GPQDMR (*wsid, index, lclengths, lcdata, errind, method, mdata, clengths, ctype, cdata*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQDMR** to inquire the current values in the specified entry of the workstation's data mapping table.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough to hold the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

If the *method* parameter has the value `1=DM_METHOD_COLOR`, then the graPHIGS API returns no additional data in the remaining parameters.

The data mapping table is 0 based; entry 0 always contains a data mapping method of `1=DM_METHOD_COLOR`.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Data mapping index (>= 0).

*lclengths* **— specified by user, fullword integer**
Length, in bytes, of the specified *clengths* parameter (>=0).

*lcdata* **— specified by user, fullword integer**
Length, in bytes, of the specified *cdata* parameter (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**3** FUNCTION REQUIRES STATE WSOP
**25** SPECIFIED WORKSTATION DOES NOT EXIST
**35** WORKSTATION HAS ONLY INPUT CAPABILITIES
**50** DATA MAPPING INDEX EXCEEDS WORKSTATION TABLE CAPACITY
**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH
**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED
**630** DATA MAPPING INDEX < ZERO

*method* **— returned by the graPHIGS API, fullword integer**
Data mapping method (`-1=IMAGE_ARRAY`, `1=DM_METHOD_COLOR`, `2=SINGLE_VALUE_UNIFORM`, `4=BI_VALUE_UNIFORM`).

*mdata* **— returned by the graPHIGS API, variable data.**
Data mapping method descriptor. The data returned in this parameter has one of the following formats, depending on the returned data mapping method (*method*):

`-1=IMAGE_ARRAY`

The following data is returned:

| | | |
|---|---|---|
| WORDS 1 | 'udindex' | Fullword integer |
| 2 | 'vdindex' | Fullword integer |
| 3 | 'ulolim' | Floatingpoint number |
| 4 | 'uhilim' | Floatingpoint number |
| 5 | 'vlolim' | Floatingpoint number |
| 6 | 'vhilim' | Floatingpoint number |

*udindex*
Index into the primitive's data list for the *u* data value.

*vdindex*
Index into the primitive's data list for the *v* data value.

*ulolim* Lower limit of the *u* data mapping range.

*uhilim* Upper limit of the *u* data mapping range.

*vlolim* Lower limit of the *v* data mapping range.

*vhilim* Upper limit of the *v* data mapping range.

1=DM_METHOD_COLOR

N/A (No data mapping descriptor is returned for this method.)

2=SINGLE_VALUE_UNIFORM

The following data is returned:

| | | |
|---|---|---|
| WORDS 1 | 'dindex' | Fullword integer |
| 2 | 'lolim' | Floatingpoint number |
| 3 | 'hilim' | Floatingpoint number |

*dindex* Index into the primitive's data list.

*lolim* Lower limit of the data mapping range.

*hilim* Upper limit of the data mapping range.

4=BI_VALUE_UNIFORM

The data mapping record returned is identical to -1=IMAGE_ARRAY.

*clengths* **— returned by the graPHIGS API, variable data**
Data mapping color data lengths. The format of this parameter is dependent on the *method* parameter:

-1=IMAGE_ARRAY

The following data is returned:

| | | |
|---|---|---|
| WORDS 1 | 'x_size' | Fullword integer |
| 2 | 'y_size' | Fullword integer |
| 3 | 'oformat' | Fullword integer |

*x_size* *x* dimension of the base color data array.

*y_size* *y* dimension of the base color data array.

*oformat*

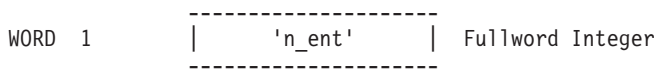
Data organization format (1=BASE_DATA, 2=SQUARE_MM, 3=RECT_MM). This format determines the filtering methods which may be used.

1=DM_METHOD_COLOR

No data is returned

2=SINGLE_VALUE_UNIFORM

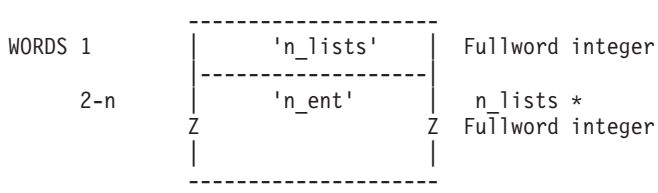
The following data is returned:




*n_ent* The number of entries in the color data list.

4=BI_VALUE_UNIFORM

The following data is returned:




*n_lists* The number of lists of data values.

*n_ent* A list of fullword integers that specify the number of entries of each color data list. There are *n_lists* entries in this list.

*ctype* **— returned by the graPHIGS API, fullword integer**

Data mapping color data type. Ignore this parameter if the method returned is 1=DM_METHOD_COLOR. Returned types are:

**1=TYPE_COLOR**

Colors consist of three short floating-point numbers in the current workstation color model (0.0<=color_component<=1.0).

**2=TYPE_PACKED_RGB**

Colors consist of four bytes. The first three bytes represent the red, green, and blue color components respectively; the fourth byte is ignored.

**3=TYPE_COLOR_TRANS**

Colors consist of four short floating-point numbers. The first three numbers represent the color in the current workstation color model (0.0<=color_component<=1.0); the fourth number represents the transparency coefficient (0.0<=transparency_coefficient<=1.0).

**4=TYPE_PACKED_RGB_ALPHA**

Colors consist of four bytes. The first three bytes represent the red, green, and blue color components respectively; the fourth byte is an unsigned integer alpha ([default]) blending value that may be derived from a transparency coefficient as follows:

```
Alpha=X'FF' x (1.0-transparency_coefficient)
```

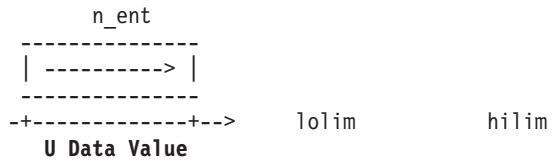
(Alpha=X'00') is fully transparent and equivalent to (transparency_coefficient=1.0). (Alpha=X'FF') is fully opaque and equivalent to (transparency_coefficient=0.0).

*cdata* **— returned by the graPHIGS API, variable data**

Data mapping color data. The data mapping color data organization is defined by the *clengths* and *ctype* parameters returned on this subroutine.

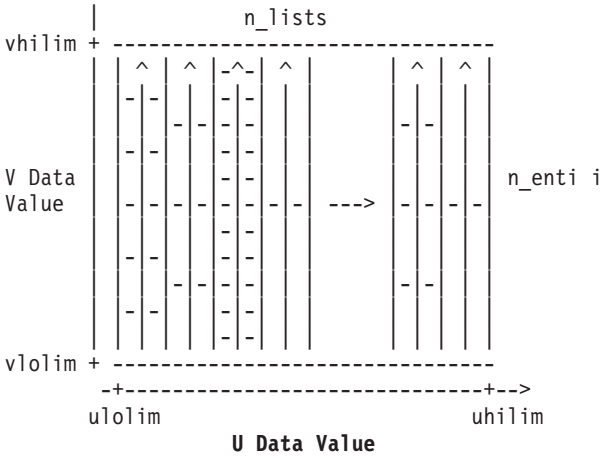
The 2=SINGLE_VALUE_UNIFORM and 4=BI_VALUE_UNIFORM color lists are supplied from the *lolim* lower limits to the *hilim* upper limits. For example, the color representing the data value *lolim* is first in each list, and the color representing *hilim* is last. See Figure 8 and Figure 9. The number of color values in each SINGLE_VALUE_UNIFORM color list is given by *n_ent*.

**Figure 8. SINGLE_VALUE_UNIFORM Color Data Organization**



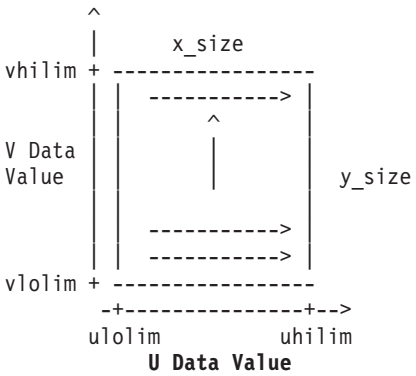

The number of color values in each of the BI_VALUE_UNIFORM color lists is specified by *n_ent[i]*, so that the total number of color values in this color array is ($n\_ent_1$ + $n\_ent_2$, + $n\_ent_3$ + ... + $n\_ent_{n\_lists}$)

**Figure 9. BI-VALUE_UNIFORM Color Data Organization**




The IMAGE_ARRAY color arrays are organized according to the *oformat* field of the *clengths* parameter. BASE_DATA array color data is supplied in row order left-to-right and bottom-to-top. See Figure 10. The number of color values in this array is: *(x_size * y_size)*

**Figure 10. IMAGE_ARRAY BASE_DATA Color Data Organization**



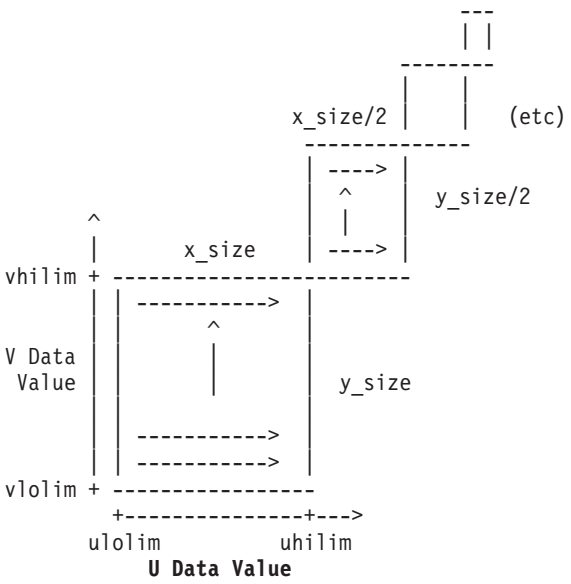

SQUARE_MM color mipmap data is supplied in the same fashion, starting with the base image and continuing with each successively smaller mipmap image, until either *x_size* or *y_size* is equal to one. See Figure 11. The number of color values in this complete array is

*(x_size * y_size) + (x_size * y_size)/4 + (x_size * y_size)/16 + ...*

which reduces to the integer portion of

*((4 * x_size * y_size) - (MAX(x_size,y_size) / MIN(x_size,y_size))) / 3*

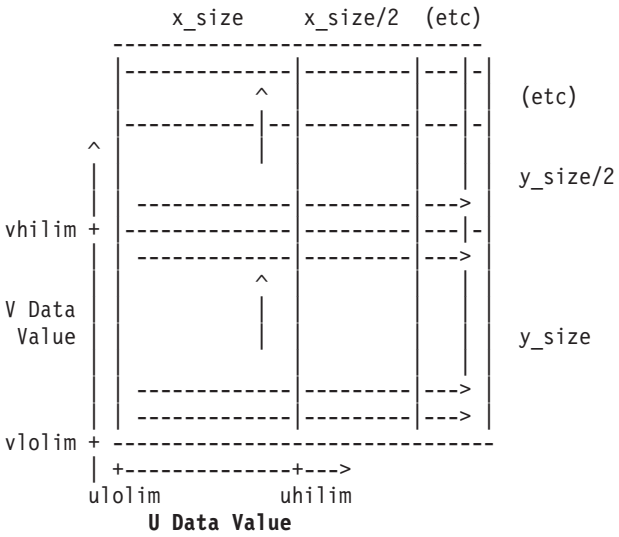
**Figure 11. IMAGE_ARRAY SQUARE_MM Color Data Organization**




RECT_MM color mipmap data is supplied as a complete set, and organized in row order left-to-right and bottom-to-top (as though the entire set of mipmap images constituted a single base texture image). See Figure 12. The number of color values in this complete array is:

*((2 * x_size) - 1) * ((2 * y_size) - 1)*

**Figure 12. IMAGE_ARRAY RECT_MM Color Data Organization**




## Error Codes

None

## Related Subroutines

**GPBDMI**
Set Back Data Mapping Index

**GPBTCO**
Set Back Transparency Coefficient

**GPDMI**
Set Data Mapping Index

**GPDMR**
Set Data Mapping Representation

**GPTCO**
Set Transparency Coefficient

**RCP code**

201339159 (X'0C003117')

## GPQDPK - Inquire Default Pick Device Data

**GPQDPK (*wstype*, *device*, *start*, *number*, *length*, *errind*, *maxpath*, *necho*, *echo*, *area*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQDPK** to return the default values of the specified pick device for the specified workstation type.

The graPHIGS API returns the default values for the requested pick device. For more information on defaults, see *The graPHIGS Programming Interface: Technical Reference*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*device* **— specified by user, fullword integer**
Pick device number.

*start* **— specified by user, fullword integer**
Starting member of the list of prompt/echo types (>=1).

*number* **— specified by user, fullword integer**
Number of prompt/echo types requested (>=0).

*length* **— specified by user, fullword integer**
Length of pick data record array, in bytes, provided by the application for the graPHIGS API to return data record.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*maxpath* **— returned by the graPHIGS API, fullword integer**
Maximum pick path depth.

*necho* **— returned by the graPHIGS API, fullword integer**
Total number of available prompt/echo types.

*echo* **— returned by the graPHIGS API, array of fullword integers**
List of available prompt/echo types. The output array must be large enough to contain the requested data.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Default echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*datalen* **— returned by the graPHIGS API, fullword integer**
Default pick data record length, in bytes.

*data* **— returned by the graPHIGS API, variable length data**
Default pick data record for the default prompt/echo type (`1=TYPE`).

**Error Codes**

None

**Related Subroutines**

**GPINPK**
Initialize Pick

**GPQPK**
Inquire Pick Device State

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201340162 (X'0C003502')

## GPQDS - Inquire Maximum Display Surface Size

**GPQDS (*wstype*, *errind*, *units*, *csize*, *asize*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQDS** to inquire the maximum display surface size for the specified workstation type.

The graPHIGS API returns the maximum display surface sizes in Device Coordinates (DC) and address units.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*units* **— returned by the graPHIGS API, fullword integer**
Device coordinate units (`1=METERS, 2=OTHER`).

*csize* **— returned by the graPHIGS API, 3 short floating-point numbers (DC)**
Maximum display surface size in *x*, *y*, and *z* directions.

*asize* **— returned by the graPHIGS API, 3 fullword integers**
Maximum display surface size in address units.

**Error Codes**

None

**Related Subroutines**

**GPINCH**
Initialize Choice

**GPINLC**
Initialize Locator

**GPINPK**
Initialize Pick

**GPINSK**
Initialize Stroke

**GPINST**
Initialize String

**GPINVL**
Initialize Valuator

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

# GPQDSK - Inquire Default Stroke Device Data

**GPQDSK (*wstype*, *device*, *start*, *number*, *length*, *errind*, *dimen*, *size*, *necho*, *echo*, *area*, *buflen*, *editpos*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQDSK** to inquire the default values of the requested stroke device for the specified workstation type.

The graPHIGS API returns the default values for the requested stroke device. For more information on defaults, see *The graPHIGS Programming Interface: Technical Reference*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*device* **— specified by user, fullword integer**
Stroke device number.

*start* **— specified by user, fullword integer**
Starting member of the list of prompt/echo types (>=1).

*number* **— specified by user, fullword integer**
Number of prompt/echo types requested (>=0).

*length* **— specified by user, fullword integer**
Length of stroke data record array, in bytes, provided by application for the graPHIGS API to return data record (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*dimen* **— returned by the graPHIGS API, fullword integer**
Type of stroke device (1=2D, 2=3D).

*size* **— returned by the graPHIGS API, fullword integer**
Maximum input buffer size.

*necho* **— returned by the graPHIGS API, fullword integer**
Total number of available prompt/echo types.

*echo* **— returned by the graPHIGS API, array of fullword integers**
List of available prompt/echo types. The output array must be large enough to contain the requested data.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Default echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*buflen* **— returned by the graPHIGS API, fullword integer**
Default initial stroke input buffer size, in points.

*editpos* **— returned by the graPHIGS API, fullword integer**
Default initial editing position.

*datalen* **— returned by the graPHIGS API, fullword integer**
Default stroke data record length, in bytes.

*data* **— returned by the graPHIGS API, variable length data**
Default stroke data record, in bytes, for the default prompt/echo type (1=TYPE).

**Error Codes**

None

**Related Subroutines**

**GPINSK**
Initialize Stroke

**GPQRCT**
Inquire Realized Connection Type

**GPQSK**
Inquire Stroke Device State

**RCP code**

201340164 (X'0C003504')

## GPQDST - Inquire Default String Device Data

**GPQDST (*wstype*, *device*, *start*, *number*, *length*, *errind*, *size*, *necho*, *echo*, *area*, *buflen*, *editpos*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQDST** to inquire the default values of the specified string device for the specified workstation type.

The graPHIGS API returns the default values for the requested string device. For more information on default values, see *The graPHIGS Programming Interface: Technical Reference*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*device* **— specified by user, fullword integer**
String device number.

*start* **— specified by user, fullword integer**
Starting member of the list of prompt/echo types (>=1).

*number* **— specified by user, fullword integer**
Number of prompt/echo types requested (>=0).

*length* **— specified by user, fullword integer**
This refers to the length of the array provided by the application for the graPHIGS API to return the data record, in bytes.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*size* **— returned by the graPHIGS API, fullword integer**
Maximum input buffer size.

*necho* **— returned by the graPHIGS API, fullword integer**
Total number of available prompt/echo types.

*echo* **— returned by the graPHIGS API, array of fullword integers.**
List of available prompt/echo types. The output array must be large enough to contain the requested data.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Default echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*buflen* **— returned by the graPHIGS API, fullword integer**
Default initial string input buffer size, in bytes.

*editpos* **— returned by the graPHIGS API, fullword integer**
Default initial cursor editing position.

*datalen* **— returned by the graPHIGS API, fullword integer**
Default string data record length, in bytes.

*data* **— returned by the graPHIGS API, variable length data**
Default string data record, in bytes for the default prompt/echo type (`1=TYPE`).

**Error Codes**

None

**Related Subroutines**

**GPINST**
Initialize String

**GPQRCT**
Inquire Realized Connection Type

**GPQST**
Inquire String Device State

**RCP code**

201340165 (X'0C003505')

## GPQDV - Inquire Deferral and Update State Values

**GPQDV (*wsid*, *errind*, *defer*, *modif*, *dissurf*, *dstat*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQDV** to inquire the current deferral and update state values for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*defer* **— returned by the graPHIGS API, fullword integer**
Deferral mode (1=ASAP, 2=BNIG, 3=BNIL, 4=ASTI, 5=WAIT).

*modif* **— returned by the graPHIGS API, fullword integer**
Modification mode (1=NO_IMMEDIATE_VISUAL_EFFECT, 2=UPDATE_WITHOUT_REGENERATION, 3=QUICK_UPDATE).

*dissurf* **— returned by the graPHIGS API, fullword integer**
Display surface empty (1=NOT_EMPTY, 2=IS_EMPTY).

*dstat* **— returned by the graPHIGS API, fullword integer**
Display status (1=CORRECT, 2=DEFERRED, 3=SIMULATED).

**Error Codes**

None

**Related Subroutines**

**GPDF** Set Deferral State

**GPQDDV**
Inquire Default Deferral State Values

**RCP code**

201336836 (X'0C002804')

## GPQDVL - Inquire Default Valuator Device Data

**GPQDVL (*wstype*, *device*, *start*, *number*, *length*, *errind*, *ivalue*, *necho*, *echo*, *area*, *lovalue*, *hivalue*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation Description Table WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQDVL** to inquire the default values of the specified valuator device for the specified workstation type.

The graPHIGS API returns the default values for the requested valuator device. For more information on default values, see *The graPHIGS Programming Interface: Technical Reference*.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator indicates that an output parameter is not large enough for the requested data, then the values are available up to the length specified. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* — **specified by user, 8-byte character string**
Workstation type.

*device* — **specified by user, fullword integer**
Valuator device number.

*start* — **specified by user, fullword integer**
Starting member of the list of prompt/echo types (>=1).

*number* — **specified by user, fullword integer**
Number of prompt/echo types requested (>=0).

*length* — **specified by user, fullword integer**
This refers to the length of the data array provided by the application for the graPHIGS API to return the data record, in bytes.

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*ivalue* — **returned by the graPHIGS API, short floating-point number**
Default initial value.

*necho* — **returned by the graPHIGS API, fullword integer**
Total number of available prompt/echo types.

*echo* — **returned by the graPHIGS API, array of fullword integers**
List of available prompt/echo types. The output array must be large enough to contain the requested data.

*area* — **returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Default echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*lovalue* — **returned by the graPHIGS API, short floating-point number**
Default low end of range for valuator.

*hivalue* — **returned by the graPHIGS API, short floating-point number**
Default high end of range for valuator.

*datalen* — **returned by the graPHIGS API, fullword integer**
Default valuator data record length, in bytes.

*data* — **returned by the graPHIGS API, variable length data**
Default valuator data record, in bytes for the default prompt/echo type (`1=TYPE`).

**Error Codes**

None

**Related Subroutines**

**GPINVL**
Initialize Valuator

**GPQRCT**
Inquire Realized Connection Type

**GPQVL**
Inquire Valuator Device State

**RCP code**

201340161 (X'0C003501')

## GPQED - Inquire List of Element Data

**GPQED (*number*, *buflen*, *errind*, *actnum*, *actlen*, *data*, *termcond*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

**Purpose**

Use **GPQED** to inquire the contents of one or more sequential structure elements starting at the one pointed to by the current element pointer. The elements will be returned in the format that is described in *The graPHIGS Programming Interface: Technical Reference*.

This subroutine does not move the element pointer.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Note:** The format of the element data that is returned by this subroutine is different from that returned by the Inquire Element Content (**GPQE**) subroutine.

**Parameters**

*number* **— specified by user, fullword integer**
Number of elements to be returned (>=1).

*buflen* **— specified by user, fullword integer**
Length, in bytes, of the data area specified by *data* into which the returned structure elements will be placed (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**4** FUNCTION REQUIRES STATE STOP

**535** CURRENT ELEMENT POINTER IS ZERO

**539** REQUESTED NUMBER < ZERO

**540** REQUESTED NUMBER < ONE

*actnum* **— returned by the graPHIGS API, fullword integer**
Total number of elements that have been returned.

*actlen* **— returned by the graPHIGS API, fullword integer**
Total length, in bytes, of the structure element data that has been returned in *data*.

*data* **— returned by the graPHIGS API, variable data**
The data buffer into which the structure elements are to be returned.

The structure elements are placed into this area so that the first structure element starts at the beginning of the area and each succeeding element can be reached by adding the length of the structure element to its offset into the buffer. Only complete structure elements will be returned. The format of each structure element is shown in *The graPHIGS Programming Interface: Technical Reference*.

*termcond* **— returned by the graPHIGS API, fullword integer**
Termination condition. The list of structure elements was terminated due to one of the following reasons:

**1-Count Exhausted**
The requested number of elements have been returned.

**2-Buffer Overflow**
The requested number of elements could not be returned because they would not all fit in the area provided. *actnum* will contain the actual number returned.

**3-End of Structure**
The last element of the structure was encountered. This condition supersedes the Count Exhausted condition (if that condition was in effect). Because of this, the total number of elements returned may or may not be equal to the requested number of elements to be returned so *actnum* should be checked to find the actual number of elements returned.

**4-Big Element**
The next element to be returned would not fit into the inbound buffer between the nucleus and shell. *actnum* will contain the number of elements preceding the one that would not fit. This number of elements will be in *data* also.

### Error Codes

None

### Related Subroutines

**GPQE** Inquire Element Content

**GPQEHD**
Inquire List of Element Headers

**GPQETS**
Inquire Element Type and Size

### RCP code

201337100 (X'0C00290C')

## GPQEDA - Inquire List of Element Data for any Structure

**GPQEDA (*strid*, *start*, *number*, *buflen*, *errind*, *actnum*, *actlen*, *data*, *termcond*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

**Purpose**

Use **GPQEDA** to inquire the contents of one or more sequential structure elements from a specified structure and element position. The elements will be returned in the format that is described in *The graPHIGS Programming Interface: Technical Reference*. If the specified element position is greater than the number of elements in the structure or is less than one then the error indicator will be non-zero.

If the inquired information is available, the error indicator is returned as zero, and the values are returned in the output parameters. If the inquired information is unavailable, the error indicator contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Note:** The format of the element data that is returned by this subroutine is the same as that returned by the Inquire Element Data (**GPQED**) subroutine.

**Parameters**

*strid* **— specified by user, fullword integer**
Structure identifier.

*start* **— specified by user, fullword integer**
The position of the first element whose element data is to be returned (>=1).

*number* **— specified by user, fullword integer**
Number of elements requested (>=1).

*buflen* **— specified by user, fullword integer**
Length in bytes of the *data* parameter.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**524** ELEMENT POSITION > NUMBER OF ELEMENTS IN STRUCTURE

**538** START VALUE < ONE

**540** REQUESTED NUMBER < ONE

**577** BUFFER LENGTH IS < ZERO

*actnum* **— returned by the graPHIGS API, fullword integer**
Total number of elements actually returned.

*actlen* **— returned by the graPHIGS API, fullword integer**
Total length in bytes of the element data returned in the *data* parameter.

*data* **— returned by the graPHIGS API, variable data**
The data buffer into which the structure elements are to be returned. The structure elements are placed into this area so that the first structure element starts at the beginning of the area and each succeeding element can be reached by adding the length of the structure element to its offset into the buffer. Only complete structure elements will be returned. The format of each structure element is shown in *The graPHIGS Programming Interface: Technical Reference*.

*termcond* **— returned by the graPHIGS API, fullword integer**
Termination condition. The list of structure elements was terminated due to one of the following reasons:

**1- Count Exhausted**
The requested number of elements have been returned.

**2- Buffer Overflow**
The requested number of elements could not be returned because they would not all fit in the area provided. *actnum* will contain the actual number returned.

**3- End of Structure**
The last element of the structure was encountered. This condition supersedes the Count Exhausted condition (if that condition was in effect). Because of this, the total number of elements may or may not be equal to the requested number of elements to be returned so *actnum* should be checked to find the actual number of elements returned.

**4- Big Element**
The next element would not fit into the inbound buffer between the nucleus and the shell. *actnum* contains the number of elements preceding the one that would not fit. This number of elements will be in *data* also.

### Error Codes

None

### Related Subroutines

**GPQED**
Inquire List of Element Data

**GPQEHD**
Inquire List of Element Headers

### RCP code

201337103 (X'0C00290F')

## GPQEDM - Inquire Edit Mode

**GPQEDM (*mode*)**

**Note:** This subroutine is a graPHIGS API tate List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQEDM** to inquire the current edit mode.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*mode* **— returned by the graPHIGS API, fullword integer**
Current edit mode (1=INSERT_MODE, 2=REPLACE_MODE).

**Error Codes**

None

**Related Subroutines**

**GPEDMO**
Set Edit Mode

**RCP code**

201336324 (X'0C002604')

## GPQEF - Inquire Edge Facilities

**GPQEF (*wstype*, *start*, *number*, *errind*, *netype*, *eltype*, *nelwidth*, *elwidth*, *minelw*, *maxelw*, *npred*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQEF** to inquire the edge facilities for the specified workstation type.

The graPHIGS API returns a number indicating the total number of available line types and their identifiers (*netype*[default] the available number of line widths (*nelwidth*), and the nominal (*eltype*), minimum (*minelw*), and maximum values (*maxelw*) and the number of predefined edge indexes (*npred*). The graPHIGS API returns the width of lines in Device Coordinate (DC) units.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of line types of edges (>=1).

*number* **— specified by user, fullword integer**
Number of line types of edges requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*netype* **— returned by the graPHIGS API, fullword integer**
Total number of available line types of edges.

*eltype* **— returned by the graPHIGS API, array of fullword integers.**
List of available edge line types in the workstation's available edge line type table. The table size and specific entries supported are workstation dependent. The default edge line type table for supported entries is defined with the following line types: (1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE). The output array must be large enough to contain the requested data.

*nelwidth* **— returned by the graPHIGS API, fullword integer**
Number of available line widths. (Zero means that the workstation supports a continuous range of line widths of edges.)

*elwidth* **— returned by the graPHIGS API, short floating-point number (DC)**
Nominal line width of edge.

*minelw* **— returned by the graPHIGS API, short floating-point number (DC)**
Minimum linewidth of edge.

*maxelw* **— returned by the graPHIGS API, short floating-point number (DC)**
Maximum linewidth of edge.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined edge bundle table entries.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**GPQXER**
Inquire Extended Edge Representation

**GPXER**
Set Extended Edge Representation

**RCP code**

201339669 (X'0C003315')

## GPQEHA - Inquire List of Element Headers for any Structure

**GPQEHA (*strid*, *start*, *number*, *errind*, *actnum*, *header*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

**Purpose**

Use **GPQEHA** to inquire a list of element headers from a specified structure. This subroutine returns the list in sequential order starting from a specified element position.

- If the specified element position is greater than the number of elements in the structure, then the *errind* parameter will contain an error.
- If the specified element position is less than one, the *errind* parameter will contain an error.
- If the requested number is larger than the number of elements between the current element and the end of the structure, only information about the existing elements is returned and the actual number (*actnum*) parameter is set to the number of element codes.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

> **Note:** The format of the element headers that are returned by this subroutine is the same as that returned by the Inquire List of Element Headers (**GPQEHD**) subroutine.

**Parameters**

*strid* **— specified by user, fullword integer**
Structure identifier.

*start* **— specified by user, fullword integer**
The position of the first element whose header is to be returned (>=1).

*number* **— specified by user, fullword integer**
Number of element headers requested (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**524** ELEMENT POSITION > NUMBER OF ELEMENTS IN STRUCTURE

**538** START VALUE < ONE

**540** REQUESTED NUMBER < ONE

*actnum* **— returned by the graPHIGS API, fullword integer**
Number of element headers actually returned.

*header* **— returned by the graPHIGS API, fullword integer**
A list of element headers. The first halfword of each element header contains the length of the element and the second halfword contains the element code. The information that corresponds to each element header and the list of element codes used by graPHIGS API can be found in *The graPHIGS Programming Interface: Technical Reference*.

**Error Codes**

None

**Related Subroutines**

**GPQED**
Inquire List of Element Data

**GPQEDA**
Inquire List of Element Data for any Structure

**GPQEHD**
Inquire List of Element Headers

**RCP code**

201337104 (X'0C002910')

## GPQEHD - Inquire List of Element Headers

**GPQEHD (*number*, *errind*, *actnum*, *header*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPQEHD** to inquire a list of element headers from the current open structure. This subroutine returns the list in the sequential order starting from the current element pointer. If the requested number is larger than the number of elements between the current element and the end of the open structure, only information about the existing elements is returned and the actual number parameter is set to the number of actually returned element codes.

This subroutine does not move the element pointer.

If the inquired information is available, the error indicator is returned as zero, and the values are returned in the output parameters. If the inquired information is unavailable, the error indicator contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Note:** The format of the element headers that is returned by this subroutine is different from the information that is returned by the older Inquire Element Type and Size (**GPQETS**) subroutine.

### Parameters

*number* **— specified by user, fullword integer**
Number of entries requested (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**4** FUNCTION REQUIRES STATE STOP

**535** CURRENT ELEMENT POINTER IS ZERO

**540** REQUESTED NUMBER < ONE

*actnum* **— returned by the graPHIGS API, fullword integer**
Number of entries actually returned.

*header* **— returned by the graPHIGS API, array of 2 halfword integers**
List of element headers. The first and second halfword integer of each element header indicates the length and code of the structure element, respectively.

The information that corresponds to each element and length and the list of element codes used by the graPHIGS API can be found in *The graPHIGS Programming Interface: Technical Reference*.

### Error Codes

None

### Related Subroutines

**GPQE** Inquire Element Content

**GPQED**
Inquire List of Element Data

**GPQETS**
Inquire Element Type and Size

### RCP code

201337101 (X'0C00290D')

## GPQEMO - Inquire Error Handling Mode

**GPQEMO (*mode*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQEMO** to inquire if the error handling mode was set `1=OFF` or `2=ON`.

The mode can be set using the Set Error Handling Mode (**GPEMO**) subroutine. The default error handling mode is `2=ON`.

### Parameters

*mode* **— returned by the graPHIGS API, fullword integer**
Error handling mode (`1=OFF`, `2=ON`).

### Error Codes

None

### Related Subroutines

**GPEMO**
Set Error Handling Mode

### RCP code

201337861 (X'0C002C05')

## GPQEMS - Inquire Error Message

**GPQEMS (*length*, *errind*, *number*, *message*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQEMS** to inquire the current error message. If an error was generated, the graPHIGS API returns the number of bytes in the error message and the error message text.

If the inquired information is available, the error indicator is returned as zero, and the values are returned in the output parameters. If the error indicator indicates that an output parameter is not large enough for the requested data, then the values will be available up to the length specified. If the inquired information is unavailable, the error indicator contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*length* **— specified by user, fullword integer**
Length of error message array in bytes provided by the application for the graPHIGS API to return message.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**505** LENGTH OF RETURN ARRAY < ZERO

**521** NOT IN ERROR STATE

*number* **— returned by the graPHIGS API, fullword integer**
Total number of bytes in error message text.

*message* **— returned by the graPHIGS API, variable length character string**
Message text.

### Error Codes

None

### Related Subroutines

None

### RCP code

201337859 (X'0C002C03')

## GPQEP - Inquire Element Pointer

**GPQEP (*errind*, *value*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPQEP** to inquire the value of the current element pointer in the currently selected structure store.

A structure must be open to invoke this subroutine.

If the required information is available, the error indicator is returned as zero, and the value is returned in the output parameters. If a structure is not open, then the error indicator contains an error number indicating the reason, and the value returned in the output parameter is unpredictable.

**Parameters**

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**4** FUNCTION REQUIRES STATE STOP

*value* **— returned by the graPHIGS API, fullword integer**
Current element pointer value.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201337091 (X'0C002903')

## GPQES - Inquire List of Available Escape Subroutines

**GPQES (*wstype*, *start*, *number*, *errind*, *totnum*, *escid*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQES** to obtain a list of the escape function identifiers for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member in the list of escape subroutine identifiers (>=1).

*number* **— specified by user, fullword integer**
Total number of escape subroutine identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available escape subroutine identifiers.

*escid* **— returned by the graPHIGS API, array of fullword integers**
List of available escape subroutine identifiers:

- 1001 - Sound Alarm
- 1002 - Enable/Disable Link Switch Notification
- 1003 - Plot Size
- 1004 - Initialize Pick Correlation State
- 1005 - Set Pick Selection Criteria
- 1006 - Set Input Echo Color
- 1007 - Read Frame Buffer
- 1008 - Geometric Text Culling
- 1009 - Window Resize Notification Control
- 1010 - Inquire Mapped Display Surface Size
- 1011 - Window Exposure Notification Control
- 1012 - Window Deletion Notification Control
- 1014 - Workstation-Dependent Output
- 1015 - Convert Coordinate Values

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201340167 (X'0C003507')

## GPQEXS - Inquire Executed Structures

**GPQEXS (*strid*, *start*, *number*, *errind*, *totnum*, *lstrid*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPQEXS** to inquire the list of structure identifiers that exist in execute structure-type elements (execute structure elements and conditional execute structure elements) within the specified structure of the currently selected structure store.

If the inquired information is available, the error indicator is returned as zero, and the values are returned in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total-number parameter is set. If the inquired information is unavailable, the error indicator contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*strid* **— specified by user, fullword integer**
Structure identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of structure identifier (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of structure identifiers contained in the execute structure elements in the specified structure.

*lstrid* **— returned by the graPHIGS API, array of fullword integers**
List of structure identifiers.

**Error Codes**

None

**Related Subroutines**

**GPEXST**
Execute Structure

**GPCEXS**
Conditional Execute Structure

**GPQRST**
Inquire Referencing Structures

**RCP code**

201337093 (X'0C002905')

# GPQFAR - Inquire Font Aspect Ratios

**GPQFAR (*csid*, *font*, *slength*, *string*, *errind*, *aspect-ratio-list*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQFAR** to inquire the font aspect ratios for the characters in a specified character string using the specified character set and font.

The list of returned ratios corresponds to the specified character set and font. Aspect ratios are specified as the ratio of the font's nominal width relative to the font's nominal height. The user requests the character set and font. The ratios are workstation independent.

If the inquired information is available, the error indicator is returned as zero, and the values are returned in the output parameter. If the inquired information is unavailable, the error indicator contains an error number indicating the reason, and the value returned in the output parameter is unpredictable.

### Parameters

*csid* **— specified by user, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

*font* **— specified by user, fullword integer**
Font identifier (>=1).

*slength* **— specified by user, fullword integer**
Length of the specified text string in bytes (>=0).

*string* **— specified by user, variable length character string**
Character string for which aspect ratios are requested.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**75** TEXT FONT VALUE IS INVALID

**108** NUMBER OF CHARACTERS IN TEXT STRING < ZERO

**542** CHARACTER SET IDENTIFIER IS INVALID

**560** CHARACTER SET/FONT COMBINATION IS NOT AVAILABLE

**564** TEXT STRING CONTAINS AN UNSUPPORTED CHARACTER CODE

*aspect-ratio-list* **— returned by the graPHIGS API, array of short floating-point numbers**
List of corresponding aspect ratios of the requested text string for the specified character set and font. The array must be large enough to hold the required data.

### Error Codes

None

### Related Subroutines

None

**RCP code**

201336330 (X'0C00260A')

## GPQFBC - Inquire Frame Buffer Characteristics

**GPQFBC (*wstype*, *errind*, *org*, *n*, *depth*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQFBC** to inquire frame buffer characteristics for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*org* **— returned by the graPHIGS API, fullword integer**
Frame buffer organization (`1=COMPONENT, 2=INDEXED`).

*n* **— returned by the graPHIGS API, fullword integer**
Number of frame buffer components (>=0).

*depth* **— returned by the graPHIGS API, array of fullword integers**
List of bit depths for frame buffer components. For this parameter, the application must supply an array large enough to contain the maximum entries which may be returned. A component frame buffer has three components. An indexed frame buffer has one component.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**GPRDFB**
Read Frame Buffer

**RCP code**

201339407 (X'0C00320F')

## GPQFCH - Inquire Font Characteristics

**GPQFCH (*csid*, *font*, *errind*, *proportional*, *top*, *bottom*, *nomaspect*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

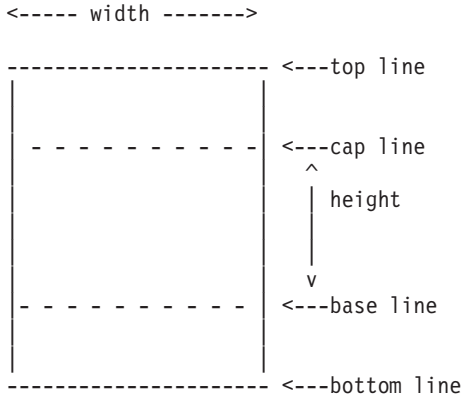
**Purpose**

Use **GPQFCH** to inquire the font characteristics for the requested character set and font.

The graPHIGS API returns data indicating the top and bottom of the character cell offset from the cap line respectively, the nominal aspect ratio relative to the character's height, and if the font is fixed or proportionally sized. For more information, see *The graPHIGS Programming Interface: Understanding Concepts*.

If the inquired information is available, the error indicator is returned as zero, and the value is returned in the output parameter. If the inquired information is unavailable, the error indicator contains an error number indicating the reason, and the value returned in the output parameter is unpredictable.

Annotation text capabilities are defined in terms of the following character box description:




Height is defined as the distance between the base line and the cap line.

**Note:** The definition of height is different from that used by the Inquire Annotation Font Characteristics (**GPQAFC**) subroutine.

**Parameters**

*csid* **— specified by user, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

*font* **— specified by user, fullword integer**
Font identifier (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**75** TEXT FONT VALUE IS INVALID

**542** CHARACTER SET IDENTIFIER IS INVALID

**560** CHARACTER SET/FONT COMBINATION IS NOT AVAILABLE

*proportional* **— returned by the graPHIGS API, fullword integer**
Font is defined as fixed or proportionally sized (`1=FIXED, 2=PROPORTIONAL`).

*top* **— returned by the graPHIGS API, short floating-point number**
Top offset from the cap line in the character cell specified as a fraction of the character's height.

*bottom* **— returned by the graPHIGS API, short floating-point number**
Bottom offset from the base line in the character cell specified as a fraction of the character's height.

*nomaspect* **— returned by the graPHIGS API, short floating-point number**
The font nominal aspect ratio.

This is the nominal width to height ratio.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201336331 (X'0C00260B')

## GPQFO - Inquire Active Fonts

**GPQFO (*wsid*, *start*, *number*, *errind*, *nfont*, *lcsid*, *lfont*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQFO** to inquire the currently active geometric character set and font identifiers, that is, the geometric font pool contents, for the specified workstation.

The graPHIGS API returns data indicating the total quantity of active geometric fonts and their character set and font identifiers.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the

available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of active fonts (>=1).

*number* **— specified by user, fullword integer**
Number of active fonts requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*nfont* **— returned by the graPHIGS API, fullword integer**
Total number of active geometric fonts.

*lcsid* **— returned by the graPHIGS API, array of fullword integers.**
List of active geometric character set identifiers. The output array must be large enough to contain the requested data.

See Appendix A. "Character Set and Font Identifiers" for more information.

*lfont* **— returned by the graPHIGS API, array of fullword integers.**
List of active geometric fonts. The output array must be large enough to contain the requested data.

> **Note:** Entry *x* of the *lcsid* list and entry *x* of the *lfont* list combine to describe entry *x* of the active geometric font pool on the specified workstation.

**Error Codes**

None

**Related Subroutines**

**GPACFO**
Activate Font

**GPDAFO**
Deactivate Font

**GPLDFO**
Load Font

**RCP code**

201336598 (X'0C002716')

## GPQFP - Inquire Font Pool Size

**GPQFP (*wstype*, *errind*, *poolsize*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQFP** to inquire the maximum font pool size for the specified workstation type.

The graPHIGS API returns data indicating the maximum number of simultaneously active fonts for the specified workstation type.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*poolsize* **— returned by the graPHIGS API, fullword integer**
Maximum font pool size from the workstation description table.

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201339671 (X'0C003317')

## GPQGD - Inquire List of Generalized Drawing Primitives

**GPQGD (*wstype*, *start*, *number*, *errind*, *totnum*, *gdpid*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQGD** to inquire the available Generalized Drawing Primitives (GDPs) for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of GDP identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of GDP identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available GDPs.

*gdpid* **— returned by the graPHIGS API, array of fullword integers**
List of GDP identifiers. One of the following GPD identifiers listed may be returned:

- 1001 - Pixel 3
- 1002 - Pixel 2
- 1003 - Disjoint Polyline 3
- 1004 - Disjoint Polyline 2
- 1005 - Circle 2
- 1006 - Circular Arc 2
- 1007 - Ellipse 2
- 1008 - Ellipse 3
- 1009 - Elliptical Arc 2
- 1010 - Elliptical Arc 3
- 1014 - Polyline Set 3 With Data
- 1016 - Polygon 3 With Data
- 1017 - Polygon 2 With Data
- 1020 - Marker Grid 3
- 1021 - Marker Grid 2

- 1022 - Line Grid 3
- 1023 - Line Grid 2
- 1027 - Composite Fill Area 2
- 1029 - Triangle Strip 3
- 1031 - Quadrilateral Mesh 3
- 1033 - Non-Uniform B-Spline Curve 3
- 1034 - Non-Uniform B-Spline Curve 2
- 1035 - Non-Uniform B-Spline Surface
- 1036 - Trimmed Non-Uniform B-Spline Surface
- 1037 - Polyhedron Edge
- 1039 - Character Line 2
- 1046 - Polysphere

The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339660 (X'0C00330C')

## GPQGDP - Inquire Generalized Drawing Primitive

**GPQGDP (*wstype*, *gdpid*, *errind*, *number*, *list*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQGDP** to inquire the list of attributes used by the specified Generalized Drawing Primitive (GDP) for the specified workstation type.

The graPHIGS API returns a list of the attributes used by the specified GDP.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*gdpid* **— specified by user, fullword integer**
List of GDP identifiers. One of the following GDP identifiers listed may be specified:

- 1001 - Pixel 3
- 1002 - Pixel 2
- 1003 - Disjoint Polyline 3
- 1004 - Disjoint Polyline 2
- 1005 - Circle 2
- 1006 - Circular Arc 2
- 1007 - Ellipse 2
- 1008 - Ellipse 3
- 1009 - Elliptical Arc 2
- 1010 - Elliptical Arc 3
- 1014 - Polyline Set 3 With Data
- 1016 - Polygon 3 With Data
- 1017 - Polygon 2 With Data
- 1020 - Marker Grid 3
- 1021 - Marker Grid 2
- 1022 - Line Grid 3
- 1023 - Line Grid 2
- 1027 - Composite Fill Area 2
- 1029 - Triangle Strip 3
- 1031 - Quadrilateral Mesh 3
- 1033 - Non-Uniform B-Spline Curve 3
- 1034 - Non-Uniform B-Spline Curve 2
- 1035 - Non-Uniform B-Spline Surface
- 1036 - Trimmed Non-Uniform B-Spline Surface
- 1037 - Polyhedron Edge
- 1039 - Character Line 2
- 1046 - Polysphere

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**41** WORKSTATION TYPE CANNOT GENERATE SPECIFIED GDP

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*number* **— returned by the graPHIGS API, fullword integer**
Number of sets of attributes used.

*list* **— returned by the graPHIGS API, 5 fullword integers**
List of sets of attributes used by the specified GDP (`1=POLYLINE, 2=POLYMARKER, 3=TEXT, 4=INTERIOR, 5=EDGE`).

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339394 (X'0C003202')

## GPQGFC - Inquire Geometric Font Characteristics

**GPQGFC (*wsid*, *csid*, *font*, *start*, *num*, *errind*, *prec*, *nhts*, *lhts*, *lnfac*, *lmnfac*, *lmxfac*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQGFC** to inquire the geometric text capabilities for the specified character set and font on the specified workstation.

This information consists of the highest text precision for this *csid*/*font* that is supported by the specified workstation (*prec*), the number of supported character heights (*nhts*), a list of the exact supported character heights (*lhts*), and the number of character expansion factors (*lnfac*), minimum expansion factor (*lmnfac*), and maximum expansion factor (*lmxfac*) corresponding to each supported character height.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*csid* **— specified by user, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

*font* **— specified by user, fullword integer**
Font identifier (>=1).

*start* **— specified by user, fullword integer**
Starting member of the list of supported heights (>=1).

*num* **— specified by user, fullword integer**
Number of list elements requested (>=0).

**Note:** The four output arrays must be large enough to hold the requested number of elements.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**75** TEXT FONT VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**542** CHARACTER SET IDENTIFIER IS INVALID

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**561** CHARACTER SET/FONT COMBINATION IS NOT ACTIVE

**563** CHARACTER SET/FONT COMBINATION IS NOT AVAILABLE FOR GEOMETRIC TEXT

*prec* **— returned by the graPHIGS API, fullword integer**
Highest available precision for the corresponding *csid*/*font* (`1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC`).

> **Note:** If the highest precision supported is `3=STROKE_PREC`, then the parameters that follow refer to the geometric text capabilities in character precision.

*nhts* **— returned by the graPHIGS API, fullword integer**
Total number of available character heights supported for the specified character set and font on the specified workstation.

A value of zero means that a continuous range of heights is supported. The lists describe the minimum and maximum character heights.

*lhts* **— returned by the graPHIGS API, array of short floating-point numbers (DC)**
List of all character heights supported on this workstation for geometric text in Device Coordinates (DC).

For a continuous range of character heights, this list contains the minimum and maximum character heights.

*lnfac* **— returned by the graPHIGS API, array of fullword integers**
List of the total number of available character expansion factors. Each element in the list corresponds to an element in the list of character heights.

A value of zero means that the workstation supports a continuous range of character expansion factors.

*lmnfac* **— returned by the graPHIGS API, array of short floating-point numbers**
List of the minimum character expansion factors supported. There is one list entry for each entry in the list of supported heights.

*lmxfac* **— returned by the graPHIGS API, array of short floating-point numbers**
List of the maximum character expansion factors supported. There is one list entry for each entry in the list of supported heights.

### Error Codes

None

### Related Subroutines

**GPCHH**
Set Character Height

**GPCHXP**
Set Character Expansion Factor

**GPQFO**
Inquire Active Fonts

**RCP code**

201336590 (X'0C00270E')

## GPQGSE - Inquire List of Available GSEs

**GPQGSE (*wstype*, *start*, *number*, *errind*, *totnum*, *gseid*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQGSE** to inquire a list of GSE identifiers available for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of GSEs (>=1).

*number* **— specified by user, fullword integer**
Number of GSE identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available GSEs.

*gseid* **— returned by the graPHIGS API, array of fullword integers**
List of GSE identifiers. One or more of the following GSE identifiers may be returned:

- 1001 - Set Frame Buffer Protect Mask
- 1002 - Set Frame Buffer Comparison

- 1003 - Set Condition
- 1004 - Conditional Execute Structure
- 1005 - Conditional Return
- 1006 - Test Extent 3
- 1007 - Test Extent 2
- 1008 - Parametric Surface Characteristics
- 1009 - Z-buffer Protect Mask
- 1010 - Workstation-Dependent Output

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339406 (X'0C00320E')

## GPQHD - Inquire Maximum Hierarchy Depth

**GPQHD (*wstype*, *errind*, *depth*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQHD** to inquire the maximum hierarchy depth for the specified workstation type.

If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*depth* **— returned by the graPHIGS API, fullword integer**
Maximum hierarchy depth.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339395 (X'0C003203')

## GPQHF - Inquire Hatch Facilities

**GPQHF (*wstype*, *errind*, *format*, *maxlen*, *npred*, *available*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQHF** to inquire the hatch facilities for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the output parameter *available*, is set to 1, then the values returned in the other output parameters are unpredictable. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*format* **— returned by the graPHIGS API, fullword integer**
Hatch definition format(`1=BIT_ARRAY`).

*maxlen* **— returned by the graPHIGS API, fullword integer**
Maximum length of hatch definition data.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined hatch patterns.

*available* **— returned by the graPHIGS API, fullword integer**
Availability of the Set Hatch Representation (**GPHR**) subroutine or Inquire Hatch Representation (**GPQHR**) subroutine subroutine (`1=NOT_AVAILABLE, 2=BOTH_AVAILABLE, 3=INQUIRE_ONLY_AVAILABLE, 4=SET_ONLY_AVAILABLE`).

**Error Codes**

None

**Related Subroutines**

**GPHR** Set Hatch Representation

**GPQHR**
Inquire Hatch Representation

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201346051 (X'0C004C03')

## GPQHLF - Inquire Highlighting Filter

**GPQHLF (*wsid*, *inlen*, *exlen*, *errind*, *inclen*, *incl*, *exclen*, *excl*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQHLF** to inquire the current highlighting inclusion and exclusion filters on the specified workstation.

The graPHIGS API returns the size of the inclusion filter and its contents and the size of the exclusion filter and its contents.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*inlen* **— specified by user, fullword integer**
Size of inclusion filter output array, specified in fullword integers (>=0).

This is the size of the array provided by the application for the graPHIGS API to return the corresponding data.

*exlen* **— specified by user, fullword integer**
Size of exclusion filter output array, specified in fullword integers (>=0).

This is the size of the array provided by the application for the graPHIGS API to return the corresponding data.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**505** LENGTH OF RETURN ARRAY < ZERO

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

*inclen* **— returned by the graPHIGS API, fullword integer**
Returned inclusion filter size.

*incl* **— returned by the graPHIGS API, array of fullword integers.**
Inclusion filter. List of class names.

*exclen* **— returned by the graPHIGS API, fullword integer**
Returned exclusion filter size.

*excl* **— returned by the graPHIGS API, array of fullword integers.**
Exclusion filter. List of class names.

**Error Codes**

None

**Related Subroutines**

**GPHLF**
Set Highlighting Filter

**GPQNCN**
Inquire Number of Available Class Names

**RCP code**

201336837 (X'0C002805')

## GPQHMO - Inquire Available HLHSR Modes

**GPQHMO (*wstype*, *start*, *number*, *errind*, *totnum*, *mode*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQHMO** to inquire the hidden line hidden surface removal (HLHSR) facilities for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available HLHSR modes (>=1).

*number* **— specified by user, fullword integer**
Number of HLHSR modes requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available HLHSR modes supported by a workstation.

*mode* **— returned by the graPHIGS API, array of fullword integers**
List of available HLHSR modes (`1=OFF, 2=ON_THE_FLY`).

**Error Codes**

None

**Related Subroutines**

**GPQCVR**
Inquire Current View Representation

**GPQRCT**
Inquire Realized Connection Type

**GPQRVR**
Inquire Requested View Representation

**GPXVR**
Set Extended View Representation

**RCP code**

201339402 (X'0C00320A')

## GPQHR - Inquire Hatch Representation

**GPQHR (*wsid*, *index*, *errind*, *format*, *length*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQHR** to inquire the current hatch pattern in the specified entry of the workstation's hatch table.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Hatch table index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**84** INTERIOR STYLE INDEX VALUE < ONE

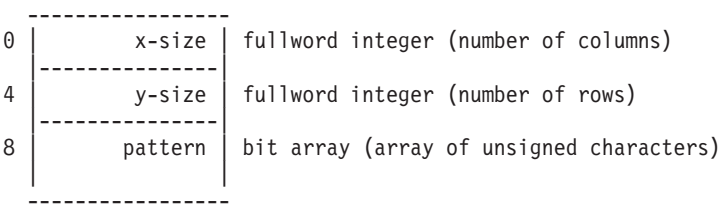
**274** THIS FUNCTION IS NOT SUPPORTED BY THE WORKSTATION

*format* **— returned by the graPHIGS API, fullword integer**
Hatch pattern format (`1=BIT_ARRAY`).

*length* **— returned by the graPHIGS API, fullword integer**
Length of the hatch pattern definition.

*data* **— returned by the graPHIGS API, variable length data**
Hatch pattern definition data. Values returned to this parameter depend on the hatch pattern format. The application must supply storage for this parameter that is large enough to contain the maximum data that the specified workstation supports (see the Inquire Hatch Facilities [**GPQHF**] [page GPQHF - Inquire Hatch Facilities]) subroutine.

**1=BIT ARRAY**




**Note:** The bit array will be in row order with each row beginning on a byte boundary. Therefore, the size of the bit array will be ((x-size + 7)/8 * y-size ) bytes.

**Error Codes**

None

**Related Subroutines**

**GPHR** Set Hatch Representation

**GPIS** Set Interior Style

**GPISI** Set Interior Style Index

**GPXIR**
Set Extended Interior Representation

**RCP code**

201339149 (X'0C00310D')

## GPQIBC - Inquire Image Board Characteristics

**GPQIBC (*ibid*, *errind*, *depth*, *h*, *v*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQIBC** to inquire the image board characteristics for the specified image board: the image board was created by the Create Image Board (**GPCRIB**)subroutine.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*ibid* **— specified by user, fullword integer**
Image board identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

*depth* **— returned by the graPHIGS API, fullword integer**
Bit depth of the image board.

*h* **— returned by the graPHIGS API, fullword integer**
Horizontal size of the image board.

*v* **— returned by the graPHIGS API, fullword integer**
Vertical size of the image board.

**Error Codes**

None

**Related Subroutines**

**GPCRIB**
Create Image Board

**GPFRCT**
Fill Rectangle

**GPRDFB**
Read Frame Buffer

**GPQIBF**
Inquire Image Board Facilities

**GPWRCT**
Write Rectangle

**RCP code**

201345800 (X'0C004B08')

## GPQIBF - Inquire Image Board Facilities

**GPQIBF (*ncid*, *start*, *number*, *errind*, *totnum*, *depth*, *h*, *v*)**

**Note:** This subroutine is a Nucleus Description Table (NDT) inquiry. For an overview, see "NDT Inquiries."

**Purpose**

Use **GPQIBF** to inquire the image board capabilities for the specified nucleus.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*start* **— specified by user, fullword integer**
Starting member in the list of available image board bit depth (>=1).

*number* **— specified by user, fullword integer**
Number of image board bit depth entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

| | |
|---|---|
| **202** | SPECIFIED NUCLEUS DOES NOT EXIST |
| **538** | START VALUE < ONE |
| **539** | REQUESTED NUMBER < ZERO |
| **543** | START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED |

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available image board bit depths.

*depth* **— returned by the graPHIGS API, array of fullword integers**
List of available image board bit depths.

*h* **— returned by the graPHIGS API, fullword integer**
Maximum horizontal size of an image board. (A value of zero means that there is no limit on the image board size.)

*v* **— returned by the graPHIGS API, fullword integer**
Maximum vertical size of an image board. (A value of zero means that there is no limit on the image board size.)

### Error Codes

None

### Related Subroutines

| | |
|---|---|
| **GPCRIB** | Create Image Board |
| **GPFRCT** | Fill Rectangle |
| **GPRDFB** | Read Frame Buffer |
| **GPQIBC** | Inquire Image Board Characteristics |
| **GPWRCT** | Write Rectangle |

### RCP code

201345801 (X'0C004B09')

## GPQICH - Inquire Image Characteristics

**GPQICH (*wsid*, *index*, *errind*, *conn*, *ctid*, *totnum*, *libid*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQICH** to inquire the current image characteristics of the specified image on the specified workstation. The specified image was created by the Define Image (**GPDFI**) subroutine. Its characteristics consist of the image boards and color tables that are grouped together to form the specified image.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

The list of image board identifiers parameter (*libid*) must have a length of at least three fullwords.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Defined image index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**216** ONE OR MORE RESOURCES IS NOT ATTACHED

**290** SPECIFIED IMAGE INDEX IS NOT DEFINED

*conn* **— returned by the graPHIGS API, fullword integer**
Connection type (`-1=FRAME_BUFFER_COMPATIBLE, 1=COMPONENT, 2=INDEXED`).

*ctid* **— returned by the graPHIGS API, fullword integer**
Color table identifier of the color table used to form the specified image.

*totnum* **— returned by the graPHIGS API, fullword integer**
Number of image boards grouped together to form the specified image.

*libid* **— returned by the graPHIGS API, array of fullword integers**
List of image board identifiers that form the specified image.

The application must supply storage that is large enough to contain the maximum number of image board identifiers that the workstation supports. (Currently, three. However, in the future, this number may increase. Therefore, it is recommended that storage be supplied for *at least* 24 image board identifiers.)

### Error Codes

None

### Related Subroutines

**GPCRIB**
Create Image Board

**GPDFI**
Define Image

### RCP code

201346561 (X'0C004E01')

## GPQICS - Inquire Input Character Set

**GPQICS (*wsid*, *class*, *device*, *errind*, *csid*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQICS** to inquire the current input character set for the specified input device on the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*class* **— specified by user, fullword integer**
Device class (1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING).

*device* **— specified by user, fullword integer**
Device number.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

*csid* **— returned by the graPHIGS API, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

**Error Codes**

None

**Related Subroutines**

**GPICS**
Set Input Character Set

**RCP code**

201338887 (X'0C003007')

## GPQID - Inquire Input Device State

**GPQID (*wsid*, *class*, *device*, *errind*, *state*, *deact*, *echosw*, *trigger*, *break*, *reset*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQID** to inquire the current operating mode of the specified logical input device.

The operating mode consists of six individual switches. With the Set Input Device Mode (**GPIDMO**)subroutine, your application can set each of these switches individually to a specified value. This subroutine returns the current value for each of the switches.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*class* **— specified by user, fullword integer**
Input device class (1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING).

*device* **— specified by user, fullword integer**
Logical input device number (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**37** WORKSTATION IS NOT OF CATEGORY OUTIN

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

*state* **— returned by the graPHIGS API, fullword integer**
State switch (1=DEVICE_INACTIVE, 2=DEVICE_ACTIVE).

*deact* **— returned by the graPHIGS API, fullword integer**
Auto deactivate switch (1=OFF, 2=ON).

*echosw* **— returned by the graPHIGS API, fullword integer**
Current echo switch (1=NOECHO, 2=ECHO).

*trigger* **— returned by the graPHIGS API, fullword integer**
Primary trigger switch (1=OFF, 2=ON).

*break* **— returned by the graPHIGS API, fullword integer**
Break switch (1=OFF, 2=ON).

*reset* **— returned by the graPHIGS API, fullword integer**
Auto reset switch (1=OFF, 2=ON).

**Error Codes**

None

**Related Subroutines**

**GPIDMO**
Set Input Device Mode

**RCP code**

201338890 (X'0C00300A')

## GPQIDD - Inquire Input Device Description

**GPQIDD (*wstype*, *class*, *devnum*, *id*, *lidata*, *idata*, *mlodata*, *errind*, *lodata*, *odata*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQIDD** to inquire the information about the specified device.

The group identifier (*id*) parameter identifies the data the graPHIGS API returns. Some data types may require additional information in the input data (*idata*) parameter.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the information you inquired is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*class* **— specified by user, fullword integer**
Input device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*devnum* **— specified by user, fullword integer**
Device number (>=1).

*id* **— specified by user, fullword integer**
Group identifier. The graPHIGS API supports the following identifiers:

**Group identifier 1**
Available pick selection criteria. The device class must be set to a value of `5=PICK`. No input data is required for this identifier.

*lidata* **— specified by user, fullword integer**
Length, in bytes, of the input data area (>=0).

*idata* **— variable length data**
Input data. Depending on the *id* parameter value specified, input data is as follows:

**If Group Identifier=1 (available pick selection criteria)**
No input data is required for this identifier.

*mlodata* **— specified by user, fullword integer**
Length, in bytes, of the output data area (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**272** GROUP IDENTIFIER IS INVALID

**328** INPUT CLASS VALUE IS INVALID

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*lodata* **— returned by the graPHIGS API, fullword integer**
Length, in bytes, of the available output data.

*odata* **— returned by the graPHIGS API, variable length data**
Output data. Depending on the specified value of the *id* parameter, output data is as follows:

**Group Identifier=1 (available pick selection criteria)**
Array of fullword integers.

The available pick selection criteria. The *lodata* parameter specifies the length of the array in bytes (`1=FIRST, 2=LAST, 3=ALL, 4=FIRST_VISIBLE, 5=LAST_VISIBLE, 6=ALL_VISIBLE`).

### Error Codes

None

### Related Subroutines

**GPPKSC**
Set Pick Selection Criteria

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201339677 (X'0C00331D')

## GPQIDF - Inquire Image Definition Facilities

**GPQIDF (*wstype*, *start*, *number*, *errind*, *nimage*, *totnum*, *conn*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQIDF** to inquire the image definition facilities for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available connection types (>=1).

*number* **— specified by user, fullword integer**
Number of connection types entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*nimage* **— returned by the graPHIGS API, fullword integer**
Maximum number of definable images. The maximum image index is *nimage* and all entries can be modified.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available connection types.

*conn* **— returned by the graPHIGS API, array of fullword integers**
List of available connection types (`-1=FRAME_BUFFER_COMPATIBLE, 1=COMPONENT, 2=INDEXED`).

**Error Codes**

None

**Related Subroutines**

**GPDFI**
Define Image

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201346062 (X'0C004C0E')

## GPQIF - Inquire Interior Facilities

**GPQIF (*wstype*, *starti*, *numi*, *starth*, *numh*, *errind*, *intnum*, *interiors*, *hatnum*, *hatch*, *npred*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQIF** to inquire the interior facilities for the specified workstation type.

The graPHIGS API returns data indicating the total number of available interior styles (*intnum*), the number of available hatch styles (*hatnum*), and the total number of indexes predefined in the interior bundle table (*npred*).

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*starti* **— specified by user, fullword integer**
Starting member of the list of interior styles (>=1).

*numi* **— specified by user, fullword integer**
Number of interior styles requested (>=0).

*starth* **— specified by user, fullword integer**
Starting member of the list of hatch styles (>=1).

*numh* **— specified by user, fullword integer**
Number of hatch styles requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*intnum* **— returned by the graPHIGS API, fullword integer**
Total number of available interior styles.

*interiors* **— returned by the graPHIGS API, array of fullword integers**
List of available interior styles (`1=HOLLOW, 2=SOLID, 3=PATTERN, 4=HATCH, 5=EMPTY`). The output array must be large enough to contain the requested data.

*hatnum* **— returned by the graPHIGS API, fullword integer**
Total number of available hatch styles.

*hatch* **— returned by the graPHIGS API, array of fullword integers**
List of available hatch styles. The output array must be large enough to contain the requested data.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined interior bundle table entries.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339657 (X'0C003309')

## GPQIMC - Inquire Image Mapping Characteristics

**GPQIMC (*wsid*, *imid*, *errind*, *view*, *index*, *origin*, *size*, *P*, *Q*, *R*, *method*, *priority*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQIMC** to inquire the current image mapping characteristics of the specified mapped image.

A defined image is displayed by mapping its rectangular part into a parallelogram in World Coordinates (WC). The mapped image is identified by an image mapping identifier.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*imid* **— specified by user, fullword integer**
Image mapping identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**296** SPECIFIED IMAGE MAPPING DOES NOT EXIST

*view* **— returned by the graPHIGS API, fullword integer**
View index.

*index* **— returned by the graPHIGS API, fullword integer**
Image index.

*origin* **— returned by the graPHIGS API, two fullword integers**
Image rectangle origin.

*size* **— returned by the graPHIGS API, two fullword integers**
Image rectangle size.

*P* **— returned by the graPHIGS API, three floating-point numbers (WC)**
Lower left corner of the image mapping.

*Q* **— returned by the graPHIGS API, three floating-point numbers (WC)**
Lower right corner of the image mapping.

*R* **— returned by the graPHIGS API, three floating-point numbers (WC)**
Upper left corner of the image mapping.

*method* **— returned by the graPHIGS API, fullword integer**
Image mapping method (`1=PIXEL_BY_PIXEL`).

*priority* **— returned by the graPHIGS API, short floating-point number**
Priority.

**Error Codes**

None

**Related Subroutines**

**GPCIM2**
Create Image Mapping 2

**GPCIM3**
Create Image Mapping 3

**GPFRCT**
Fill Rectangle

**GPRDFB**
Read Frame Buffer

**GPQIMF**
Inquire Image Mapping Facilities

**GPWRCT**
Write Rectangle

**RCP code**

201346562 (X'0C004E02')

## GPQIMF - Inquire Image Mapping Facilities

**GPQIMF (*wstype*, *start*, *number*, *errind*, *nprio*, *totnum*, *method*)**

> **Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQIMF** to inquire the image mapping facilities for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available image mapping methods (>=1).

*number* **— specified by user, fullword integer**
Number of image mapping methods requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*nprio* **— returned by the graPHIGS API, fullword integer**
Number of image priorities supported. The value zero means that the workstation supports a contiguous range of priorities.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available image mapping methods.

*method* **— returned by the graPHIGS API, array of fullword integers**
List of available image mapping methods (`1=PIXEL_BY_PIXEL`).

**Error Codes**

None

**Related Subroutines**

**GPCIM2**
Create Image Mapping 2

**GPCIM3**
Create Image Mapping 3

**GPFRCT**
Fill Rectangle

**GPRDFB**
Read Frame Buffer

**GPQIMC**
Inquire Image Mapping Characteristics

**GPQRCT**
Inquire Realized Connection Type

**GPWRCT**
Write Rectangle

**RCP code**

# GPQIMI - Inquire Image Mapping of Image

**GPQIMI (*wsid*, *index*, *start*, *number*, *errind*, *totnum*, *limid*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQIMI** to inquire the current image mappings which contain the specified image.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Defined image index (>=1).

*start* **— specified by user, fullword integer**
Starting member of the list of image mappings (>=1).

*number* **— specified by user, fullword integer**
Number of image mappings requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**290** SPECIFIED IMAGE INDEX IS NOT DEFINED

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of image mappings of the image.

*limid* **— returned by the graPHIGS API, array of fullword integers**
List of image mapping identifiers of the image.

### Error Codes

None

### Related Subroutines

**GPCIM2**
Create Image Mapping 2

**GPCIM3**
Create Image Mapping 3

**GPCRIB**
Create Image Board

**GPDFI**
Define Image

**RCP code**

201346563 (X'0C004E03')

## GPQIMV - Inquire Image Mapping on View

**GPQIMV (*wsid*, *view*, *start*, *number*, *errind*, *totnum*, *limid*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQIMV** to inquire the current image mappings on the specified view.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*start* **— specified by user, fullword integer**
Starting member of the list of defined image mappings (>=1).

*number* **— specified by user, fullword integer**
Number of image mappings requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of image mappings on the view.

*limid* **— returned by the graPHIGS API, array of fullword integers**
List of image mapping identifiers on the view.

**Error Codes**

None

**Related Subroutines**

**GPCIM2**
Create Image Mapping 2

**GPCIM3**
Create Image Mapping 3

**GPCRIB**
Create Image Board

**GPDFI**
Define Image

**RCP code**

201346564 (X'0C004E04')

## GPQIMW - Inquire Image Mapping on Workstation

**GPQIMW (*wsid*, *start*, *number*, *errind*, *totnum*, *limid*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQIMW** to inquire the current image mappings for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of defined image mappings (>=1).

*number* **— specified by user, fullword integer**
Number of image mappings requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of image mappings on the workstation.

*limid* **— returned by the graPHIGS API, array of fullword integers**
List of image mapping identifiers on the workstation.

**Error Codes**

None

**Related Subroutines**

**GPCIM2**
Create Image Mapping 2

**GPCIM3**
Create Image Mapping 3

**GPCRIB**
Create Image Board

**GPDFI**
Define Image

**RCP code**

201346565 (X'0C004E05')

## GPQIQO - Inquire Input Queue Overflow

**GPQIQO (*errind*, *major*, *class*, *minor*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQIQO** to inquire identification of the event report causing the event queue overflow.

Once the event queue overflow occurs, no more events are added to the event queue until the overflow situation is cleared. The overflow situation is cleared when the event queue becomes empty. The application can make the event queue empty by using the Await Event (**GPAWEV**) subroutine, Flush Device Events (**GPFLEV**) subroutine, or the Flush Workstation Event (**GPFWEV**) subroutine.

If the event queue has overflowed since you last called the Open graPHIGS (**GPOPPH**) subroutine, the event class, major and minor codes of the event causing the overflow are returned (if the event class has corresponding major and minor codes).

The details of the possible event classes and meanings of their major and minor codes are shown in *The graPHIGS Programming Interface: Technical Reference*.

The event queue overflow is not reported to the application when the overflow occurs. It is reported on the next invocation of the following subroutines which may change the contents of the event queue.

- Await Event (**GPAWEV**) subroutine
- Flush Device Events (**GPFLEV**) subroutine
- Flush Workstation Events (**GPFWEV**)subroutine
- Detach Resource (**GPDTR**) subroutine (for a workstation)
- Close Workstation (**GPCLWS**) subroutine
- Disconnect Nucleus (**GPDNC**) subroutine(owning at least one workstation)

The event queue overflow is reported to the application only once per event queue overflow situation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**148** EVENT QUEUE HAS NOT OVERFLOWED

*major* **— returned by the graPHIGS API, fullword integer**
Major event code.

*class* **— returned by the graPHIGS API, fullword integer**
Event class.

*minor* **— returned by the graPHIGS API, fullword integer**
Minor event code.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201336328 (X'0C002608')

## GPQISF - Inquire Input Character Set Facilities

**GPQISF (*wstype*, *class*, *device*, *start*, *number*, *errind*, *ncsid*, *csid*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQISF** to inquire the character set identifiers that are supported for the specified input device on the specified workstation type.

The graPHIGS API returns data indicating the total number of available character sets and their character set identifiers.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*class* **— specified by user, fullword integer**
Device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*device* **— specified by user, fullword integer**
Device number.

*start* **— specified by user, fullword integer**
Starting member of the list of character set identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of character set identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*ncsid* **— returned by the graPHIGS API, fullword integer**
Total number of character set identifiers supported.

*csid* **— returned by the graPHIGS API, array of fullword integers**
List of supported character set identifiers.

See Appendix A. "Character Set and Font Identifiers" for more information.

The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339667 (X'0C003313')

## GPQISN - Inquire Identifiers of Structures in Network

**GPQISN (*strid*, *start*, *number*, *errind*, *totnum*, *istrid*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

**Purpose**

Use **GPQISN** to inquire a list of the structure identifiers in the specified structure network.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*strid* **— specified by user, fullword integer**
Structure identifier of the root structure.

*start* **— specified by user, fullword integer**
The starting member of the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of structures in network.

*istrid* **— returned by the graPHIGS API, array of fullword integers**
List of conflicting structure identifiers. In a complete list of structure identifiers in the network, the first structure identifier entry is always the root structure. No structure identifiers are duplicated in the list (e.g., if a structure is referenced in the network more than once, then it only appears once in the list).

**Error Codes**

None

**Related Subroutines**

**GPQSTE**
Inquire Structure Existence

**GPQSTI**
Inquire Structure Identifiers

**RCP code**

201347589 (X'0C005205')

## GPQIT - Inquire Input Trigger Capabilities

**GPQIT (*wstype*, *class*, *devnum*, *start*, *number*, *errind*, *ntrigs*, *ltrigs*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQIT** to inquire the input device trigger capabilities of a specified device for a specified workstation type. If the triggers are programmable, the graPHIGS API returns a list of available triggers. The returned list corresponds to the available triggers for all trigger levels of the specified input device.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation identifier.

*class* **— specified by user, fullword integer**
Input device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*devnum* **— specified by user, fullword integer**
Input device number (>=1).

*start* **— specified by user, fullword integer**
Starting member in the list of available trigger types (>=1).

*number* **— specified by user, fullword integer**
Number of triggers requested from the list (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

**569** DEVICE DOES NOT SUPPORT PROGRAMMABLE TRIGGERS

*ntrigs* **— returned by the graPHIGS API, fullword integer**
Total number of entries in the list of available triggers.

*ltrigs* **— returned by the graPHIGS API, array of fullword integers**
List of trigger descriptor triplets. The list is an array of trigger descriptors in which each descriptor consists of three fullword integers designating the trigger type, low trigger qualifier, and high trigger qualifier. The trigger type field has the following meanings:

| Type | Meaning |
|---|---|
| **>0** | Identifier of physical device within the button category. The trigger qualifiers for this trigger type are a range of choice numbers generated by the physical device. |
| **-1** | Change of the measure of the logical input device. Ignore the trigger qualifier fields. |
| **-2** | The secondary trigger fires when the primary trigger fires. This type is valid only for secondary (>0) trigger list identifier. Ignore the trigger qualifier fields. |

The parameter *ntrigs* identifies the total number of triplets in the available trigger list. The actual number returned will depend on the setting of the *start* and *number* parameters. **Error Codes**

None

**Related Subroutines**

**GPIT** Set Input Device Trigger

**GPQDIT**
Inquire Default Input Device Triggers

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339400 (X'0C003208')

# GPQITS - Inquire Input Device Trigger State

**GPQITS (*wsid*, *class*, *devnum*, *listid*, *start*, *number*, *errind*, *ntrigs*, *ltrigs*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQITS** to inquire the current trigger list for a specified level of a particular device on a specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*class* **— specified by user, fullword integer**
Input device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*devnum* **— specified by user, fullword integer**
Input device number (>=1).

*listid* **— specified by user, fullword integer**
Trigger list identifier for return of trigger list (>=0).

Trigger list identifier zero is always present and is called the primary trigger. The primary trigger causes the input to be returned to the application.

Secondary triggers may have different intermediate functions used in the processing of the input. They are identified with trigger list identifiers beginning with 1.

*start* **— specified by user, fullword integer**
Starting member in the list of current triggers (>=1).

*number* **— specified by user, fullword integer**
Number of triggers requested from the list (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*ntrigs* **— returned by the graPHIGS API, fullword integer**
Number of triggers in the current trigger list.

*ltrigs* **— returned by the graPHIGS API, array of fullword integers**
List of trigger descriptor triplets. The list is an array of trigger descriptors in which each descriptor consists of 3 fullword integers designating the trigger type, low trigger qualifier, and high trigger qualifier. The trigger type field has the following meanings:

| Type | Meaning |
|---|---|
| `-1` | Change of the measure of the corresponding physical input device. The low qualifier specifies the granularity of movement which causes the trigger to fire. The granularity is specified as the amount that the physical device measure must change since the last trigger was fired in order for the trigger to be fired again. The high qualifier will be zero. |
| `0` | An implementation dependent trigger that is only valid as the default value. |
| `>0` | Physical device number within the button category. The trigger qualifiers for this trigger type are a range of choices on the indicated physical device. |

The parameter *ntrigs* identifies the total number of triplets in the available trigger list. The actual number returned will depend on the setting of the *start* and *number* parameters.

### Error Codes

None

### Related Subroutines

**GPIT** Set Input Device Trigger

**GPQDIT**
Inquire Default Input Device Triggers

### RCP code

201338889 (X'0C003009')

## GPQIVF - Inquire Invisibility Filter

**GPQIVF (*wsid*, *inlen*, *exlen*, *errind*, *inclen*, *incl*, *exclen*, *excl*)**

> **Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQIVF** to inquire the current invisibility filter values for the visibility inclusion and exclusion filters on the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired

information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*inlen* **— specified by user, fullword integer**
Length of inclusion filter array provided by the application for the graPHIGS API to return the corresponding filters (>=0).

This is the size of the array provided by the application for the graPHIGS API to return the corresponding data.

*exlen* **— specified by user, fullword integer**
Length of exclusion filter array provided by the graPHIGS API to return the corresponding filter (>=0).

This is the size of the array provided by the application for the graPHIGS API to return the corresponding data.

*errind* **— returned by the graPHIGS API, fullword integer**
If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**505** LENGTH OF RETURN ARRAY < ZERO

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

*inclen* **— returned by the graPHIGS API, fullword integer**
Number of class names in the inclusion filter.

*incl* **— returned by the graPHIGS API, array of fullword integers.**
List of class names in the inclusion filter.

*exclen* **— returned by the graPHIGS API, fullword integer**
Number of class names in the exclusion filter length.

*excl* **— returned by the graPHIGS API, array of fullword integers.**
List of class names in the exclusion filter.

**Error Codes**

None

**Related Subroutines**

**GPIVF** Set Invisibility Filter

**GPQNCN**
Inquire Number of Available Class Names

**RCP code**

201336838 (X'0C002806')

# GPQIW - Inquire List of Images on the Workstation

**GPQIW (*wsid*, *start*, *number*, *errind*, *totnum*, *lindex*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQIW** to inquire the currently defined images on the specified workstation. Images were defined on the workstation by the Define Image (**GPDFI**) subroutine.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of indexes of defined images (>=1).

*number* **— specified by user, fullword integer**
Number of image indexes requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of defined images on the workstation.

*lindex* **— returned by the graPHIGS API, array of fullword integers**
List of image indexes for the defined images.

### Error Codes

None

### Related Subroutines

**GPCAI**
Cancel Image

**GPDFI**
Define Image

**RCP code**

201346566 (X'0C004E06')

# GPQLC - Inquire Locator Device State

**GPQLC (*wsid*, *device*, *type*, *length*, *errind*, *mode*, *echosw*, *view*, *pos*, *echo*, *area*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQLC** to inquire the current state of a locator device attached to a specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Locator device number.

*type* **— specified by user, fullword integer**
Type of returned values (`1=SET`).

*length* **— specified by user, fullword integer**
Length of array, in bytes, provided by the application for the graPHIGS API to return the locator data record array.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**534** TYPE VALUE IS INVALID

*mode* **— returned by the graPHIGS API, fullword integer**
Current operating mode (`1=REQUEST, 2=SAMPLE, 3=EVENT, 4=APPLICATION_DEFINED`). The graPHIGS API only returns a mode of `4=APPLICATION_DEFINED` if the application set the device mode using the Set Input Device State (**GPIDMO**) subroutine and the mode does not emulate Request, Sample or Event mode.

*echosw* **— returned by the graPHIGS API, fullword integer**
Current echo switch (`1=NOECHO, 2=ECHO`).

*view* — **returned by the graPHIGS API, fullword integer**
Current initial view index.

*pos* — **returned by the graPHIGS API, 3 short floating-point numbers (WC)**
Current initial locator position.

*echo* — **returned by the graPHIGS API, fullword integer**
Current prompt/echo type.

*area* — **returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Current echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*datalen* — **returned by the graPHIGS API, fullword integer**
Current locator data record length.

*data* — **returned by the graPHIGS API, variable length data**
Current locator data record.

**Error Codes**

None

**Related Subroutines**

**GPIDMO**
Set Input Device Mode

**GPINLC**
Initialize Locator

**GPQDLC**
Inquire Default Locator Device Data

**RCP code**

201338882 (X'0C003002')

## GPQLCF - Inquire List of Color Facilities

**GPQLCF (*wstype*, *number*, *ids*, *errind*, *data*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQLCF** to inquire one or more groups describing the color facilities for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* — **specified by user, 8 byte character string**
Workstation type.

*number* — **specified by user, fullword integer**
Number of groups requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*data* **— returned by the graPHIGS API, array of fullword quantities**
Data array. When you specify a list of group identifiers, the data is returned to your application in the order you specified your request. Below is a list of the contents of each group:

**Group Identifier 1 - Color model of the workstation**
A fullword integer (`1=RGB, 2=HSV, 3=CMY, 4=CIELUV`).

**Group Identifier 2 - Color available on the workstation**
A fullword integer (`1=MONOCHROME, 2=COLOR`).

**Group Identifier 3 - Number of available colors (total color palette size)**
A fullword integer indicating the number of available colors on the workstation.

**Group Identifier 4 - Number of predefined default color table entries**
A fullword integer indicating the number of predefined default color table entries.

**Group Identifier 5 - Number of definable color processing mode table entries**
A fullword integer indicating the number of definable color processing mode table entries.

**Group Identifier 6 - Number of predefined color processing mode table entries**
A fullword integer indicating the number of predefined color processing mode table entries.

**Group Identifier 7 - Order of color components for color quantization**
A fullword integer (`1=RGB, 2=BGR`).

**Group Identifier 8 - `CIELUV` color components**
An array of nine floating-point numbers which are the color components for the three monitor primaries (`1=RGB`). For each monitor primary, graPHIGS API returns the `CIELUV` chromaticity coefficients ($u'$, $v'$) and the luminance value $Y$.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201339672 (X'0C003318')

# GPQLI - Inquire List of Logical Input Devices

**GPQLI (*wstype*, *class*, *start*, *number*, *errind*, *ndev*, *dev*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQLI** to inquire the number of available logical input devices for the specified device class for the specified workstation type.

The graPHIGS API returns data indicating the total number of logical input devices and their device numbers for the specified class.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*class* **— specified by user, fullword integer**
Device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*start* **— specified by user, fullword integer**
Starting member of the list of input devices (>=1).

*number* **— specified by user, fullword integer**
Number of input device numbers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**328** INPUT CLASS VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*ndev* **— returned by the graPHIGS API, fullword integer**
Total number of logical device numbers in the device class.

*dev* **— returned by the graPHIGS API, array of fullword integers.**
List of device numbers in the device class. The output array must be large enough for the requested data.

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201339662 (X'0C00330E')

## GPQLNR - Inquire List of Line Rendering Styles

**GPQLNR (*wstype*, *start*, *number*, *errind*, *totnum*, *rstyle*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQLNR** to inquire the list of line rendering styles for a specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of line rendering styles (>=1).

*number* **— specified by user, fullword integer**
Number of line rendering styles requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available line rendering styles supported by a workstation.

*rstyle* **— returned by the graPHIGS API, array of fullword integers**
List of line rendering styles (1=WORKSTATION_DEPENDENT_RENDERING, 2=SCALED_TO_FIT_RENDERING).

**Error Codes**

None

**Related Subroutines**

**GPLNR**
Set Linetype Rendering

**GPLT** Set Linetype

**GPLTR**
Set Linetype Representation

**RCP code**

201339412 (X'0C003214')

## GPQLSF - Inquire Light Source Facilities

**GPQLSF (*wstype*, *start*, *number*, *errind*, *maxe*, *totnum*, *ltype*, *maxa*, *npred*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQLSF** to inquire the light source facilities for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available light source types (>=1).

*number* **— specified by user, fullword integer**
Number of light source type entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*maxe* **— returned by the graPHIGS API, fullword integer**
Maximum number of light source table entries.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available light source types supported on the workstation.

*ltype* **— returned by the graPHIGS API, array of fullword integers**
List of light source types (`1=AMBIENT, 2=DIRECTIONAL, 3=POSITIONAL, 4=SPOT`).

*maxa* **— returned by the graPHIGS API, fullword integer**
Maximum number of simultaneously active non-ambient light sources.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined light source indexes.

**Error Codes**

None

**Related Subroutines**

**GPLSR**
Set Light Source Representation

**GPQLSR**
Inquire Light Source Representation

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201346052 (X'0C004C04')

## GPQLSR - Inquire Light Source Representation

**GPQLSR (*wsid*, *index*, *type*, *errind*, *lstype*, *color*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQLSR** to inquire the current attribute values in the specified entry in the light source table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

The data parameter (*data*) must be large enough to include the maximum number of light source parameters.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Light source table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

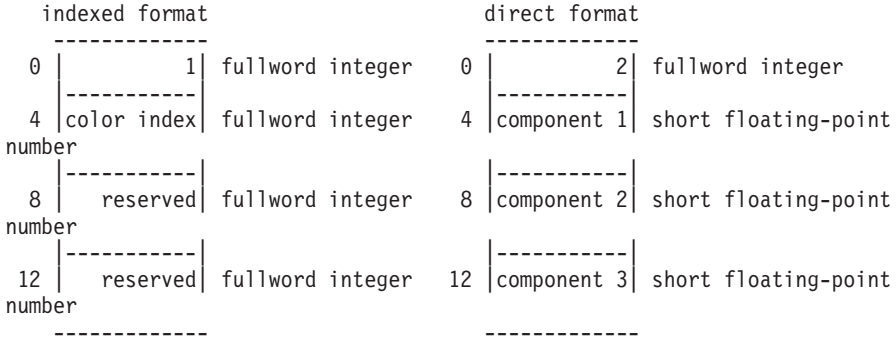
**254** LIGHT SOURCE INDEX < ONE

**255** LIGHT SOURCE INDEX EXCEEDS THE WORKSTATION TABLE CAPACITY

**534** TYPE VALUE IS INVALID

*lstype* **— returned by the graPHIGS API, fullword integer**
Light source type (1=AMBIENT, 2=DIRECTIONAL, 3=POSITIONAL, 4=SPOT).

*color* **— returned by the graPHIGS API, four fullwords of data**
Light source color. This parameter includes one of the following two formats:




*data* **— returned by the graPHIGS API, array of short floating-point numbers**
List of light source parameters. Values returned to this parameter depend on the light source type. The application must supply storage for this parameter that is large enough to contain the maximum data listed below:

**1=AMBIENT**
None

**2=DIRECTIONAL**
Light source direction 3 short floating-point numbers (WC)

**3=POSITIONAL**
Light source position -3 short floating-point numbers (WC)

Attenuation coefficients - 2 short floating-point numbers

**4=SPOT** Light source position - 3 short floating-point numbers (WC). Light source direction - 3 short floating-point numbers (WC). Concentration exponent - short floating-point number. Attenuation coefficients - 2 short floating-point numbers. Spread angle - short floating-point number.

### Error Codes

None

### Related Subroutines

**GPLSR**
Set Light Source Representation

### RCP code

201339150 (X'0C00310E')

## GPQLTF - Inquire Linetype Facilities

**GPQLTF (*wstype*, *errind*, *sections*, *maxlen*, *unit*, *npred*, *available*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQLTF** to inquire the line pattern facilities for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the output parameter *available* is set to a value of one, then the values returned in the other output parameters are unpredictable. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*sections* **— returned by the graPHIGS API, fullword integer**
Maximum number of sections that a line pattern can have.

*maxlen* **— returned by the graPHIGS API, fullword integer**
Maximum length of a line pattern. The sum of the lengths of all sections in a line pattern must be less than this value.

*unit* **— returned by the graPHIGS API, short floating-point number (DC)**
Size of the line pattern unit. All line pattern definitions are specified as multiples of this value.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined entries of the line pattern table.

*available* **— returned by the graPHIGS API, fullword integer**
Availability of the Set Linetype Representation (**GPLTR**) subroutine and Inquire Linetype Representation(**GPQLTR**) subroutine (`1=NOT_AVAILABLE, 2=BOTH_AVAILABLE, 3=INQUIRE_ONLY_AVAILABLE, 4=SET_ONLY_AVAILABLE`).

**Error Codes**

None

**Related Subroutines**

**GPLT** Set Linetype

**GPLTR**
Set Linetype Representation

**GPQLTR**
Inquire Linetype Representation

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201346049 (X'0C004C01')

# GPQLTR - Inquire Linetype Representation

**GPQLTR (*wsid*, *ltype*, *errind*, *number*, *pattern*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQLTR** to inquire the current line pattern in the specified entry in the line type table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ltype* **— specified by user, fullword integer**
Line type table index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**63** LINETYPE VALUE < ONE

**64** SPECIFIED LINETYPE NOT AVAILABLE ON WORKSTATION

**274** THIS FUNCTION IS NOT SUPPORTED BY THE WORKSTATION

*number* **— returned by the graPHIGS API, fullword integer**
Number of sections in the line pattern.

*pattern* **— returned by the graPHIGS API, array of fullword integers**
List of the length of each section in the line pattern. The entries of the array alternate between `SOLID` and `VOID` sections with the first entry being `SOLID`. Each length is specified in terms of a multiple of the minimum size section for the workstation. The application must supply storage for this parameter that is large enough to contain the maximum number of sections that this workstation supports.

### Error Codes

None

### Related Subroutines

**GPLTR**
Set Linetype Representation

**GPQLTF**
Inquire Linetype Facilities

### RCP code

201339151 (X'0C00310F')

## GPQLW - Inquire Length of Workstation State Tables

**GPQLW (*wstype*, *errind*, *ltable*, *mtable*, *ttable*, *itable*, *etable*, *pttable*, *ctable*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQLW** to inquire the maximum number of entries supported for workstation tables for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the output parameter *available* is set to a value of one, then the values returned in the other output parameters are unpredictable. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*ltable* **— returned by the graPHIGS API, fullword integer**
Maximum number of polyline bundle table entries.

*mtable* **— returned by the graPHIGS API, fullword integer**
Maximum number of polymarker bundle table entries.

*ttable* **— returned by the graPHIGS API, fullword integer**
Maximum number of text bundle table entries.

*itable* **— returned by the graPHIGS API, fullword integer**
Maximum number of interior bundle table entries.

*etable* **— returned by the graPHIGS API, fullword integer**
Maximum number of edge bundle table entries.

*pttable* **— returned by the graPHIGS API, fullword integer**
Maximum number of pattern indexes.

*ctable* **— returned by the graPHIGS API, fullword integer**
Maximum number of default color table entries.

**Error Codes**

None

**Related Subroutines**

**GPPAR**
Set Pattern Representation

**GPQRCT**
Inquire Realized Connection Type

**GPXCR**
Set Extended Color Representation

**GPXER**
Set Extended Edge Representation

**GPXIR**
Set Extended Interior Representation

**GPXPLR**
Set Extended Polyline Representation

**GPXPMR**
Set Extended Polymarker Representation

**GPXTXR**
Set Extended Text Representation

**RCP code**

# GPQMDS - Inquire Mapped Display Surface Size

**GPQMDS (*wsid*, *errind*, *units*, *csize*, *asize*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQMDS** to inquire the size of the mapped display surface on the specified workstation. The mapped display surface is the subarea of the window that the workstation uses as the workstation's display surface for graphical output and input. Only workstations which use the facilities of a window system (e.g., X-Windows) support this inquire.

The mapped display surface size may change if the user changes the size of the window that contains the mapped display surface. By enabling the Window Resize Notification function of the Escape (**GPES**) subroutine, your application can receive notification of such size changes. Then issue **GPQMDS**) to obtain the new size of the mapped display surface. In addition, your application can control the aspect ratio of the mapped display surface by using the Window Aspect Ratio (`XWINDASP`) procopt.

If your application uses the **GPDCMM** subroutine and sets the method to `2=DIRECT`, then the graPHIGS API returns the current size of the window, constrained to the same area as the root window. If the method is set to `1=MAPPED`, then the graPHIGS API returns the size of the area that is used to display the device coordinate range, constrained to an area with the same aspect ratio as the root window, centered in the window.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

This subroutine is assigned escape identifier 1010.

**Note:** This subroutine is an escape subroutine, and therefore, may not be available on all workstations. Use the Inquire List of Available Escape Subroutines (**GPQES**) subroutine to determine if this subroutine is supported by a specific workstation.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**527** ESCAPE FUNCTION NOT AVAILABLE

*units* **— returned by the graPHIGS API, fullword integer**
Mapped display surface coordinate units (`1=METERS`, `2=OTHER`).

*csize* **— returned by the graPHIGS API, 3 short floating-point numbers**
Mapped display size surface in Device Coordinate (DC) units.

*asize* **— returned by the graPHIGS API, 3 fullword integers**
Mapped display surface size in address units.

**Error Codes**

None

**Related Subroutines**

**GPES** Escape

**RCP code**

201336853 (X'0C002815')

## GPQMTF - Inquire Marker Type Facilities

**GPQMTF (*wstype*, *errind*, *format*, *maxlen*, *npred*, *available*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQMTF** to inquire the marker pattern facilities for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the output parameter *available* is set to a value of one, then the values returned in the other output parameters are unpredictable. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer.**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*format* **— returned by the graPHIGS API, fullword integer**
Marker definition format (`1=VECTOR`).

*maxlen* **— returned by the graPHIGS API, fullword integer**
Maximum length of marker definition data.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined marker pattern.

*available* **— returned by the graPHIGS API, fullword integer**
Availability of the Set Marker Type Representation (**GPMTR**) subroutine and Inquire Marker Type Representation (**GPQMTR**) subroutine: (1=NOT_AVAILABLE, 2=BOTH_AVAILABLE, 3=INQUIRE_ONLY_AVAILABLE, 4=SET_ONLY_AVAILABLE).

### Error Codes

None

### Related Subroutines

**GPMTR**
Set Marker Type Representation

**GPQMTR**
Inquire Marker Type Representation

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201346050 (X'0C004C02')

## GPQMTR - Inquire Marker Type Representation

**GPQMTR (*wsid*, *mtype*, *errind*, *format*, *length*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQMTR** to inquire the current marker pattern in the specified entry in the marker type table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*mtype* **— specified by user, fullword integer**
Marker type table index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**69** MARKER TYPE VALUE < ONE

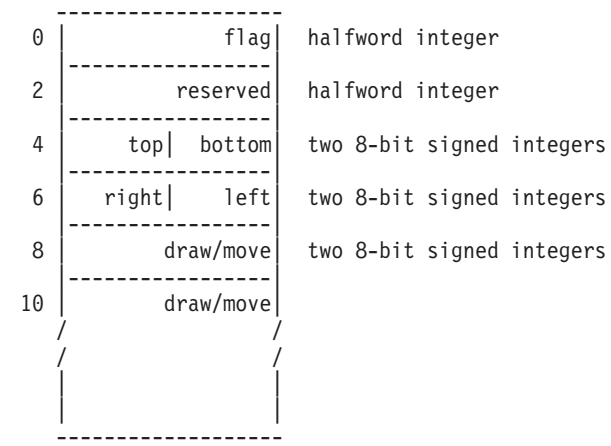
**70** SPECIFIED MARKER TYPE NOT AVAILABLE ON WORKSTATION

*format* **— returned by the graPHIGS API, fullword integer**
Marker pattern format (1=VECTOR).

*length* **— returned by the graPHIGS API, fullword integer**
Length of the entire marker pattern definition.

*data* **— returned by the graPHIGS API, data array**
Marker pattern definition data. Values returned to this parameter depend on the marker pattern format. The application must supply storage for this parameter that is large enough to contain the maximum data that the specified workstation supports.

**Format 1=graPHIGS VECTOR FONT**



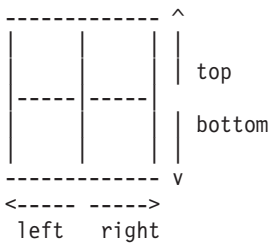

Each field specifies:

**flag**


If the fill flag=1, and the workstation supports filled characters, then the marker will be filled.

**reserved**
Must be 0.

**top, bottom, right, left**
Four 8-bit signed integers specifying the marker rectangle measured from the origin (top > bottom, right > left).




**draw/move**
A pair of 8-bit strings specifying a relative draw/move starting from the origin (the marker position) with the format *sxxxxxx*1 and *syyyyyyb*, where *s* is sign bit and *b*

is blank bit. When the blank bit=1, then the *sxxxxxx* and *syyyyyy* strings specify a relative move; otherwise, they specify a relative draw.

### Error Codes

None

### Related Subroutines

**GPMTR**
Set Marker Type Representation

**GPQMTF**
Inquire Marker Type Facilities

**GPQXTX**
Inquire Extended Text Facilities

### RCP code

201339152 (X'0C003110')

## GPQNCC - Inquire Nucleus Connection State

**GPQNCC (*ncid*, *state*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQNCC** to inquire the application process's connection state to the specified nucleus. The returned state will indicate:

**1=ACTIVE**
The application process is connected to the specified nucleus and the communication path is active.

**2=INACTIVE**
The application process is connected to the nucleus but the communication path is no longer active. The application process should issue a Disconnect from Nucleus (**GPDNC**) subroutine, to clean up and close the connection.

**3=NON_EXISTENT**
The specified nucleus does not exist. Therefore, the application process is not connected to the specified nucleus at this time.

### Parameters

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*status* **— returned by the graPHIGS API, fullword integer**
Status of the application's connection state to the nucleus (`1=ACTIVE`, `2=INACTIVE`, `3=NON_EXISTENT`).

### Error Codes

None

### Related Subroutines

**GPCNC**
Connect Nucleus

**GPDNC**
Disconnect Nucleus

### RCP code

201345806 (X'0C004B0E')

## GPQNCE - Inquire Nucleus Environment

**GPQNCE (*ncid*, *length*, *datatype*, *errind*, *hardware*, *datalen*, *data*)**

**Note:** This subroutine is a Nucleus Description Table (NDT) inquiry. For an overview, see "NDT Inquiries."

### Purpose

Use **GPQNCE** to determine the environment information for the specified nucleus.

The graPHIGS API uses the *datatype* parameter to determine the type of environment information the graPHIGS API returns. If you specify a type of `1=SYSTEM_LEVEL`, then the graPHIGS API returns the hardware and operating system environment information. If you specify a type of `2=ENVIRONMENT_DESCRIPTOR`, then the graPHIGS API returns an environment descriptor, which is an indicator for the character encoding, floating-point format, and byte order specification for the specified nucleus. Use this descriptor as input to the Convert Data (**GPCVD**) subroutine.

### Parameters

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*length* **— specified by user, fullword integer**
Length, in bytes, of the data area specified by *data* into which the graPHIGS API returns the environment information.

*datatype* **— specified by user, fullword integer**
Type of environment information to be returned (`1=SYSTEM_LEVEL, 2=ENVIRONMENT_DESCRIPTOR`).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. Zero indicates that the request has completed successfully.

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

**534** TYPE VALUE IS INVALID

*hardware* **— returned by the graPHIGS API, fullword integer**
Hardware type on which the nucleus is executing (`1=RISC_6000, 2=IBM_6095, 3=IBM_370`).

*datalen* **— returned by the graPHIGS API, fullword integer**
Length of data returned in the data area specified by the *data* parameter.

*data* **— returned by the graPHIGS API, variable length data**
Operating system specific data. The value of each field is expressed in the data format listed below:

**Type 1**

SYSTEM_LEVEL

- Four fullwords of data defined as follows:

```
    ----------------------
  0 |       opsys        | fullword integer
    |--------------------|
  4 |      version       | fullword integer
    |--------------------|
  8 |      release       | fullword integer
    |--------------------|
 12 |nucleus identifier  | fullword integer
    ----------------------
```

  where:
  - *opsys* is the operating system type (1=AIX®_RISC_6000, 2=6095).
  - *version* is the operating system version level.
  - *release* is the operating system release level.
  - *nucleus identifier* is the identifier of the nucleus receiving the inquiry request.

**Type 2**

ENVIRONMENT_DESCRIPTOR

- Four bytes of data defined as follows:

```
    ----------------------
  0 |   env_desc         | 4-byte character string
    ----------------------
```

  where:
  - *env_desc* is the environment descriptor for the specified nucleus.

### Error Codes

None

### Related Subroutines

**GPCVD**
Convert Data

**GPEXAP**
Execute Application Process

**GPINAP**
Initiate Application Process

**GPTMAP**
Terminate Application Process

**GPWDO**
Workstation-Dependent Output

### RCP code

201345807 (X'0C004B0F')

# GPQNCN - Inquire Number of Available Class Names

**GPQNCN (*wstype*, *errind*, *number*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQNCN** to inquire the total number of class names that are available for the specified workstation type.

The graPHIGS API returns data indicating the maximum number of class names that are available for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*number* **— returned by the graPHIGS API, fullword integer**
Number of available class names.

### Error Codes

None

### Related Subroutines

**GPADCN**
Add Class Name to Set

**GPHLF**
Set Highlighting Filter

**GPIVF** Set Invisibility Filter

**GPPKF**
Set Pick Filter

**GPQRCT**
Inquire Realized Connection Type

**GPRCN**
Remove Class Name from Set

**RCP code**

201339396 (X'0C003204')

# GPQNCR - Inquire Nucleus Resource Identifier

**GPQNCR (*type*, *id*, *errind*, *ncid*, *rid*)**

> **Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQNCR** to inquire the nucleus resource identifier of the specified resource. It is the resource identifier assigned by the nucleus when the resource was created.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*type* **— specified by user, fullword integer**
Resource type (`1=WORKSTATION, 2=STRUCTURE_STORE, 3=IMAGE_BOARD, 4=FONT_DIRECTORY, 5=ARCHIVE_FILE`).

*id* **— specified by user, fullword integer**
Identifier of the resource as known to the shell.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**211** RESOURCE TYPE IS INVALID

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**222** SPECIFIED STRUCTURE STORE DOES NOT EXIST

**232** SPECIFIED IMAGE BOARD DOES NOT EXIST

**242** SPECIFIED FONT DIRECTORY DOES NOT EXIST

*ncid* **— returned by the graPHIGS API, fullword integer**
Nucleus identifier.

*rid* **— returned by the graPHIGS API, fullword integer**
Nucleus resource identifier.

**Error Codes**

None

**Related Subroutines**

**GPATR**
Attach Resource

**GPSBMS**
Send Broadcast Message

**GPSPMS**
Send Private Message

**RCP code**

201345796 (X'0C004B04')

## GPQNCS - Inquire Available Nucleus Storage Size

**GPQNCS (*ncid*, *errind*, *size*)**

**Note:** This subroutine is a Nucleus State List (NSL) inquiry. For an overview, see "NSL Inquiries."

**Purpose**

Use **GPQNCS** to determine the amount of free storage available to the nucleus for resource allocation and modification. The returned value may only be approximate since the memory allocation mechanisms may be different in each environment. Requests from other applications connected to this nucleus will compete for this storage.

A value of zero indicates there is unlimited storage available.

**Parameters**

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**571** INQUIRED INFORMATION IS NOT AVAILABLE

*size* **— returned by the graPHIGS API, fullword integer**
The number of bytes of free storage currently available to the nucleus.

**Error Codes**

None

**Related Subroutines**

**GPSSTH**
Set Structure Store Threshold

**RCP code**

201345805 (X'0C004B0D')

# GPQNS - Inquire Nucleus Specification

**GPQNS (*ncid*, *ilen*, *errind*, *conn*, *olen*, *spec*)**

**Note:** This subroutine is a Nucleus Description Table (NDT) inquiry. For an overview, see "NDT Inquiries."

### Purpose

Use **GPQNS** to inquire the nucleus connection method and specification that was used to connect the application process to the specified nucleus.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 536 (the actual connection specification is greater than the length of the area provided), then only the actual length (*olen*) of the connection specification is returned. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*ilen* **— specified by user, fullword integer**
Length of the area provided to contain the nucleus connection specification.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**536** INQUIRY DATA EXCEEDS AREA. LENGTH OF REQUIRED AREA RETURNED

*conn* **— returned by the graPHIGS API, fullword integer**
Nucleus connection method.

*olen* **— returned by the graPHIGS API, fullword integer**
Actual length of the nucleus connection specification.

*spec* **— returned by the graPHIGS API, variable data**
Nucleus connection specification. For a description of the format of the nucleus specification, see the Connect Nucleus (**GPCNC**) subroutine.

### Error Codes

None

### Related Subroutines

**GPCNC**
Connect Nucleus

### RCP code

201345808 (X'0C004B10')

# GPQNSP - Inquire Number of Structure Priorities Supported

**GPQNSP (*wstype*, *errind*, *npri*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQNSP** to inquire the number of the structure priorities supported for the specified workstation type.

When associating a structure to a view, the application specifies a structure priority, which is a real number between 0.0 and 1.0. The structures in each view are traversed in order, from lowest to highest priority.

The graPHIGS API returns values indicating the total number of supported structure priorities. For example, if a workstation uses a 4-bit mask to keep track of priorities, then it is able to support only 16 different priorities and must map the real number to one of 16 values. If a workstation can support a continuous range of structure priorities, then the inquiry returns a value of zero.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*npri* **— returned by the graPHIGS API, fullword integer**
Number of structure priorities supported. The value zero indicates that the workstation supports a continuous range of structure priorities.

### Error Codes

None

### Related Subroutines

**GPARV**
Associate Root with View

**GPQRCT**
Inquire Realized Connection Type

### RCP code

# GPQNST - Inquire Number of Secondary Triggers

**GPQNST (*wstype*, *class*, *devnum*, *errind*, *number*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQNST** to inquire the number of secondary triggers for a specified device for the specified workstation type.

A zero is returned if the specified device has only a primary trigger.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*class* **— specified by user, fullword integer**
Input device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*devnum* **— specified by user, fullword integer**
Input device number (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*number* **— returned by the graPHIGS API, fullword integer**
Number of secondary triggers for the specified device.

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

# GPQNV - Inquire Number of Definable View Table Entries

**GPQNV (*wstype*, *errind*, *number*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQNV** to inquire the number of definable view table entries for the specified workstation type.

The graPHIGS API returns data indicating the maximum number of views that can be defined in the view table for the specified workstation type. View zero is not included since it cannot be modified.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*number* **— returned by the graPHIGS API, fullword integer**
Number of definable view table entries. Entry 0 of the view table may not be changed.

### Error Codes

None

### Related Subroutines

**GPCIM2**
Create Image Mapping 2

**GPCIM3**
Create Image Mapping 3

**GPQRCT**
Inquire Realized Connection Type

**GPXVR**
Set Extended View Representation

### RCP code

201339653 (X'0C003305')

## GPQOPS - Inquire Open Structure

**GPQOPS (*type*, *strid*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPQOPS** to inquire the identifier of the open structure in the currently selected structure store.

If a structure is open, the graPHIGS API returns data indicating the type of open structure and the structure's identifier. If no structure is currently open, a type of `1=NIL` is returned.

### Parameters

*type* — **returned by the graPHIGS API, fullword integer**
Type of open structure (`1=NIL, 2=STRUCTURE`).

*strid* — **returned by the graPHIGS API, fullword integer**
Identifier of the open structure if type returned is `2=STRUCTURE`. If the value returned in the *type* parameter is `1=NIL`, this parameter is not set by the graPHIGS API and should be ignored.

### Error Codes

None

### Related Subroutines

None

### RCP code

201337094 (X'0C002906')

## GPQOPW - Inquire Set of Open Workstations

**GPQOPW (*start*, *number*, *errind*, *totnum*, *lwsid*)**

**Note:** This subroutine is a graPHIGS API State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQOPW** to inquire the list of the currently open workstations on a nucleus with an identifier of 1.

The graPHIGS API returns data indicating the total number of open workstation identifiers and a list of these identifiers that exist on a nucleus with an identifier of 1.

If the inquired information is available, the error indicator is returned as zero, and the values are returned in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available

data), then only the total-number parameter is set. If the inquired information is unavailable, the error indicator contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*start* **— specified by user, fullword integer**
Starting member of the list of open workstations (>=1).

*number* **— specified by user, fullword integer**
Number of open workstation identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of open workstations.

*lwsid* **— returned by the graPHIGS API, array of fullword integers.**
List of open workstation identifiers. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPCLWS**
Close Workstation

**GPDTR**
Detach Resource

**GPQATR**
Inquire List of Attached Resources

**RCP code**

201336326 (X'0C002606')

## GPQPAF - Inquire Pattern Facilities

**GPQPAF (*wstype*, *errind*, *maxrow*, *maxcol*, *indexes*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPAF** to inquire the pattern facilities for the specified workstation type.

The graPHIGS API returns the maximum pattern array dimensions, including the maximum number of rows (*maxrow*) and columns (*maxcol*) and the number of predefined pattern indexes (*indexes*) for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*maxrow* **— returned by the graPHIGS API, fullword integer**
Maximum number of rows in the pattern array.

*maxcol* **— returned by the graPHIGS API, fullword integer**
Maximum number of columns in the pattern array.

*indexes* **— returned by the graPHIGS API, fullword integer**
Number of predefined pattern indexes.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339658 (X'0C00330A')

## GPQPAR - Inquire Pattern Representation

**GPQPAR (*wsid*, *index*, *type*, *maxrow*, *maxcol*, *errind*, *drow*, *dcol*, *array*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQPAR** to inquire the current pattern representation in the specified entry in the pattern table of the specified workstation. This includes the pattern array row dimensions (*drow*), the column dimensions (*dcol*), and the pattern array (*array*) for the requested table entry.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Pattern table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*maxrow* **— specified by user, fullword integer**
Maximum number of rows to be returned by the graPHIGS API.

*maxcol* **— specified by user, fullword integer**
Maximum number of columns to be returned by the graPHIGS API.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**48** PATTERN INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**85** PATTERN INDEX VALUE < ONE

**90** INTERIOR STYLE NOT SUPPORTED ON WORKSTATION

**91** STARTING POINT OR DIMENSION < ONE

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**534** TYPE VALUE IS INVALID

*drow* **— returned by the graPHIGS API, fullword integer**
Pattern array row dimension (number of rows).

*dcol* **— returned by the graPHIGS API, fullword integer**
Pattern array column dimension (number of columns). The output parameters *drow* and *dcol* identify the actual size of the requested pattern entry. Depending on the values of *maxrow* and *maxcol*, these may or may not match the amount of data actually placed in the output area.

*array* **— returned by the graPHIGS API, array of fullword integers.**
Pattern array of color indexes in row order. The pattern array of color indexes is returned within the array bounds specified by *maxrow* and *maxcol*. Each pattern row is returned in the corresponding row of array beginning in column one. The high numbered rows and columns of the returned pattern are omitted as necessary to fit the number of rows and columns specified for array by *maxrow* and *maxcol*. Error indicator 533 is set in this case. If either dimension of the pattern is smaller than the dimension of array, the unused elements contain unpredictable values.

### Error Codes

None

### Related Subroutines

**GPPAR**
Set Pattern Representation

### RCP code

201339142 (X'0C003106')

## GPQPAS - Inquire Ancestors of Structure

**GPQPAS (*strid*, *order*, *depth*, *start*, *number*, *buflen*, *errind*, *actnum*, *actlen totnum*, *data*, *termcond*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPQPAS** to inquire the ancestral paths of a specified structure from the currently selected structure store. A path of ancestors of a structure *S* is a list of ordered pairs ((*A*1, *E*1), (*A*2, *E*2), (*S*,0)) where each ordered pair consists of an identifier of a structure (*Ax*) that is an ancestor of the specified structure (*S*) and the position of an execute structure-type element (*Ex*) that references the next structure in the path. Ancestor structure *A1* is the top of the path (e.g., it is not referenced by any other structure) and *S* is the bottom of the path.

The path order and path depth determine the portion of each path to be returned. The path depth determines the maximum number of ordered pairs returned in any one path. Specifying a path depth of zero returns each path in its entirety. When truncation occurs, the path order determines whether the head or tail portion of the path is returned. This truncation may result in two or more portions of paths having the same set of element references. Only one such portion is returned so that all of the returned path portions are distinct.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*strid* **— specified by user, fullword integer**
Structure identifier.

*order* **— specified by user, fullword integer**
Path order (`1=TOPFIRST, 2=BOTTOMFIRST`).

*depth* **— specified by user, fullword integer**
Path depth (>=0).

*start* **— specified by user, fullword integer**
Starting member of the list of paths (>=1).

*number* **— specified by user, fullword integer**
Number of paths requested (>=0).

*buflen* **— specified by user, fullword integer**
Length, in bytes, of the specified data parameter (*data*) (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**552** PATH ORDER IS INVALID

**558** PATH DEPTH < ZERO

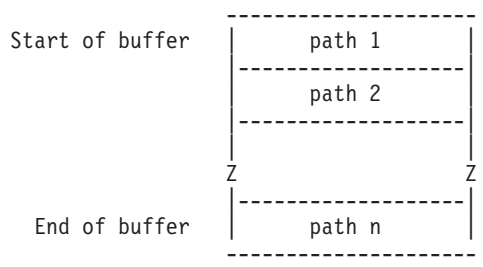
**577** BUFFER LENGTH IS < ZERO

*actnum* **— returned by the graPHIGS API, fullword integer**
Total number of paths returned.

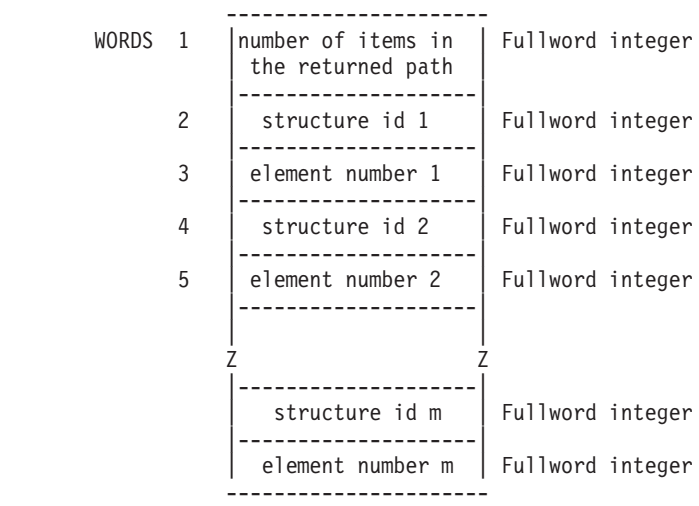
*actlen* **— returned by the graPHIGS API, fullword integer**
Total length, in bytes, of the paths returned in the data parameter (*data*).

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of the distinct paths available for the specified structure identifier.

*data* **— returned by the graPHIGS API, variable data**
The data buffer into which the paths are to be returned. The format of the data is as follows:




where each path has the following format:



*termcond* **— returned by the graPHIGS API, fullword integer**
Termination condition. The list of paths was terminated due to one of the following reasons:

**1-Count Exhausted**
The requested number of paths have been returned.

**2-Buffer Overflow**
The requested number of paths could not be returned because they would not all fit in the area provided. *actnum* contains the actual number returned.

**3-End of Paths**
No more paths exists. This condition supersedes the Count Exhausted condition (if that condition was in effect). Because of this, the total number of paths may or may not be equal to the requested number of paths to be returned. *actnum* should be checked to find the actual number of paths returned.

**4-Large Path**
The next path would not fit into the inbound buffer between the nucleus and the shell. *actnum* contains the number of paths returned excluding the path that would not fit.

### Error Codes

None

### Related Subroutines

**GPQPDS**
Inquire Descendents of Structure

### RCP code

201347586 (X'0C005201')

## GPQPCR - Inquire Predefined Color Representation

**GPQPCR (*wstype*, *start*, *number*, *errind*, *npred*, *indexes*, *colors*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQPCR** to inquire the color values in the predefined color table entries in the default color table for the specified workstation. If you can modify the workstation's display color table, then the display color table is the workstation's default. Otherwise, the rendering color table is the workstation's default color table. Use the Inquire Extended Color Facilities (**GPQXCF**) subroutine to inquire the characteristics of the workstation's color table.

The graPHIGS API returns the predefined color components corresponding to the specified indexes. This data includes the total number of predefined color table entries (*npred*), the index of the color values (*indexes*), and the color components for those indexes (*colors*).

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is

unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the predefined default color table entries(>0).

*number* **— specified by user, fullword integer**
Number of predefined default color table entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*npred* **- returned by the graPHIGS API, fullword integer**
Total number of predefined default color table entries.

*indexes* **— returned by the graPHIGS API, array of fullword integers**
List of indexes of the predefined color representation. The output array must be large enough to contain the requested data.

*colors* **— returned by the graPHIGS API, array of short floating-point numbers**
Color components to be interpreted by the default color model. The array contains a list of color table entries ordered by row. Each entry in the list refers to the corresponding index in the *indexes* array. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339905 (X'0C003401')

## GPQPCS - Inquire Primary Character Set

**GPQPCS (*wstype*, *errind*, *csid*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPCS** to inquire the primary character set identifier for the specified workstation type.

The graPHIGS API returns the primary character identifier for that workstation type. If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*csid* **— returned by the graPHIGS API, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339668 (X'0C003314')

## GPQPDC - Inquire Physical Device Characteristics

**GPQPDC (*wstype*, *category*, *device*, *number*, *errind*, *flags*, *type*, *totnum*, *vrange*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPDC** to retrieve the characteristics of a physical input device for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is

unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*category* **— specified by user, fullword integer**
Physical device category (1=BUTTON, 2=SCALAR, 3=2D_VECTOR).

*device* **— specified by user, fullword integer**
Physical device number (>=1).

*number* **— specified by user, fullword integer**
Number of array entries for parameter *vrange* that have been provided by the application (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**160** PHYSICAL INPUT DEVICE CATEGORY IS INVALID

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**539** REQUESTED NUMBER < ZERO

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*flags* **— returned by the graPHIGS API, fullword integer**
Indicates whether this device is actually present and if it may be emulated by the application (1=NOT_PRESENT_CANNOT_BE_EMULATED, 2=NOT_PRESENT_CAN_BE_EMULATED, 3=PRESENT_CANNOT_BE_EMULATED, 4=PRESENT_CAN_BE_EMULATED).

*type* **— returned by the graPHIGS API, fullword integer**
Type of physical input device. The interpretation of this parameter is dependent on the category of input device as follows:

**Button -**
(1=KEYBOARD, 2=OTHER). This field identifies whether the button device is a keyboard or not.

**Scalar -**
(1=ABSOLUTE, 2=RELATIVE). This parameter indicates whether the physical input device generates relative or absolute values.

**2D Vector -**
(1=ABSOLUTE, 2=RELATIVE). This parameter indicates whether the physical input device generates relative or absolute values.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of value ranges that describe the specified physical input device.

*vrange* **— returned by the graPHIGS API, array of fullword integers**
The parameter contains a list of integer pairs that define value ranges for the specified physical device. The interpretation of these value ranges are dependent on the category of the physical input device as follows:

**Button -**
Each value range defines a series of button values that may be generated by the device.

**Scalar -**
There is only one range for this device in all cases.

If the physical device generates absolute values then this range defines the minimum and maximum value that may be generated. The minimum and maximum values generally correspond to the minimum and maximum range of the logical input device.

If the device generates relative values, then the minimum of this range is always zero and has no meaning. The maximum of the range represents the number of increments in one unit of physical motion. A unit of physical motion is different for different physical devices.

> **Note:** For a dial physical device, one unit of physical motion equals one turn of the dial.

**2D Vector -**
There are two ranges for this device in all cases. The first corresponds to the *x* value that is generated and the second corresponds to the *y* value.

If the physical device generates absolute values, then this range defines the minimum and maximum value that may be generated. The minimum and maximum values generally correspond to the edges of the screen in the case of a device connected to the graphics cursor.

If the device generates relative values, then the minimum of this range is always zero and has no meaning. The maximum of the range represents the number of increments in one unit of physical motion. A unit of physical motion is different for different devices.

> **Note:** For a mouse physical device, one unit of physical motion equals the amount of motion necessary to move the cursor from one edge of the screen to its opposite edge.

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201339409 (X'0C003211')

## GPQPDS - Inquire Descendants of Structure

**GPQPDS (*strid*, *order*, *depth*, *start*, *number*, *buflen*, *errind*, *actnum*, *actlen totnum*, *data*, *termcond*)**

> **Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPQPDS** to inquire the descendant paths of a specified structure from the currently selected structure store. A path of descendants of a structure *S* is a list of ordered pairs ((*S*, *E*0), (*D*1, *E*1), (*D*2, *E*2), ...,

(*Dn*,0 *m*)), where each ordered pair consists of an identifier of a structure (*Dx*) that is a descendant of the specified structure (*S*) and the position of an execute structure-type element (*Ex*) that references the next structure in the path. The specified structure *S* is the top of the path and descendant structure *Dn* is the bottom of the path (e.g., it does not reference any other structure).

The path order and path depth determine the portion of each path to be returned. The path depth determines the maximum number of ordered pairs returned in any one path. Specifying a path depth of zero returns each path in its entirety. When truncation occurs, the path order determines whether the head or tail portion of the path is returned. This truncation may result in two or more portions of paths having the same set of element references. Only one such portion is returned so that all of the returned path portions are distinct.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*strid* **— specified by user, fullword integer**
Structure identifier.

*order* **— specified by user, fullword integer**
Path order (`1=TOPFIRST, 2=BOTTOMFIRST`).

*depth* **— specified by user, fullword integer**
Path depth (>=0).

*start* **— specified by user, fullword integer**
Starting member of the list of paths (>=1).

*number* **— specified by user, fullword integer**
Number of paths requested (>=0).

*buflen* **— specified by user, fullword integer**
Length, in bytes, of the specified data parameter (*data*) (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**552** PATH ORDER IS INVALID

**558** PATH DEPTH < ZERO

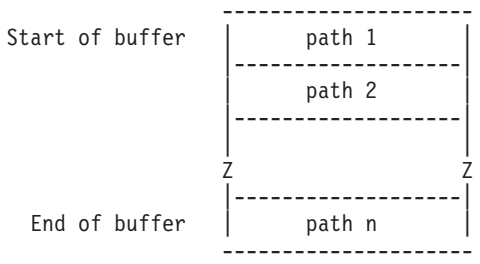
**577** BUFFER LENGTH IS < ZERO

*actnum* **— returned by the graPHIGS API, fullword integer**
Total number of paths returned.

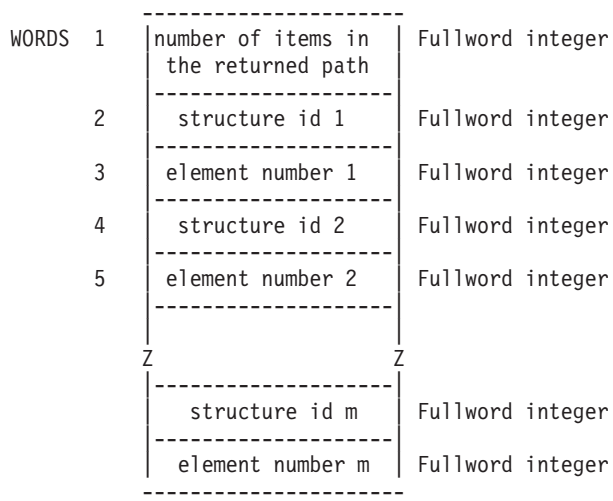
*actlen* **— returned by the graPHIGS API, fullword integer**
Total length, in bytes, of the paths returned in the data parameter (*data*).

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of the distinct paths available for the specified structure identifier.

*data* **— returned by the graPHIGS API, variable data**
The data buffer into which the paths are to be returned. The format of the data is as follows:




where each path has the following format:




*termcond* **— returned by the graPHIGS API, fullword integer**
Termination condition. The list of paths was terminated due to one of the following reasons:

**1-Count Exhausted**
The requested number of paths have been returned.

**2-Buffer Overflow**
The requested number of paths could not be returned because they would not all fit in the area provided. *actnum* contains the actual number returned.

**3-End of Paths**
No more paths exists. This condition supersedes the Count Exhausted condition (if that condition was in effect). Because of this, the total number of paths may or may not be equal to the requested number of paths to be returned. *actnum* should be checked to find the actual number of paths returned.

**4-Large Path**
The next path would not fit into the inbound buffer between the nucleus and the shell. *actnum* contains the number of paths returned excluding the path that would not fit.

## Error Codes

None

## Related Subroutines

**GPQPAS**
Inquire Ancestors of Structure

**RCP code**

201347586 (X'0C005202')

## GPQPER - Inquire Predefined Edge Representation

**GPQPER (*wstype*, *index*, *errind*, *edgefg*, *edgelt*, *edgesf*, *ecol*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPER** to inquire the predefined settings for the edge attributes in the edge bundle table for the specified workstation type.

The graPHIGS API returns the edge flag setting (*edgefg*), edge line type (*edgelt*), edge scale factor (*edgesf*), and edge color for the predefined edge bundle table (*ecol*). The returned attributes correspond to the requested bundle table index.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*index* **— specified by user, fullword integer**
Predefined edge bundle table index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*edgefg* **— returned by the graPHIGS API, fullword integer**
Edge flag corresponding to the specified entry (`1=OFF, 2=ON, 3=GEOMETRY_ONLY`).

*edgelt* **— returned by the graPHIGS API, fullword integer**
Line type of an edge corresponding to the specified entry.

*edgesf* **— returned by the graPHIGS API, short floating-point number**
Edge scale factor corresponding to the specified entry.

*ecol* **— returned by the graPHIGS API, fullword integer**
Edge color index corresponding to the specified entry.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339906 (X'0C003402')

## GPQPIR - Inquire Predefined Interior Representation

**GPQPIR (*wstype*, *index*, *errind*, *style*, *sindex*, *icol*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPIR** to inquire the predefined settings for the interior attributes in the interior bundle table for the specified workstation type.

The graPHIGS API returns the interior style (*style*), the style index (*sindex*), and the interior color index for the predefined interior bundle table (*icol*). The returned attributes correspond to the requested bundle table index.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*index* **— specified by user, fullword integer**
Predefined interior bundle table index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*style* — **returned by the graPHIGS API, fullword integer**
Interior style corresponding to the specified entry (1=HOLLOW, 2=SOLID, 3=PATTERN, 4=HATCH, 5=EMPTY).

*sindex* — **returned by the graPHIGS API, fullword integer**
Style index corresponding to the specified entry.

*icol* — **returned by the graPHIGS API, fullword integer**
Interior color index corresponding to the specified entry.

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201339907 (X'0C003403')

## GPQPK - Inquire Pick Device State

**GPQPK (*wsid*, *device*, *type*, *inlen*, *exlen*, *pathlen*, *length*, *errind*, *mode*, *echosw*, *inclen*, *incl*, *exclen*, *excl*, *depth*, *pickpath*, *echo*, *area*, *datalen*, *data*, *order*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQPK** to inquire the current state of the specified pick device attached to a specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*device* — **specified by user, fullword integer**
Pick device number.

*type* — **specified by user, fullword integer**
Type of returned values (1=SET).

*inlen* — **specified by user, fullword integer**
Length of inclusion filter array provided by the application for the graPHIGS API to return the corresponding data (>=0).

*exlen* — **specified by user, fullword integer**
Length of exclusion filter array provided by the application for the graPHIGS API to return the corresponding data (>=0).

*pathlen* **— specified by user, fullword integer**
Length of initial pick path array provided by the application for the graPHIGS API to return the corresponding data.

*length* **— specified by user, fullword integer**
Length of pick data record array provided by the application for the graPHIGS API to return the corresponding data.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**37** WORKSTATION IS NOT OF CATEGORY OUTIN

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**534** TYPE VALUE IS INVALID

*mode* **— returned by the graPHIGS API, fullword integer**
Current operating mode (`1=REQUEST, 2=SAMPLE, 3=EVENT, 4=APPLICATION_DEFINED`). The graPHIGS API only returns a mode of `4=APPLICATION_DEFINED` if the application set the device mode using the Set Input Device State (**GPIDMO**) subroutine and the mode does not emulate Request, Sample or Event mode.

*echosw* **— returned by the graPHIGS API, fullword integer**
Current echo switch (`1=NOECHO, 2=ECHO`).

*inclen* **— returned by the graPHIGS API, fullword integer**
Current inclusion pick filter length.

*incl* **— returned by the graPHIGS API, array of fullword integers**
Current inclusion pick filter.

*exclen* **— returned by the graPHIGS API, fullword integer**
Current exclusion pick filter length.

*excl* **— returned by the graPHIGS API, array of fullword integers**
Current exclusion pick filter.

*depth* **— returned by the graPHIGS API, fullword integer**
Current initial pick path depth.

*pickpath* **— returned by the graPHIGS API, array of fullword integers**
Current initial pick path.

The pickpath array contains a list of pickpath entries in row order. Each entry is a triplet consisting of a structure identifier, pick identifier, and element number.

*echo* **— returned by the graPHIGS API, fullword integer**
Current prompt/echo type.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Current echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*datalen* **— returned by the graPHIGS API, fullword integer**
Current pick data record length.

*data* **— returned by the graPHIGS API, variable length data**
Current pick data record.

*order* **— returned by the graPHIGS API, fullword integer**
Current pick path order (`1=TOP_FIRST, 2=BOTTOM_FIRST`).

**Error Codes**

None

**Related Subroutines**

**GPIDMO**
Set Input Device Mode

**GPINPK**
Initialize Pick

**GPQDPK**
Inquire Default Pick Device Data

**GPQNCN**
Inquire Number of Available Class Names

**RCP code**

201338883 (X'0C003003')

## GPQPKA - Inquire Pick Aperture

**GPQPKA (*wsid*, *device*, *errind*, *size*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQPKA** to inquire the current size of the pick aperture for the specified pick device on the specified workstation.

The pick aperture is returned as the length of a side of a square in Device Coordinates (DC).

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Pick device number.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**37** WORKSTATION IS NOT OF CATEGORY OUTIN

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

*size* **— returned by the graPHIGS API, short floating-point number (DC)**
Aperture size. This is specified as the length of a side of a square in Device Coordinates (DC).

### Error Codes

None

### Related Subroutines

None

### RCP code

201338888 (X'0C003008')

## GPQPKT - Inquire Pick Measure Type

**GPQPKT (*wstype*, *device*, *errind*, *type*)**

> **Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQPKT** to determine the type of measure that the pick device supports for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*device* **— specified by user, fullword integer**
Device number (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*type* **— returned by the graPHIGS API, fullword integer**
Pick measure type (`1=NORMAL, 2=EXTENDED`).

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339410 (X'0C003212')

## GPQPLF - Inquire Polyline Facilities

**GPQPLF (*wstype*, *start*, *number*, *errind*, *ntype*, *ltype*, *nlwidth*, *lwidth*, *minlw*, *maxlw*, *npred*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPLF** to inquire the polyline facilities for the specified workstation type.

The graPHIGS API returns the total number of available line types and their identifiers (*ntype*[default] the number of available line widths (*nlwidth*) and the nominal (*lwidth*), minimum (*minlw*), and maximum line width size (*maxlw*[default] and the number of predefined polyline bundle table indexes for the specified workstation type (*npred*).

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of line types (>=1).

*number* **— specified by user, fullword integer**
Number of line types requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*ntype* **— returned by the graPHIGS API, fullword integer**
Total number of available line types.

*ltype* **— returned by the graPHIGS API, array of fullword integers.**
List of available line types in the workstation's line type table. The table size and specified entries supported are workstation dependent. The default line type table for supported entries is defined with the following line types: (1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE). The output array must be large enough to contain the requested data.

*nlwidth* **— returned by the graPHIGS API, fullword integer**
Number of available line widths. (Zero means that the workstation supports a continuous range of line widths.)

*lwidth* **— returned by the graPHIGS API, short floating-point number (DC)**
Nominal line width.

*minlw* **— returned by the graPHIGS API, short floating-point number (DC)**
Minimum line width.

*maxlw* **— returned by the graPHIGS API, short floating-point number (DC)**
Maximum line width.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined polyline bundle table entries.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339654 (X'0C003306')

# GPQPLR - Inquire Predefined Polyline Representation

**GPQPLR (*wstype*, *index*, *errind*, *ltype*, *lwidth*, *color*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPLR** to inquire the predefined polyline attributes in an entry of the bundle table for the specified workstation type.

The graPHIGS API returns the polyline type (*ltype*), width (*lwidth*), and color (*color*) for the specified index of the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*index* **— specified by user, fullword integer**
Predefined polyline bundle table index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*ltype* **— returned by the graPHIGS API, fullword integer**
Line type corresponding to the specified entry in the workstation's line type table. The table size and specific entries supported are workstation dependent. The default line type table for supported entries is defined with the following line types: (`1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE`).

*lwidth* **— returned by the graPHIGS API, short floating-point number**
Polyline width scale factor corresponding to the specified entry.

*color* **— returned by the graPHIGS API, fullword integer**
Polyline color index corresponding to the specified entry.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339908 (X'0C003404')

## GPQPMF - Inquire Polymarker Facilities

**GPQPMF (*wstype*, *start*, *number*, *errind*, *ntype*, *mtype*, *nsize*, *size*, *minms*, *maxms*, *npred*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPMF** to inquire the polymarker facilities for the specified workstation type.

The graPHIGS API returns data indicating the total number of available marker types and their identifiers (*ntype*[default] the nominal (*size*), minimum (*minms*), and maximum (*maxms*) marker sizes; and the

number of predefined polymarker bundle table indexes for the specified workstation type (*npred*). If the number of available marker sizes is returned as zero, then the workstation supports a continuous range of marker sizes.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of marker types (>=1).

*number* **— specified by user, fullword integer**
Number of polymarker types requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*ntype* **— returned by the graPHIGS API, fullword integer**
Total number of available marker types.

*mtype* **— returned by the graPHIGS API, array of fullword integers.**
List of available maker types in the workstation's available marker type table. The table size and specific entries supported are workstation dependent. The default marker type table for supported entries is defined with the following marker types: (`1=DOT, 2=PLUS_SIGN, 3=ASTERISK, 4=CIRCLE, 5=DIAGONAL_CROSS, 6-n=ASTERISK`).

*nsize* **— returned by the graPHIGS API, fullword integer**
Number of available marker sizes. Zero means that the workstation supports a continuous range of marker sizes.

*size* **— returned by the graPHIGS API, short floating-point number (DC)**
Nominal marker size.

*minms* **— returned by the graPHIGS API, short floating-point number (DC)**
Minimum marker size.

*maxms* **— returned by the graPHIGS API, short floating-point number (DC)**
Maximum marker size.

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined polymarker bundle table entries.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339655 (X'0C003307')

## GPQPMR - Inquire Predefined Polymarker Representation

**GPQPMR (*wstype*, *index*, *errind*, *mtype*, *msize*, *color*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPMR** to inquire the predefined polymarker attributes corresponding to the specified entry in the predefined bundle table for the specified workstation type.

The graPHIGS API returns the polymarker type (*mtype*), size (*msize*), and color (*color*) for the specified index of the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*index* **— specified by user, fullword integer**
Predefined polymarker bundle table index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*mtype* **— returned by the graPHIGS API, fullword integer**
Marker type corresponding to the specified entry in the workstation's marker type table. The table size and specific entries supported are workstation dependent. The default marker type table for supported entries is defined with the following marker types: (`1=DOT, 2=PLUS_SIGN, 3=ASTERISK, 4=CIRCLE, 5=DIAGONAL_CROSS, 6-n=ASTERISK`).

*msize* **— returned by the graPHIGS API, short floating-point number**
Marker size scale factor corresponding to the specified entry in the polymarker bundle table.

*color* **— returned by the graPHIGS API, fullword integer**
Polymarker color index corresponding to the specified entry in the polymarker bundle table.

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201339909 (X'0C003405')

## GPQPO - Inquire Available Pixel Operations

**GPQPO (*ncid*, *type*, *start*, *number*, *errind*, *totnum*, *op*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQPO** to inquire the available pixel operations on the specified nucleus.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*type* **— specified by user, fullword integer**
Type of pixel operations (`1=TWO_OPERAND, 2=THREE_OPERAND`).

*start* **— specified by user, fullword integer**
Starting member in the list of available operations (>=1).

*number* **— specified by user, fullword integer**
Number of pixel operation entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**295** PIXEL OPERATION TYPE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available operand pixel operations.

*op* **— returned by the graPHIGS API, array of fullword integers**
List of available operations.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201345802 (X'0C004B0A')

## GPQPPR - Inquire Predefined Pattern Representation

**GPQPPR (*wstype*, *index*, *maxrow*, *maxcol*, *errind*, *drow*, *dcol*, *array*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQPPR** to inquire a predefined pattern table entry for the specified workstation type.

For the specified workstation type, this inquiry returns the values corresponding to the specified index of the predefined pattern table. The graPHIGS API returns the number of pattern array rows (*drow*) and columns (*dcol*), and the array of color indexes corresponding to the pattern (*array*).

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*index* **— specified by user, fullword integer**
Pattern table index (>=1).

*maxrow* **— specified by user, fullword integer**
Maximum number of requested pattern array rows (>=1).

*maxcol* **— specified by user, fullword integer**
Maximum number of requested pattern array columns (>=1)

The parameters *maxrow* and *maxcol* define the dimensions of the application's output area *array* Using these values, the graPHIGS API either fills in or truncates the output information, as appropriate.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**48** PATTERN INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**85** PATTERN INDEX VALUE < ONE

**90** INTERIOR STYLE NOT SUPPORTED ON WORKSTATION

**91** STARTING POINT OR DIMENSION < ONE

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*drow* **— returned by the graPHIGS API, fullword integer**
Total number of pattern array rows in the entry specified.

*dcol* **— returned by the graPHIGS API, fullword integer**
Total number of pattern array columns in the entry specified.

The output parameters *drow* and *dcol* identify the actual size of the requested pattern entry. Depending on the values of *maxrow* and *maxcol*, these may or may not match the amount of data actually placed in the output area.

*array* **— returned by the graPHIGS API, array of fullword integers**
Pattern array of color indexes in row order. The pattern array of color indexes is returned within the array bounds specified by *maxrow* and *maxcol*. Each pattern row is returned in the corresponding row of *array* beginning in column 1. The high numbered rows and columns of the returned pattern are omitted as necessary to fit the number of rows and columns specified for *array* by *maxrow* and *maxcol*. Error indicator 533 is set in this case. If either dimension of the pattern is smaller than the dimension of *array*, then the unused elements contain unpredictable values.

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201339910 (X'0C003406')

## GPQPTR - Inquire Predefined Text Representation

**GPQPTR (*wstype*, *index*, *errind*, *font*, *prec*, *factor*, *space*, *color*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQPTR** to inquire a set of predefined text attributes as set in the text bundle table for the specified workstation type.

The graPHIGS API returns the requested index for text font (*font*) and precision (*prec*), character expansion factor (*factor*), character spacing (*space*), and text color (*color*) for the specified entry.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*index* **— specified by user, fullword integer**
Predefined text bundle table index (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*font* **— returned by the graPHIGS API, fullword integer**
Text font corresponding to the specified entry.

*prec* **— returned by the graPHIGS API, fullword integer**
Text precision corresponding to the specified entry (`1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC`).

*factor* **— returned by the graPHIGS API, short floating-point number**
Character expansion factor corresponding to the specified entry (the deviation of the width-to-height ratio of the font).

*space* **— returned by graPHIGS API, short floating-point number**
Character spacing corresponding to the specified entry, which is specified as a fraction of the font-nominal character height.

*color* **— returned by the graPHIGS API, fullword integer**
Text color index corresponding to the specified entry.

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339911 (X'0C003407')

## GPQRCM - Inquire Available Rendering Color Models

**GPQRCM (*wstype*, *start*, *number*, *errind*, *totnum*, *model*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQRCM** to inquire a list of available rendering color models for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available rendering color models (>=1).

*number* **— specified by user, fullword integer**
Number of rendering color models requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available rendering color models.

*model* **— returned by the graPHIGS API, array of fullword integers**
List of rendering color models (`1=RGB_NORMAL, 2=RGB_B_ONLY`).

**Error Codes**

None

**Related Subroutines**

**GPCPR**
Set Color Processing Representation

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339404 (X'0C00320C')

## GPQRCT - Inquire Realized Connection and Type

**GPQRCT (*wsid*, *ilen*, *errind*, *olen*, *connid*, *wstype*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQRCT** to inquire the connection identifier and the realized workstation type of the specified workstation identifier. When you create a workstation using the Open Workstation (**GPOPWS**) subroutine, or the Create Workstation (**GPCRWS**) subroutine, the graPHIGS API assigns a workstation type. This type is the realized workstation type.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 536 (the actual connection identifier is greater than the length of the area provided), then only the actual length (*olen*) of the connection identifier is returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ilen* **— specified by user, fullword integer**
Length of the area provided to contain the connection identifier.

*errind* **— returned by the graPHIGS API fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**536** INQUIRY DATA EXCEEDS AREA. LENGTH OF REQUIRED AREA RETURNED

*olen* **— returned to user, fullword integer**
Actual length of the connection identifier.

*connid* **— returned by the graPHIGS API variable length character string**
Connection identifier.

*wstype* **— returned by the graPHIGS API 8-byte character string**
Realized workstation type.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201336850 (X'0C002812')

## GPQRST - Inquire Referencing Structures

**GPQRST (*strid*, *start*, *number*, *errind*, *totnum*, *istrid*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

**Purpose**

Use **GPQRST** to inquire a list of identifiers of structures that contain execute structure-type elements (execute structure elements and conditional execute structure elements) that reference the specified structure.

A structure identifier can be in the list once for every execute structure-type element that it contains that references the specified structure.

If the inquired information is available, the error indicator is returned as zero, and the values are returned in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number parameter is set. If the inquired information is unavailable, the error indicator contains an error number indicating the reason and the values returned in the output parameters are unpredictable.

**Parameters**

*strid* **— specified by user, fullword integer**
Structure identifier.

*start* **— specified by user, fullword integer**
The starting member of the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of referencing structures.

*istrid* **— returned by the graPHIGS API, array of fullword integers**
List of structure identifiers that reference the specified structure.

**Error Codes**

None

**Related Subroutines**

**GPQEXS**
Inquire Executed Structures

**RCP code**

201337102 (X'0C00290E')

## GPQRV - Inquire Set of Roots in View

**GPQRV (*wsid*, *view*, *start*, *number*, *errind*, *totnum*, *strid*, *priority*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQRV** to inquire a list of the root structure identifiers associated with the specified view.

Structure identifiers returned by this subroutine are those in the structure store associated with the view.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*start* **— specified by user, fullword integer**
Starting member of the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of structure identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of the roots in the view.

*strid* **— returned by the graPHIGS API, array of fullword integers**
List of root structure identifiers.

*priority* **— returned by the graPHIGS API, array of short floating-point numbers**
List of root priorities.

**Error Codes**

None

**Related Subroutines**

**GPARV**
Associate Root with View

**RCP code**

201337095 (X'0C002907')

## GPQRVE - Inquire Requested View Table Entries Input

**GPQRVE (*wsid*, *start*, *number*, *errind*, *totnum*, *view*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQRVE** to inquire the requested view table indexes in input priority order for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of view table entries (>=1).

*number* **— specified by user, fullword integer**
Number of view table entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of view table entries.

*view* **— returned by the graPHIGS API, array of fullword integers**
List of view table indexes, in decreasing view input priority order. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPVIP**
Set View Input Priority

**RCP code**

201336839 (X'0C002807')

## GPQRVO - Inquire Requested View Table Entries Output

**GPQRVO (*wsid*, *start*, *number*, *errind*, *totnum*, *view*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQRVO** to inquire the requested view table indexes in output priority order for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of view table entries (>=1).

*number* **— specified by user, fullword integer**
Number of view table entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of view table entries.

*view* **— returned by the graPHIGS API, array of fullword integers**
List of view table indexes, in decreasing view output priority order. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPVIP**
Set View Input Priority

**RCP code**

201336849 (X'0C002811')

# GPQRVR - Inquire Requested View Representation

**GPQRVR (*wsid*, *view*, *number*, *ids*, *errind*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQRVR** to inquire one or more fields from the specified requested view table entry.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. The output parameter must be large enough to store all requested data.

If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*number* **— specified by user, fullword integer**
Number of groups requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**571** INQUIRED INFORMATION IS NOT AVAILABLE

*data* **— returned by the graPHIGS API, variable data**
Data array containing the values in the requested groups. Each field in the view table entry is identified by a group identifier. The value to be set for each group is expressed in the following data formats:

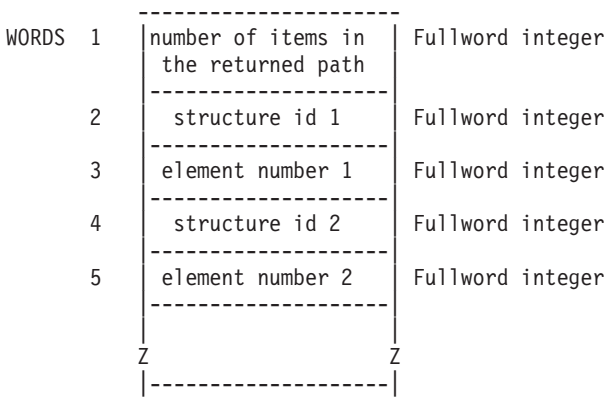
**Group Identifier 1 - Window clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP)

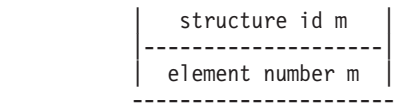
**Group Identifier 2 - Near clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP)

**Group Identifier 3 - Far clipping indicator**
A fullword integer (1=NOCLIP, 2=CLIP)

**Group Identifier 4 - Shielding indicator**
A fullword integer (1=OFF, 2=ON)

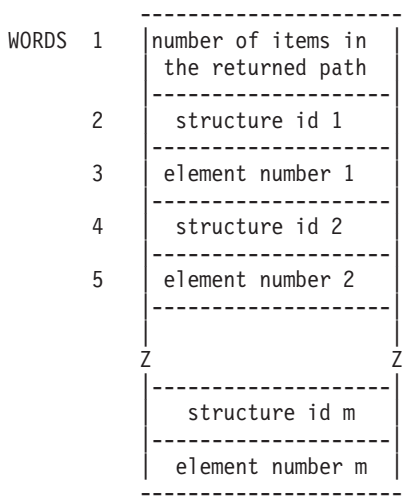
**Group Identifier 5 - Shielding color**
Four fullwords of data with either of the following two formats:






**Group Identifier 6 - Border indicator**
A fullword integer (1=OFF, 2=ON)

**Group Identifier 7 - Border color**
Four fullwords of data with either of the following two formats:




**Group Identifier 8 - Reserved**
This field is reserved.

**Group Identifier 9 - Temporary view indicator**
A fullword integer (1=OFF, 2=ON)

**Group Identifier 10 - HLHSR mode**
A fullword integer (1=OFF, 2=ON_THE_FLY)

**Group Identifier 11 - Transparency processing mode**
A fullword integer (1=OFF, 2=PARTIAL_TRANSPARENT, 3=BLEND, 4=BLEND_ALL)

**Group Identifier 12 - Initial color processing mode index**
A fullword integer (>=0)

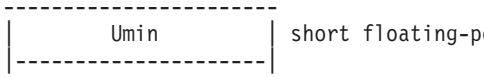
**Group Identifier 13 - Initial frame buffer write protect mask**
A 32-bit bit string.

**Group Identifier 14 - Viewport, 2D form**
4 short floating-point numbers (including only Xmin, Xmax, Ymin, Ymax). For the set subroutine, Zmin and Zmax are set to their default values.

**Group Identifier 15 - Viewport, 3D form**
6 short floating-point numbers (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax)

**Group Identifier 16 - View volume, 2D form**
4 short floating-point numbers specifying the view window (Umin, Umax, Vmin, Vmax). For the set subroutine, other fields of the view volume group are set to their default values.

**Group Identifier 17 - View volume, 3D form**
10 short floating-point numbers and a fullword integer specifying a view window (Umin,Umax,Vmin,Vmax), near plane distance, far plane distance, projection reference point (*u*, *v*, *n*), view plane distance and a projection type (1=PARALLEL, 2=PERSPECTIVE).



| | |
|---|---|
| Umax | short floating-point number |
| Vmin | short floating-point number |
| Vmax | short floating-point number |
| near plane distance | short floating-point number |
| far plane distance | short floating-point number |
| U projection-reference point | short floating-point number |
| V projection-reference point | short floating-point number |
| N projection-reference point | short floating-point number |
| view plane distance | short floating-point number |
| projection type | fullword integer |

**Group Identifier 18 - View matrix, 2D form**

9 short floating-point numbers. For the output matrix, the elements are in the following order:

```
|m11  m12  m14|
|m21  m22  m24|--->(m11,m12,m14,m21.....m44)
|m41  m42  m44|
```

The output 3[default]3 matrix is extracted by the graPHIGS API from the 4[default]4 matrix of the three-dimensional form:

```
|m11  m12  m13  m14|
|m21  m22  m23  m24|--->(m11,m12,m13,m14,m21,m22.....m44)
|m31  m32  m33  m34|
|m41  m42  m43  m44|
```

**Group Identifier 19 - View matrix, 3D form**

16 short floating-point numbers, M11, M12, M13. For the output view matrix, the elements are in the following order:

```
|m11  m12  m13  m14|
|m21  m22  m23  m24|--->(m11,m12,m13,m14,m21,m22.....m44)
|m31  m32  m33  m34|
|m41  m42  m43  m44|
```

**Group Identifier 20 - View input active flag**

A fullword integer (1=INACTIVE, 2=ACTIVE)

**Group Identifier 21 - View output active flag**

A fullword integer (1=INACTIVE, 2=ACTIVE)

**Group Identifier 22 - View mapping matrix, 2D form**

9 short floating-point numbers:

```
|m11  m12  m14|
|m21  m22  m24|--->(m11,m12,m14,m21.....m44)
|m41  m42  m44|
```

**Group Identifier 23 - View mapping matrix, 3D**

16 short floating-point numbers:

```
|m11  m12  m13  m14|
|m21  m22  m23  m24|--->(m11,m12,m13,m14,m21,m22.....m44)
|m31  m32  m33  m34|
|m41  m42  m43  m44|
```

**Group Identifier 24 - Antialiasing mode**
A fullword integer (1=OFF, 2=SUBPIXEL_ON_THE_FLY, 3=NON_SUBPIXEL_ON_THE_FLY)

**Group Identifier 25 - Shield alpha value**
A fullword integer (0<=*alpha*<=255)

### Error Codes

None

### Related Subroutines

**GPQAMO**
Inquire Available Antialiasing Modes

**GPQCVR**
Inquire Current View Representation

**GPQHMO**
Inquire Available HLHSR Modes

**GPQTMO**
Inquire Available Transparency Modes

**GPXVR**
Set Extended View Representation

### RCP code

201336847 (X'0C00280F')

## GPQSDF - Inquire Surface Display Facilities

**GPQSDF (*wstype*, *start*, *number*, *errind*, *order*, *totnum*, *criteria*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQSDF** to inquire the surface facilities for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* — **specified by user, fullword integer**
Starting member of the list of available surface approximation criteria (>=1).

*number* — **specified by user, fullword integer**
Number of surface approximation criteria requested (>=0).

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*order* — **returned by the graPHIGS API, fullword integer**
Maximum surface order supported.

*totnum* — **returned by the graPHIGS API, fullword integer**
Total number of available surface approximation criteria.

*criteria* — **returned by the graPHIGS API, array of fullword integers**
List of surface approximation criteria (`1=WORKSTATION_DEPENDENT`, `3=CONSTANT_SUBDIVISION_BETWEEN_KNOTS`, `8=VARIABLE_SUBDIVISION_BETWEEN_KNOTS`).

### Error Codes

None

### Related Subroutines

**GPNBS**
Non-Uniform B-Spline Surface

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201346059 (X'0C004C0B')

## GPQSEV - Inquire More Simultaneous Events

**GPQSEV (*simevnt*)**

**Note:** This subroutine is a State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQSEV** to inquire whether additional simultaneous events are waiting in the input queue.

The graPHIGS API returns a value indicating that additional events are waiting that occurred from the same device trigger as the event previously in the current event report (CEV).

This subroutine can be called after the appropriate Get subroutine was used to remove the previous event from the CEV.

**Parameters**

*simevnt* **— returned by the graPHIGS API, fullword integer**
More simultaneous events (1=NOMORE, 2=MORE).

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201336327 (X'0C002607')

## GPQSH - Inquire Shell Identifier

**GPQSH (*ncid*, *errind*, *shid*, *env*)**

**Note:** This subroutine is a State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQSH** to inquire your application's shell identifier on the specified nucleus and its environment descriptor. The descriptor contains information about the environment for the application process issuing the inquiry.

If this application process is communicating with another application process in a different environment, and it passes data to that application process, or it receives data from that application process, it may need to convert the data to a form that will be recognized by the application process using the data. The differences in data in different environments could be:

- character encoding can be EBCDIC or ASCII
- floating-point format can be IBM single precision or IEEE single precision
- the byte order of data can be swapped

The Convert Data (**GPCVD**) subroutine allows you to convert data by specifying an environment descriptor and origin parameter to a form that another application process can use.

An application process may pass its environment descriptor to another application process by issuing the Send Broadcast Message (**GPSBMS**) subroutine, or Send Private Message (**GPSPMS**) subroutine.

If the inquired information is available, the error indicator is returned as zero, and the values are returned in the output parameters. If the inquired information is unavailable, the error indicator contains an error number indicating the reason and the values returned in the output parameters are unpredictable.

**Parameters**

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**202** SPECIFIED NUCLEUS DOES NOT EXIST

*shid* **— returned by the graPHIGS API, fullword integer**
Shell identifier.

*env* **— returned by the graPHIGS API, 4-byte character string**
Environment descriptor.

**Error Codes**

None

**Related Subroutines**

**GPATR**
Attach Resource

**GPCVD**
Convert Data

**GPSBMS**
Send Broadcast Message

**GPSPMS**
Send Private Message

**RCP code**

201345795 (X'0C004B03')

## GPQSHD - Inquire Shell Deferral State

**GPQSHD (*deferral*, *update*)**

**Note:** This subroutine is a State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQSHD** to inquire the current shell deferral state values.

**Parameters**

*deferral* **— returned by the graPHIGS API, fullword integer**
Shell deferral mode (`1=FLUSH, 2=DEFERRED, 3 DEFERRED_PLUS_MSGS`).

*update* **— returned by the graPHIGS API, fullword integer**
Update notification mode (`1=NO, 2=YES`).

**Error Codes**

None

**Related Subroutines**

**GPSHDF**
Set Shell Deferral State

**RCP code**

201345799 (X'0C004B07')

## GPQSID - Inquire List of Socket Identifiers

**GPQSID (*start*, *number*, *errind*, *totnum*, *socketids*)**

### Purpose

Use **GPQSID** to inquire the currently existing socket identifiers in use by the graPHIGS API and available to the graPHIGS shell. This inquiry is intended for applications that use the graPHIGS SYNCPROC default. See *The graPHIGS Programming Interface: Technical Reference* for further information on the SYNCPROC default and the ramifications of its use.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (totnum) parameter is set. If the inquired information is unavailable, then the error indicator (errind) contains an error number indicating the reason, and the values returned in the output parameters are unpredicatable.

### Parameters

*start* **— specified by user, fullword integer**
Starting member of the list of socket identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** STARTS EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of socket identifiers in list.

*socketids* **— returned by the graPHIGS API, array of fullword integers**
List of socket identifiers.

### Error Codes

None

### Related Subroutines

**GPRDEV**
Redrive X Events

### RCP code

# GPQSK - Inquire Stroke Device State

**GPQSK (*wsid*, *device*, *type*, *lenpts*, *length*, *errind*, *mode*, *echosw*, *view*, *npoint*, *pointarray*, *echo*, *area*, *buflen*, *editpos*, *datalen*, *data*)**

> **Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQSK** to inquire the current state of a stroke device attached to the specified workstation.

The graPHIGS API returns the current values for the specified device.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Stroke device number (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (`1=SET`).

*lenpts* **— specified by user, fullword integer**
Length of initial stroke points array, in points, provided by the application for the graPHIGS API to return corresponding data.

*length* **— specified by user, fullword integer**
Length of requested stroke data array provided by the application for the graPHIGS API to return corresponding data.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**534** TYPE VALUE IS INVALID

*mode* **— returned by the graPHIGS API, fullword integer**
Current operating mode (`1=REQUEST, 2=SAMPLE, 3=EVENT, 4=APPLICATION_DEFINED`). The graPHIGS

API only returns a mode of 4=APPLICATION_DEFINED if the application set the device mode using the Set Input Device State (**GPIDMO**) subroutine and the mode does not emulate Request, Sample, or Event mode.

*echosw* **— returned by the graPHIGS API, fullword integer**
Current echo switch (1=NOECHO, 2=ECHO).

*view* **— returned by the graPHIGS API, fullword integer**
Current initial view index.

*npoint* **— returned by the graPHIGS API, fullword integer**
Current number of points in the initial stroke.

*pointarray* **— returned by the graPHIGS API, array of short floating-point numbers (WC)**
Current coordinates of initial points in stroke. Array is formatted as a list of points in row order.

*echo* **— returned by the graPHIGS API, fullword integer**
Current prompt/echo type.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Current echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*buflen* **— returned by the graPHIGS API, fullword integer**
Current stroke input buffer size in points.

*editpos* **— returned by the graPHIGS API, fullword integer**
Current editing position.

*datalen* **— returned by the graPHIGS API, fullword integer**
Current stroke data record length in bytes.

*data* **— returned by the graPHIGS API, variable length data**
Current stroke data record.

### Error Codes

None

### Related Subroutines

**GPIDMO**
Set Input Device Mode

**GPQDSK**
Inquire Default Stroke Device Data

**GPINSK**
Initialize Stroke

### RCP code

201338884 (X'0C003004')

## GPQSPD - Inquire Source Physical Device

**GPQSPD (*wstype*, *class*, *ldevice*, *errind*, *category*, *pdevice*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQSPD** to inquire the category and number of the physical device that is connected to the specified logical input device.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*class* **- specified by user, fullword integer**
Logical input device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*ldevice* **- specified by user, fullword integer**
Logical device number (>=1).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*category* **— returned by the graPHIGS API, fullword integer**
Physical device category (`1=BUTTON, 2=SCALAR, 3=2D_VECTOR`).

*pdevice* **— returned by the graPHIGS API, fullword integer**
Physical device number.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339408 (X'0C003210')

## GPQSPL - Inquire Shell Product Level

**GPQSPL (*level*)**

**Note:** This subroutine is a State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQSPL** to obtain the product level of the shell that the application is using.

The information that is returned will be an array of 3 fullword integers containing the product level information of the shell which consists of:

- Version level
- Release level
- Modification level.

### Parameters

*level* **— returned by the graPHIGS API, 3 fullword integers**
The information returned will be the Version, Release, and Modification level of the shell that the application is using.

### Error Codes

None

### Related Subroutines

None

### RCP code

201345804 (X'0C004B0C')

## GPQSSS - Inquire Selected Structure Store

**GPQSSS (*status*, *ssid*)**

**Note:** This subroutine is a State List (PSL) inquiry. For an overview, see "PSL Inquiries."

### Purpose

Use **GPQSSS** to inquire the currently selected structure store.

If the selected structure store exists, the graPHIGS API returns value `2=EXISTENT` to the *status* parameter and the structure store identifier to the second parameter. If the selected structure store does not exist, the graPHIGS API returns value `1=NON_EXISTENT` to the *status* parameter and the second parameter is not set.

### Parameters

*status* **— returned by the graPHIGS API, fullword integer**
Selected structure store status (`1=NON_EXISTENT`, `2=EXISTENT`).

*ssid* **— returned by the graPHIGS API, fullword integer**
Structure store identifier.

### Error Codes

None

### Related Subroutines

**GPSSS**
Select Structure Store

**RCP code**

201345797 (X'0C004B05')

# GPQST - Inquire String Device State

**GPQST (*wsid*, *device*, *type*, *slen*, *length*, *errind*, *mode*, *echosw*, *strlen*, *string*, *echo*, *area*, *buflen*, *editpos*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQST** to inquire the current state of a string device attached to the specified workstation.

The graPHIGS API returns the current values of the specified device.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
String device number (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*slen* **— specified by user, fullword integer**
Length of the current initial string array, in bytes, provided by the application for the graPHIGS API to return corresponding data.

*length* **— specified by user, fullword integer**
Length of requested string data record array, in bytes, provided by the application for the graPHIGS API to return corresponding data.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**534** TYPE VALUE IS INVALID

*mode* **— returned by the graPHIGS API, fullword integer**
Current operating mode (1=REQUEST, 2=SAMPLE, 3=EVENT, 4=APPLICATION_DEFINED). The graPHIGS API only returns a mode of 4=APPLICATION_DEFINED if the application set the device mode using the Set Input Device State (**GPIDMO**) subroutine and the mode does not emulate Request, Sample, or Event mode.

*echosw* **— returned by the graPHIGS API, fullword integer**
Current echo switch (1=NOECHO, 2=ECHO).

*strlen* **— returned by the graPHIGS API, fullword integer**
Current initial string length.

*string* **— returned by the graPHIGS API, variable length character string**
Current initial string.

*echo* **— returned by the graPHIGS API, fullword integer**
Current prompt/echo type.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Current echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*buflen* **— returned by the graPHIGS API, fullword integer**
Current string input buffer size.

*editpos* **— returned by the graPHIGS API, fullword integer**
Current cursor editing position.

*datalen* **— returned by the graPHIGS API, fullword integer**
Current string data record length.

*data* **— returned by the graPHIGS API, variable length data**
Current string data record.

### Error Codes

None

### Related Subroutines

**GPIDMO**
Set Input Device Mode

**GPINST**
Initialize String

**GPQDST**
Inquire Default String Device Data

### RCP code

201338885 (X'0C003005')

## GPQSTI - Inquire Structure Identifiers

**GPQSTI (*start*, *number*, *errind*, *totnum*, *lstrid*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPQSTI** to inquire the currently existing structure identifiers in the currently selected structure store.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*start* **— specified by user, fullword integer**
Starting member of the list of structure identifier (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of existing structure identifiers.

*lstrid* **— returned by the graPHIGS API, array of fullword integers**
List of structure identifiers.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201337097 (X'0C002909')

## GPQSTS - Inquire Structure Status

**GPQSTS (*strid*, *errind*, *flag*, *count*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

**Purpose**

Use **GPQSTS** to inquire whether the specified structure exists in the currently selected structure store, and, if it exists, how many elements are contained in the structure.

If there is no structure store currently selected, that is, if the current structure state is Structure Store Close (SSCL) or Structure Store Open (SSOP), an error is issued.

**Parameters**

*strid* **— specified by user, fullword integer**
Structure identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

*flag* **— returned by the graPHIGS API, fullword integer**
Structure status indicator (1=NON_EXISTENT, 2=EXISTENT).

*count* **— returned by the graPHIGS API, fullword integer**
Number of elements in the structure.

**Error Codes**

None

**Related Subroutines**

**GPQSTE**
Inquire Structure Existence

**RCP code**

201337105 (X'0C002911')

## GPQSTV - Inquire Structure State Value

**GPQSTV (*state*)**

**Note:** This subroutine is a State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQSTV** to inquire the structure store state of the graPHIGS API.

The structure state has one of the following values and meanings:

**1=STCL** A structure store is selected but no structure is open.

**2=STOP** A structure store is selected and a structure is open.

**3=SSCL** No structure store is attached to the graPHIGS API shell.

**4=SSOP** At least one structure store is attached to the graPHIGS API shell but no structure store is selected.

**5=NROP** The Begin Structure sequence has been started.

**Parameters**

*state* **— returned by the graPHIGS API, fullword integer**
Structure state value (1=STCL, 2=STOP, 3=SSCL, 4=SSOP, 5=NROP).

**Error Codes**

None

**Related Subroutines**

**GPBGST**
Begin Structure

**GPCLST**
Close Structure

**GPENST**
End Structure

**GPOPST**
Open Structure

**RCP code**

201336323 (X'0C002603')

## GPQSYV - Inquire System State Value

**GPQSYV (*state*)**

**Note:** This subroutine is a State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQSYV** to inquire the system state of the graPHIGS API.

**Parameters**

*state* **— returned by the graPHIGS API, fullword integer**
System state value (`1=CLOSED, 2=OPEN`).

**Error Codes**

None

**Related Subroutines**

**GPCLPH**
Close graPHIGS

**GPOPPH**
Open graPHIGS

**RCP code**

201336321 (X'0C002601')

## GPQTDF - Inquire Trimming Curve Display Facilities

**GPQTDF (*wstype*, *start*, *number*, *errind*, *order*, *totnum*, *criteria*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQTDF** to inquire the trimming curve facilities for the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string.**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available trimming curve approximation criteria (>=1).

*number* **— specified by user, fullword integer**
Number of trimming curve approximation criteria requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*order* **— returned by the graPHIGS API, fullword integer**
Maximum trimming curve order supported.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available trimming curve approximation criteria.

*criteria* **— returned by the graPHIGS API, array of fullword integers**
List of trimming curve approximation criteria (`1=WORKSTATION_DEPENDENT, 3=CONSTANT_SUBDIVISION_BETWEEN_KNOTS, 8=VARIABLE_SUBDIVISION_BETWEEN_KNOTS`).

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

**GPTNBS**
Trimmed Non-Uniform B-Spline Surface

### RCP code

201346058 (X'0C004C0A')

# GPQTMO - Inquire Available Transparency Modes

**GPQTMO (*wstype*, *start*, *number*, *errind*, *totnum*, *mode*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQTMO** to inquire the transparency mode facilities for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available transparency modes (>=1).

*number* **— specified by user, fullword integer**
Number of transparency modes requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available transparency modes.

*mode* **— returned by the graPHIGS API, array of fullword integers**
List of available transparency modes (`1=OFF, 2=PARTIAL_TRANSPARENT, 3=BLEND, 4=BLEND_ALL`).

### Error Codes

None

### Related Subroutines

**GPQRCT**
Inquire Realized Connection Type

**GPXVR**
Set Extended View Representation

**RCP code**

201339403 (X'0C00320B')

## GPQVF - Inquire View Facilities

**GPQVF (*wstype*, *errind*, *shield*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQVF** to inquire whether shielding is available for the specified workstation type.

The graPHIGS API returns data indicating whether shielding is available on the specified workstation type.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*shield* **— returned by the graPHIGS API, fullword integer**
Shielding available (`1=NOT_AVAILABLE, 2=AVAILABLE`).

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**GPXVR**
Set Extended View Representation

**RCP code**

# GPQVL - Inquire Valuator Device State

**GPQVL (*wsid*, *device*, *type*, *length*, *errind*, *mode*, *echosw*, *ivalue*, *echo*, *area*, *lovalue*, *hivalue*, *datalen*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQVL** to inquire the current state of a valuator device attached to the specified workstation.

The graPHIGS API returns the values of the specified device. The format and content of these values returned by the graPHIGS API depends on the prompt/echo type defined in the subroutine that initializes the input device.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 533 (an output parameter is not large enough for all the requested data), then the values up to the length specified are returned. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*device* **— specified by user, fullword integer**
Valuator device number (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (`1=SET`).

*length* **— specified by user, fullword integer**
Length of requested valuator data array.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**534** TYPE VALUE IS INVALID

*mode* **— returned by the graPHIGS API, fullword integer**
Current operating mode (`1=REQUEST, 2=SAMPLE, 3=EVENT, 4=APPLICATION_DEFINED`). The graPHIGS API only returns a mode of `4=APPLICATION_DEFINED` if the application set the device mode using the Set Input Device State (**GPIDMO**) subroutine and the mode does not emulate Request, Sample, or Event mode.

*echosw* **— returned by the graPHIGS API, fullword integer**
Current echo switch (1=NOECHO, 2=ECHO).

*ivalue* **— returned by the graPHIGS API, short floating-point number**
Current initial value.

*echo* **— returned by the graPHIGS API, fullword integer**
Current prompt/echo type.

*area* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Current echo area (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*lovalue* **— returned by the graPHIGS API, short floating-point number**
Current low end of range for valuator.

*hivalue* **— returned by the graPHIGS API, short floating-point number**
Current high end of range for valuator.

*datalen* **— returned by the graPHIGS API, fullword integer**
Current valuator data record length.

*data* **— returned by the graPHIGS API, variable length data**
Current valuator data record.

**Error Codes**

None

**Related Subroutines**

**GPIDMO**
Set Input Device Mode

**GPINVL**
Initialize Valuator

**GPQDVL**
Inquire Default Valuator Device Data

**RCP code**

201338886 (X'0C003006')

## GPQVR - Inquire Set of View Which Contain Root

**GPQVR (*wsid*, *strid*, *start*, *number*, *errind*, *totnum*, *view*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQVR** to inquire a list of views which contains the specified structure in the currently selected structure store as their roots.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*strid* — **specified by user, fullword integer**
Structure identifier.

*start* — **specified by user, fullword integer**
Starting member of the list of view indexes (>=1).

*number* — **specified by user, fullword integer**
Number of view indexes requested (>=0).

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**215** SPECIFIED RESOURCES DO NOT EXIST ON THE SAME NUCLEUS

**227** STRUCTURE STORE IS NOT SELECTED

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* — **returned by the graPHIGS API, fullword integer**
Total number of views which contain the root.

*view* — **returned by the graPHIGS API, array of fullword integers**
List of view indexes which contain the root.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201337098 (X'0C00290A')

## GPQWC - Inquire Workstation Category

**GPQWC (*wstype*, *errind*, *type*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQWC** to inquire the category for the specified workstation type.

The graPHIGS API returns the category of the workstation type indicating whether it is 1=OUTPUT (only), 2=INPUT (only), 3=OUTIN.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*type* **— returned by the graPHIGS API, fullword integer**
Workstation category (1=OUTPUT, 2=INPUT, 3=OUTIN).

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339651 (X'0C003303')

## GPQWD - Inquire Workstation Display Classification

**GPQWD (*wstype, errind, type*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQWD** to inquire the display category for the specified workstation type.

The graPHIGS API returns a value indicating whether the workstation type utilizes vector, raster, or other types of display technology.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*type* **— returned by the graPHIGS API, fullword integer**
Display type (1=VECTOR, 2=RASTER, 3=OTHERS).

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201339650 (X'0C003302')

## GPQWDT - Inquire Workstation Description

**GPQWDT (*wstype, id, lidata, idata, mlodata, errind, lodata, odata*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQWDT** to inquire particular information from the Workstation Description Table for the specified workstation. The *id* parameter identifies the data that the graPHIGS API returns. Some data types may require additional information in the *idata* parameter.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameters are unpredictable.

The *mlodata* parameter specifies the length of the *odata* output parameter. In general, the length of the *odata* output area must be a certain minimum length. See the parameter descriptions for the required mimimum lengths.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*id* **— specified by user, fullword integer**
Group identifier. The following identifiers are supported:

**1:** Modeling Clipping Facilities

**2:** Transparency Facilities

**3:** Data Mapping Facilities

**4:** Text Encoding Facilities

**5:** Morphing Facilities

*lidata* **— specified by user, fullword integer**
Length in bytes of input data area (>= 0).

*idata* **— specified by user, variable-length data**
Input data. Depending on the *id* parameter value specified, input data is as follows:

**1 = Modeling Clipping Facilities**
N/A (none required for this type)

**2 = Transparency Facilities**
N/A (none required for this type)

**3 = Data Mapping Facilities**
N/A (none required for this type)

**4 = Text Encoding Facilities**
N/A (none required for this type)

**5 = Morphing Facilities**
N/A (none required for this type)

*mlodata* **— specified by user, fullword integer**
Length in bytes of output data area. Depending on the *id* parameter value specified, the *minimum* length value is as follows:

**1 = Modeling Clipping Facilities**
Minimum length value = 8

**2 = Transparency Facilities**
Minimum length value = 16

**3 = Data Mapping Facilities**
Minimum length value = 12

**4 = Text Encoding Facilities**
Minimum length value = 4

**5 = Morphing Facilities**
Minimum length value = 4

*errind* **— returned by the graPHIGS API, fullword-integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** FUNCTION REQUIRES STATE STOP OR NROP (NOT STCL)

**272** GROUP IDENTIFIER IS INVALID

**509** DATA LENGTH VALUE < ZERO OR REQUIRED LENGTH

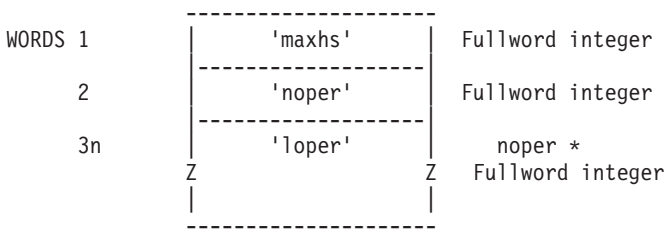
**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*lodata* **— returned by the graPHIGS API, fullword integer**
Length of available output data, in bytes.

*odata* **— returned by the graPHIGS API, Variable-length data**
Output data. Depending on the *id* parameter value specified, output data is as follows:

**1 = Modeling Clipping Facilities**
The following data is returned:

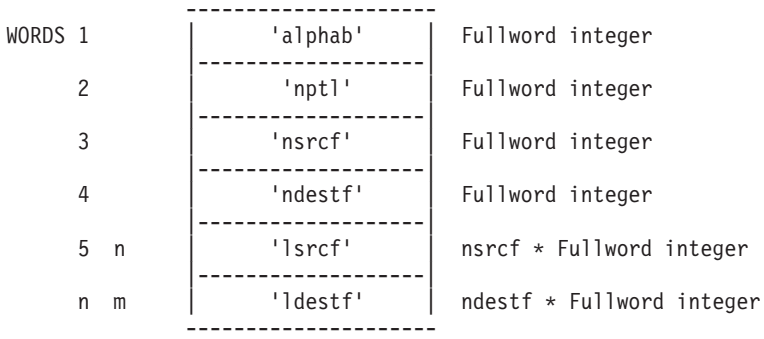



**maxhs**
Maximum number of clipping half-spaces.

**noper** Number of available modeling clipping operators.

**loper** List of available modeling clipping operators (1=REPLACE_VOLUME, 2=INTERSECT_VOLUME).

**2 = Transparency Facilities**
The following data is returned:




**alphab**
Availability of alpha buffer (1=NO, 2=YES).

**nptl** Number of partial transparency levels supported.

**nsrcf** Number of source blending functions.

**ndestf** Number of destination blending functions.

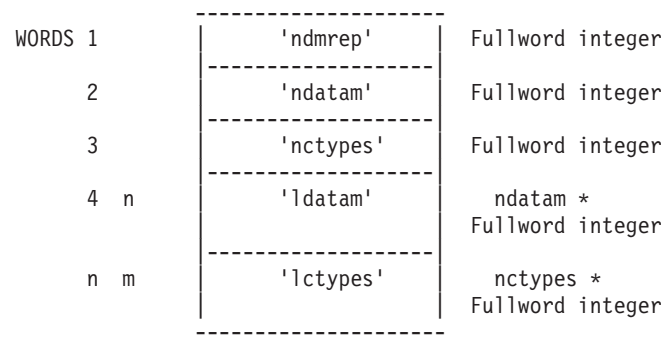
**lsrcf** List of source blending functions (1=SRCBF_ZERO, 2=SRCBF_ONE, 3=SRCBF_SRC_ALPHA, 4=SRCBF_ONE_MINUS_SRC_ALPHA, 5=SRCBF_DST_ALPHA,

6=SRCBF_ONE_MINUS_DST_ALPHA, 7=SRCBF_DST_COLOR,
8=SRCBF_ONE_MINUS_DST_COLOR, 9=SRCBF_MIN_SRC_ALPHA_ONE_MINUS_DST_ALPHA).

**ldestf** List of destination blending functions (1=DSTBF_ZERO, 2=DSTBF_ONE,
3=DSTBF_SRC_ALPHA, 4=DSTBF_ONE_MINUS_SRC_ALPHA, 5=DSTBF_DST_ALPHA,
6=DSTBF_ONE_MINUS_DST_ALPHA, 7=DSTBF_SRC_COLOR,
8=DSTBF_ONE_MINUS_SRC_COLOR).

### 3 = Data Mapping Facilities

The following data is returned:




**ndmrep**
Number of definable data mapping table entries. Entry 0 of the data mapping table may not be changed.

**ndatam**
Number of available data mapping methods.

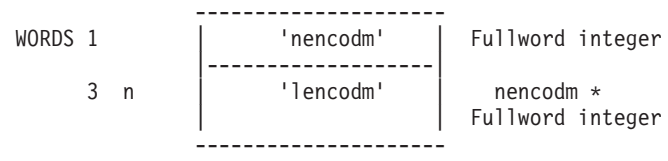
**nctypes**
Number of available data mapping color data types.

**ldatam**
List of available data mapping methods (-1=IMAGE_ARRAY, 1=DM_METHOD_COLOR,
2=SINGLE_VALUE_UNIFORM, 4=BI_VALUE_UNIFORM).

**lctypes**
List of available data mapping color data types (1=TYPE_COLOR,
2=TYPE_PACKED_RGB, 3=TYPE_COLOR_TRANS, 4=TYPE_PACKED_RGB_ALPHA).

### 4 = Text Encoding Facilities

The following data is returned:




**nencodm**
Number of available text encoding methods.

**lencodm**
List of available text encoding methods (1=UNICODE).

### 5 = Morphing Facilities

The following data is returned:

```
                    --------------------
WORDS 1             |      'nmorphv'     |  Fullword integer
                    --------------------
```

**nmorphv**
Maximum number of morphing vectors supported.

### Error Codes

None

### RCP code

201346065 (X'0C004C11')

## GPQWSA - Inquire Set of Workstations to Which Associated

**GPQWSA (*strid*, *start*, *number*, *errind*, *totnum*, *lwsid*)**

**Note:** This subroutine is a Structure State List (SSL) inquiry. For an overview, see "SSL Inquiries."

### Purpose

Use **GPQWSA** to inquire the list of workstations with which the specified structure in the currently selected structure store is associated.

This subroutine returns the number of workstations and list of workstation identifiers with which the specified structure is associated and are attached to the shell. Note that the total number of workstations returned by this subroutine may be less than the actual number of workstations with which the structure is associated, because some of the workstations may not be attached to the shell.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*strid* **— specified by user, fullword integer**
Structure identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of workstations (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**12** FUNCTION REQUIRES STATE SSSL

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of workstations to which the structure is associated.

*lwsid* **— returned by the graPHIGS API, array of fullword integers**
List of workstation identifiers to which the specified structure is associated.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201337099 (X'0C00290B')

## GPQWSU - Inquire Workstation Storage Utilization

**GPQWSU (*wsid*, *errind*, *total*, *lgblock*, *numblks*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQWSU** to inquire information about the utilization of available storage on the specified workstation. This information includes the total number of bytes available (*total*), the largest contiguous block of storage available (*lgblock*), and the number of blocks of storage available (*numblks*).

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**274** THIS FUNCTION IS NOT SUPPORTED BY THE WORKSTATION

**571** INQUIRED INFORMATION IS NOT AVAILABLE

*total* **— returned by the graPHIGS API, fullword integer**
Total number of bytes of storage currently available on this workstation.

*lgblock* **— returned by the graPHIGS API, fullword integer**
Largest contiguous block of storage available in bytes.

*numblks* **— returned by the graPHIGS API, fullword integer**
Number of blocks of storage available.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201336600 (X'0C002718')

## GPQWSV - Inquire Workstation State Value

**GPQWSV (*state*)**

**Note:** This subroutine is a State List (PSL) inquiry. For an overview, see "PSL Inquiries."

**Purpose**

Use **GPQWSV** to inquire the current workstation state of the graPHIGS API.

The graPHIGS API returns a value indicating whether any workstations are open.

**Parameters**

*state* **— returned by the graPHIGS API, fullword integer**
Workstation state value (`1=CLOSED, 2=OPEN, 3=SELECTED`).

**Error Codes**

None

**Related Subroutines**

**GPBGTR**
Begin Traversal

**GPCLWS**
Close Workstation

**GPENTR**
End Traversal

**GPOPWS**
Open Workstation

**RCP code**

# GPQWSX - Inquire Workstation Transformation

**GPQWSX (*wsid, errind, state, rwindow, cwindow, rviewpt, cviewpt*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQWSX** to inquire the current and requested workstation transformation values of a specified workstation.

If your application has not updated the workstation, then the graPHIGS API returns a value of `2=PENDING`. In this case, the requested values reflect the settings established in the application with the Set Workstation Transformation 2 (**GPWSX2**) subroutine. The current values reflect the workstation's current transformation values. As soon as the workstation is updated, the requested and current values are the same and the state is `1=NOT_PENDING`. The values returned by the graPHIGS API are the window and viewport definitions.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*state* **— returned by the graPHIGS API, fullword integer**
Workstation transformation update state (`1=NOT_PENDING, 2=PENDING`).

*rwindow* **— returned by the graPHIGS API, 6 short floating-point numbers (NPC)**
Requested workstation window (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*cwindow* **— returned by the graPHIGS API, 6 short floating-point numbers (NPC)**
Current workstation window (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*rviewpt* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Requested workstation viewport (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*cviewpt* **— returned by the graPHIGS API, 6 short floating-point numbers (DC)**
Current workstation viewport (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

### Error Codes

None

**Related Subroutines**

**GPWSX2**
Set Workstation Transformation 2

**GPWSX3**
Set Workstation Transformation 3

**RCP code**

201336842 (X'0C00280A')

## GPQWTN - Inquire List of Available Workstation Types on Nucleus

**GPQWTN (*ncid*, *start*, *number*, *errind*, *maxa*, *totnum*, *wstype*)**

**Note:** This subroutine is a Nucleus Description Table (NDT) inquiry. For an overview, see "NDT Inquiries."

**Purpose**

Use **GPQWTN** to inquire a list of available generic workstation types on the specified nucleus.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*start* **— specified by user, fullword integer**
Starting member in the list of available generic workstation types (>=1).

*number* **— specified by user, fullword integer**
Number of generic workstation types requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*maxa* **— returned by the graPHIGS API, fullword integer**
Maximum number of workstations to which a structure store can be simultaneously associated.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available generic workstation types.

*wstype* **— returned by the graPHIGS API, array of 8-byte character strings**
A list of generic workstation types available on the specified nucleus. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPCRWS**
Create Workstation

**RCP code**

201336332 (X'0C00260C')

## GPQWTO - Inquire Workstation Type and Options

**GPQWTO (*wsid*, *ilen*, *errind*, *wstype*, *olen*, *options*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQWTO** to inquire the generic workstation type and workstation creation options that were used to create the specified workstation (*wsid*).

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 536 (the actual creation workstation options are greater than the length of the area provided), then only the actual length (*olen*) of the options is returned. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ilen* **— specified by user, fullword integer**
Length of the area provided to contain the creation workstation options.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

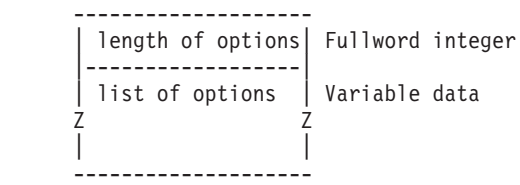
**536** INQUIRY DATA EXCEEDS AREA. LENGTH OF REQUIRED AREA RETURNED

*wstype* **— returned by the graPHIGS API, 8-byte character string**
Generic workstation type.

*olen* **— returned by the graPHIGS API, fullword integer**
Actual length of the creation workstation options. If the area provided is large enough to contain the options, then the graPHIGS API returns the actual length of the options in this field. This actual

length is the same as the length provided in the first word of the *options* parameter. If the area provided is not large enough to contain the options, then *olen* contains the actual length required and the graPHIGS API sets the error indicator to error 536.

*options* **— returned by the graPHIGS API, variable data**
Workstation creation options. This parameter has the following format:




This format is the same format specified by the Create Workstation (**GPCRWS**) subroutine. See GPCRWS - Create Workstation for this format.

### Error Codes

None

### Related Subroutines

**GPCRWS**
Create Workstation

### RCP code

201336854 (X'0C002816')

## GPQXAF - Inquire Extended Annotation Font Characteristics

**GPQXAF (*wsid*, *csid*, *font*, *start*, *num*, *errind*, *prec*, *nomh*, *nahsf*, *lahsf*, *lnfac*, *lmnfac*, *lmxfac*, *proportional*, *top*, *bottom*, *nomaspect*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

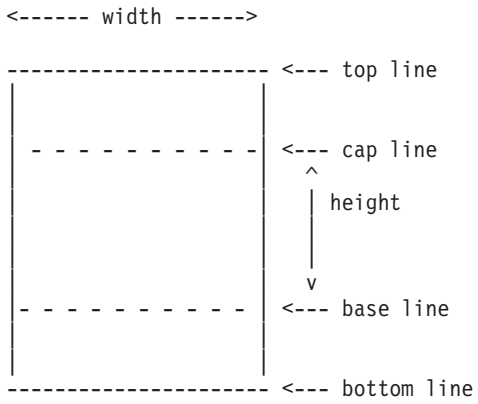
### Purpose

Use **GPQXAF** to inquire the annotation text capabilities for the requested character set and font on the specified workstation.

This information consists of the highest text precision for this *csid*/*font* that is supported by the specified workstation (*prec*), the nominal character height in Device Coordinates (DC) (*nomh*), the number of supported annotation height scale factors (*nahsf*), a list indicating the exact supported height scale factors (*lahsf*), and the number of character expansion factors (*lnfac*), minimum expansion (*lmnfac*) and maximum (*lmxfac*) expansion factors corresponding to each height scale factor, whether the font is fixed or proportional (*proportional*), and ratios describing the character box (*top*, *bottom*, and *nomaspect*).

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

Annotation text capabilities are defined in terms of the following character box description:




Height is defined as the distance between the base line and the cap line.

> **Note:** The definition of height is different from that used by the Inquire Annotation Font Characteristics (**GPQAFC**) subroutine.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*csid* — **specified by user, fullword integer**
Character set identifier.

*font* — **specified by user, fullword integer**
Font identifier (>=1).

*start* — **specified by user, fullword integer**
Starting member of the list of supported height scale factors (>=1).

*num* — **specified by user, fullword integer**
Number of list elements requested (>=0). This number refers to the number of list elements that are returned.

> **Note:** The four output arrays must be large enough to hold the requested number of elements.

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**75** TEXT FONT VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**542** CHARACTER SET IDENTIFIER IS INVALID

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**562** CHARACTER SET/FONT COMBINATION IS NOT AVAILABLE FOR ANNOTATION

**571** INQUIRED INFORMATION IS NOT AVAILABLE

*prec* **— returned by the graPHIGS API, fullword integer**
Highest available precision for the corresponding *csid*/*font* (1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC). If the highest precision supported is 3=STROKE_PREC, then the following parameters refer to the workstation's capabilities for character precision with annotation text.

*nomh* **— returned by the graPHIGS API, short floating-point number (DC)**
The base to cap size, (cap - base), in Device Coordinates (DC) for the nominal height font.

*nahsf* **— returned by the graPHIGS API, fullword integer**
The total number of annotation sizes that are supported. This also defines the number of entries in each of the next four parameter lists except when this values is zero. A value of zero means that a continuous range of annotation sizes is supported and each of the lists contains two entries.

*lahsf* **— returned by the graPHIGS API, array of short floating-point numbers**
List of the annotation height scale factors supported. If the device supports a continuous range of annotation heights, then the first entry is 1.0 [default] 10 (-30) and the second is 9.9 [default] 10 (30).

> **Note:** This list has one entry for each annotation size supported. Its length is determined by the *nahsf* field except when that contains 0. In this case, there are two entries in this list. The first entry contains information for the minimum annotation height scale factor. The second contains information for the maximum.

*lnfac* **— returned by the graPHIGS API, array of fullword integers**
List of numbers of expansion factors that are supported. There is one entry for each of the supported annotation height scale factors. Zero means that a continuous range of expansion factors is supported for the font size.

> **Note:** The length of this list is determined by the *nahsf* field except when that contains 0. In this case, there are two entries in this list. The first entry contains information for the minimum annotation height scale factor. The second contains information for the maximum.

*lmnfac* **— returned by the graPHIGS API, array of short floating-point numbers**
List of minimum expansion factors that are supported. There is one entry for each of the supported annotation height scale factors. For devices which support all possible expansion factors, this parameter is a very small positive number (e.g.10 (-30)).

> **Note:** The length of this list is determined by the *nahsf* field except when that contains 0. In this case, there are two entries in this list. The first entry contains information for the minimum annotation height scale factor. The second contains information for the maximum.

> **Note:** The expansion factor is computed as follows: expf = (W/H) / *nomaspect* where 'W' and 'H' are the width and the height for that font.

*lmxfac* **— returned by the graPHIGS API, array of short floating-point numbers.**
List of maximum expansion factors that are supported. There is one entry for each of the supported annotation height scale factors.

> **Note:** For devices which support all possible expansion factors, this parameter contains a very large positive number (e.g. 9.9 [default] 10 (30)).

> **Note:** The length of this list is determined by the *nahsf* field except when this contains 0. In this case, there are two entries in this list. The first entry contains information for the minimum annotation height scale factor. The second contains information for the maximum.

> **Note:** The expansion factor is computed as follows: expf = (W/H) / *nomaspect* where 'W' and 'H' are the width and height for that font.

*proportional* **— returned by the graPHIGS API, fullword integer**
Indicates whether the font definition contains per character positioning information so that proportional spacing can be performed (1=FIXED, 2=PROPORTIONAL).

*top* **— returned by the graPHIGS API, short floating-point number**
Top ratio, (top-cap)/ *nomh*, is returned in this field (>=0). This is a constant for all sizes of a font.

*bottom* **— returned by the graPHIGS API, short floating-point number**
Bottom ratio, (base-bot)/*nomh*, is returned in this field (>=0). This is a constant for all sizes of a font.

*nomaspect* **— returned by the graPHIGS API, short floating-point number**
Aspect ratio, (width/height), of the nominal size font.

### Error Codes

None

### Related Subroutines

**GPAH** Set Annotation Height

**GPAHSC**
Set Annotation Height Scale Factor

**GPQAFC**
Inquire Annotation Font Characteristics

### RCP code

201336591 (X'0C00270F')

## GPQXCF - Inquire Extended Color Facilities

**GPQXCF (*wstype*, *start*, *number*, *errind*, *charact*, *nmax*, *lmax*, *totnum*, *model*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

### Purpose

Use **GPQXCF** to inquire the extended color facilities for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of available color models (>=1).

*number* **— specified by user, fullword integer**
Number of color models requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*charact* **— returned by the graPHIGS API, fullword integer**
Workstation color table characteristic (`1=NEITHER_MODIFIABLE, 2=ONLY_DISPLAY_COLOR_TABLE_MODIFIABLE, 3=ONLY_RENDERING_COLOR_TABLE_MODIFIABLE, 4=BOTH_MODIFIABLE`).

*nmax* **— returned by the graPHIGS API, fullword integer**
Maximum number of definable image color tables.

*lmax* **— returned by the graPHIGS API, fullword integer**
Maximum length of definable image color tables.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of available color models for image color tables.

*model* **— returned by the graPHIGS API, array of fullword integers**
List of available color models for image color tables (`1=RGB, 2=HSV, 3=CMY, 4=CIELUV`).

**Error Codes**

None

**Related Subroutines**

**GPDFI**
Define Image

**GPQRCT**
Inquire Realized Connection Type

**GPXCR**
Set Extended Color Representation

**RCP code**

201346056 (X'0C004C08')

## GPQXCR - Inquire Extended Color Representation

**GPQXCR (*wsid*, *ctid*, *start*, *number*, *type*, *errind*, *color*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQXCR** to inquire the current color values in the specified color table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*ctid* **— specified by user, fullword integer**
Color table identifier of a color table that exists on the workstation (`-1=DISPLAY_COLOR_TABLE, 0=RENDERING_COLOR_TABLE`).

*start* **— specified by user, fullword integer**
Index specifying an entry into the specified color table to start returning the requested color values (>=0).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*type* **— specified by user, fullword integer**
Type of returned values (`1=SET`).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**284** COLOR TABLE IDENTIFIER DOES NOT EXIST

**534** TYPE VALUE IS INVALID

**539** REQUESTED NUMBER < ZERO

**544** START VALUE < ZERO

**551** START VALUE EXCEEDS COLOR TABLE SIZE

**571** INQUIRED INFORMATION IS NOT AVAILABLE

*color* **— returned by the graPHIGS API, array of three short floating-point numbers**
Array of color values ordered by row. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPQCCH**
Inquire Color Table Characteristics

**GPQCID**
Inquire List of Color Table Identifiers

**GPXCR**
Set Extended Color Representation

**RCP code**

201339153 (X'0C003111')

## GPQXER - Inquire Extended Edge Representation

**GPQXER (*wsid*, *index*, *type*, *number*, *ids*, *errind*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQXER** to inquire the current value of one or more fields in the specified edge bundle table entry of the specified workstation's edge bundle table.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. The output parameter must be large enough to store all requested data. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Edge bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned value (1=SET).

*number* **— specified by user, fullword integer**
Number of group identifiers requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers requested.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**534** TYPE VALUE IS INVALID

*data* — **returned by the graPHIGS API, variable data**
Data array containing the values in the requested groups.

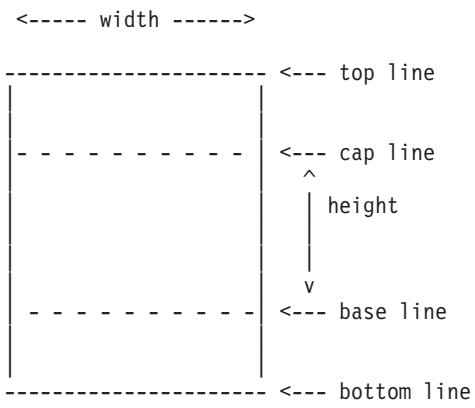
The value that may be set for each field is expressed in the data format listed below:

**Group Identifier 1 - Edge flag**
A fullword integer (1=OFF, 2=ON, 3=GEOMETRY_ONLY).

**Group Identifier 2 - Line type table index**
A fullword integer (>=1). Specifies an index into the workstation's edge line type table. The table size and specified entries supported are workstation dependent. The default edge line type table for supported entries is defined with the following line types: (1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE).

**Group Identifier 3 - Edge linewidth scale factor**
A short floating-point number.

**Group Identifier 4 - Edge color**
Four fullwords of data with either of the following two formats:




### Error Codes

None

### Related Subroutines

**GPXER**
Set Extended Edge Representation

### RCP code

201339154 (X'0C003112')

## GPQXIR - Inquire Extended Interior Representation

**GPQXIR (*wsid*, *index*, *type*, *number*, *ids*, *errind*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQXIR** to inquire the current values of one or more fields in the specified interior bundle table entry of the specified workstation's interior bundle table.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. The output parameter must be large enough to store all requested data. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Interior bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned value (1=SET).

*number* **— specified by user, fullword integer**
Number of group identifiers requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**534** TYPE VALUE IS INVALID

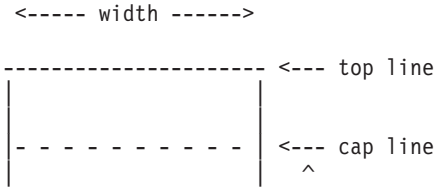
*data* **— returned by the graPHIGS API, variable data**
Data array containing the values in the requested groups.

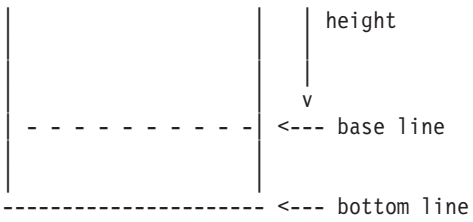
The value that may be set for each field is expressed in the data format listed below:

**Group Identifier 1 - Interior style**
A fullword integer (1=HOLLOW, 2=SOLID, 3=PATTERN, 4=HATCH, 5=EMPTY).

**Group Identifier 2 - Interior style index**
A fullword integer (>=1).

**Group Identifier 3 - Interior color**
Four fullwords of data with either of the following two formats:






### Error Codes

None

### Related Subroutines

**GPXIR**
Set Extended Interior Representation

### RCP code

201339155 (X'0C003113')

## GPQXLR - Inquire Extended Polyline Representation

**GPQXLR (*wsid*, *index*, *type*, *number*, *ids*, *errind*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQXLR** to inquire the current values of one or more fields in the specified polyline bundle table entry in the specified workstation's polyline bundle table.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. The output parameter must be large enough to store all requested data. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Polyline bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned value (1=SET).

*number* **— specified by user, fullword integer**
Number of group identifiers requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers requested.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**534** TYPE VALUE IS INVALID

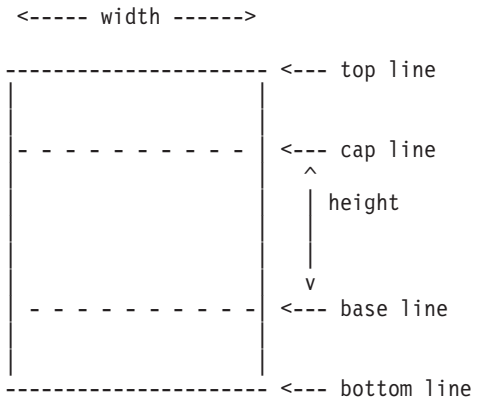
*data* **— returned by the graPHIGS API, variable data**
Data array containing the values in the requested groups.

The value that may be set for each field is expressed in the data format listed below:

**Group Identifier 1 - Line type table index**
A fullword integer (>=1). Specifies an index into the workstation's line type table. The table size and specific entries supported is workstation dependent. The default line type table for supported entries is defined with the following line types: (1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE).

**Group Identifier 2 - Linewidth scale factor**
A short floating point number.

**Group Identifier 3 - Polyline color**
Four fullwords of data with either of the following two formats:




**Error Codes**

None

**Related Subroutines**

**GPXPLR**
Set Extended Polyline Representation

**RCP code**

201339156 (X'0C003114')

# GPQXMR - Inquire Extended Polymarker Representation

**GPQXMR (*wsid*, *index*, *type*, *number*, *ids*, *errind*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

**Purpose**

Use **GPQXMR** to inquire the current values of one or more fields in the specified polymarker bundle table entry in the specified workstation's polymarker bundle table.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. The output parameter must be large enough to store all requested data. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Polymarker bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned value (1=SET).

*number* **— specified by user, fullword integer**
Number of group identifiers requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers requested.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**534** TYPE VALUE IS INVALID

*data* **— returned by the graPHIGS API, variable data**
Data array containing the values in the requested groups.

**Group Identifier 1 - Marker type table index**
A fullword integer (>=1). Specifies an index into the workstation's marker type table. The

size and specific entries supported are workstation dependent. The default marker type table for supported entries is defined with the following marker types: (1=DOT, 2=PLUS_SIGN, 3=ASTERISK, 4=CIRCLE, 5=DIAGONAL_CROSS, 6-n=ASTERISK).

**Group Identifier 2 - Marker size scale factor**
A short floating-point number.

**Group Identifier 3 - Polymarker color**
Four fullwords of data with either of the following two formats:

```
   indexed format                        direct format
    -------------                         -------------
  0 |           1| fullword integer     0 |           2| fullword integer
    |------------|                       |------------|
  4 |color index | fullword integer     4 |component 1| short floating-point
number
    |------------|                       |------------|
  8 |   reserved | fullword integer     8 |component 2| short floating-point
number
    |------------|                       |------------|
 12 |   reserved | fullword integer    12 |component 3| short floating-point
number
    -------------                         -------------
```

### Error Codes

None

### Related Subroutines

**GPXPMR**
Set Extended Polymarker Representation

### RCP code

201339157 (X'0C003115')

## GPQXTR - Inquire Extended Text Representation

**GPQXTR (*wsid*, *index*, *type*, *number*, *ids*, *errind*, *data*)**

**Note:** This subroutine is a Workstation State List (WSL) inquiry. For an overview, see "WSL Inquiries."

### Purpose

Use **GPQXTR** to inquire the current value of one or more fields in the specified text bundle table entry in the specified workstation's text bundle table.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. The output parameter must be large enough to store all requested data. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Text bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned value (1=SET).

*number* **— specified by user, fullword integer**
Number of group identifiers requested (>=1).

*ids* **— specified by user, array of fullword integers**
A list of group identifiers requested.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**272** GROUP IDENTIFIER IS INVALID

**273** NUMBER OF GROUP IDENTIFIERS < ONE

**534** TYPE VALUE IS INVALID

*data* **— returned by the graPHIGS API, variable data**
Data array containing the values in the requested groups.

Each field that may be set in the interior bundle table entry is identified by a group identifier. The value that may be set for each field is expressed in the data format listed below:

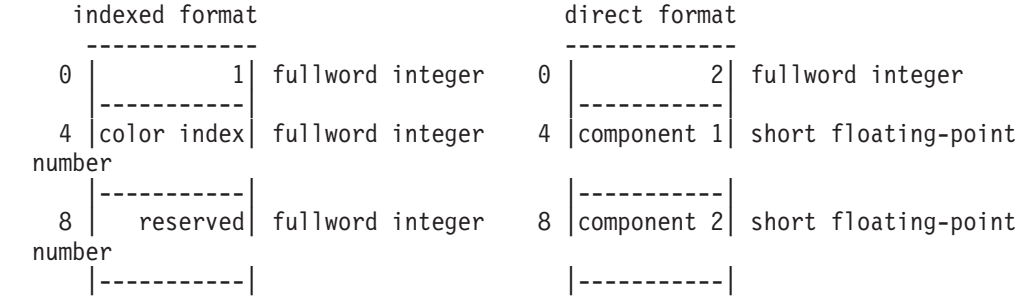
**Group Identifier 1 - Text font**
A fullword integer (1-255).

**Group Identifier 2 - Text precision**
A fullword integer (1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC).

**Group Identifier 3 - Character expansion factor**
A short floating-point value (>=0).

**Group Identifier 4 - Character spacing**
A short floating-point value.

**Group Identifier 5 - Text color**
Four fullwords of data with either of the following two formats:



```
 12 |   reserved| fullword integer   12 |component 3| short floating-point
number
    ------------                       ------------
```

**Error Codes**

None

**Related Subroutines**

**GPXTXR**
Set Extended Text Representation

**RCP code**

201339158 (X'0C003116')

## GPQXTX - Inquire Extended Text Facilities

**GPQXTX (*wstype*, *errind*, *npred*, *filled*, *proportional*)**

**Note:** This subroutine is a Workstation Description Table (WDT) inquiry. For an overview, see "WDT Inquiries."

**Purpose**

Use **GPQXTX** to inquire the extended text facilities for the specified workstation.

If the information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameter. If the inquired information is unavailable, then the error indicator (*errind*) contains the error number indicating the reason, and the values returned in the output parameter are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined text bundle table entries.

*filled* **— returned by the graPHIGS API, fullword integer**
Filled font support (1=NOT_SUPPORTED, 2=SUPPORTED).

*proportional* **— returned by the graPHIGS API, fullword integer**
Proportional font support (1=NOT_SUPPORTED, 2=SUPPORTED).

**Error Codes**

None

**Related Subroutines**

**GPCHPM**
Set Character Positioning Mode

**GPMTR**
Set Marker Type Representation

**GPQRCT**
Inquire Realized Connection Type

**GPTX2**
Geometric Text 2

**GPTX3**
Geometric Text 3

**RCP code**

201346064 (X'0C004C10')

## GPRAS - Retrieve Ancestors to Structures

**GPRAS (*arid*, *strid*, *order*, *depth*, *start*, *number*, *buflen*, *errind*, *actnum*, *actlen*, *totnum*, *data*, *termcond*)**

**Note:** This subroutine is an Archive inquiry. For an overview, see "Archive Inquiries."

**Purpose**

Use **GPRAS** to retrieve the ancestral paths of a specified structure from the specified open archive file.

A path of ancestors of a structure *S* is a list of ordered pairs ((*A*1, *E*1), (*A*2,*E*2),..., (*Am*,*Em*), (*S*,0)) where each ordered pair consists of an identifier of a structure (*Ax*) that is an ancestor of the specified structure (*S*) and the position of an execute structure-type element (*Ex*) that references the next structure in the path. Ancestor structure *A*1 is the top of the path (e.g., it is not referenced by any other structure) and *S* is the bottom of the path.

The path order and path depth determine the portion of each path that the graPHIGS API returns. The path depth determines the maximum number of ordered pairs returned in any one path. Specifying a path depth of zero returns each path in its entirety. When truncation occurs, the path order determines whether the graPHIGS API returns the head or tail portion of the path. This truncation can result in two or more portions of paths having the same set of element references. The graPHIGS API returns only one such portion so that all the returned path portions are distinct.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*arid* **— specified by user, fullword integer**
Archive file identifier.

*strid* — **specified by user, fullword integer**
Structure identifier.

*order* — **specified by user, fullword integer**
Path order (1=TOPFIRST, 2=BOTTOMFIRST).

*depth* — **specified by user, fullword integer**
Path depth (>=0).

*start* — **specified by user, fullword integer**
Starting member of the list of paths (>=1).

*number* — **specified by user, fullword integer**
Number of paths requested (>=0).

*buflen* — **specified by user, fullword integer**
Length, in bytes, of the data area specified by the *data* parameter into which the graPHIGS API returns the paths.

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**7** FUNCTION REQUIRES STATE AROP

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**552** PATH ORDER IS INVALID

**558** PATH DEPTH < ZERO

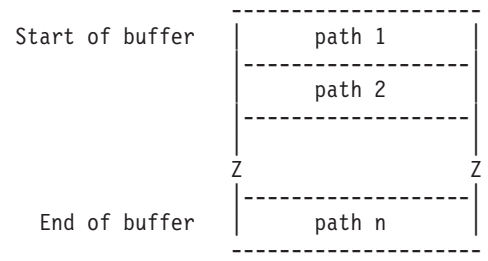
**577** BUFFER LENGTH IS < ZERO

*actnum* — **returned by the graPHIGS API, fullword integer**
Total number of paths returned.

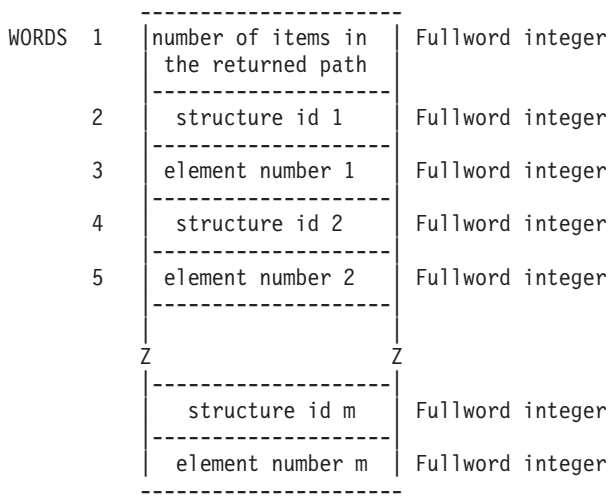
*actlen* — **returned by the graPHIGS API, fullword integer**
Total length, in bytes, of the paths that the graPHIGS API returns in the *data* parameter.

*totnum* — **returned by the graPHIGS API, fullword integer**
Total number of complete paths available for the specified structure identifier.

*data* — **returned by the graPHIGS API, variable data**
The data buffer into which the graPHIGS API returns the paths. The format of the data is as follows:




where each path has the following format:




*termcond* **— returned by the graPHIGS API, fullword integer**

Termination condition. The graPHIGS API terminated the list of paths due to one of the following reasons:

**1- Count Exhausted**

The graPHIGS API returned the requested number of paths.

**2- Buffer Overflow**

The graPHIGS API could not return the requested number of paths because there was not enough room in the area provided. *actnum* contains the actual number returned.

**3- End of the Paths**

No more paths exist. This condition supercedes the Count Exhausted condition (if that condition was in effect). The total number of paths returned may or may not be equal to the number of returned paths you requested. Check *actnum* to find the actual number of paths returned.

**4- Large Path**

The next path would not fit into the inbound buffer between the nucleus and the shell. *actnum* contains the number of paths excluding the one that would not fit. This number of paths is also in the *data* parameter.

### Error Codes

None

### Related Subroutines

**GPRDS**

Retrieve Descendants to Structures

### RCP code

201347594 (X'0C00520A')

## GPRDS - Retrieve Descendants to Structures

**GPRDS (*arid*, *strid*, *order*, *depth*, *start*, *number*, *buflen*, *errind*, *actnum*, *actlen*, *totnum*, *data*, *termcond*)**

**Note:** This subroutine is an Archive inquiry. For an overview, see "Archive Inquiries."

**Purpose**

Use **GPRDS** to retrieve the descendant paths of a specified structure from the specified open archive file.

A path of descendants of a structure *S* is a list of ordered pairs ((*S*, *E*0), (*D*1, *E*1) (*D*2, *E*2),..., (*Dn*,0)) where each ordered pair consists of an identifier of a structure (*Dx*) that is a descendant of the specified structure (*S*) and the position of an execute structure-type element (*Ex*) that references the next structure in the path. The specified structure *S* is the top of the path and the descendant structure *Dn* is the bottom of the path (e.g., it does not reference any other structure).

The path order and path depth determine the portion of each path that the graPHIGS API returns. The path depth determines the maximum number of ordered pairs returned in any one path. Specifying a path depth of zero returns each path in its entirety. When truncation occurs, the path order determines whether the graPHIGS API returns the head or tail portion of the path. This truncation can result in two or more portions of paths having the same set of element references. The graPHIGS API returns only one such portion so that all the returned path portions are distinct.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*arid* **— specified by user, fullword integer**
Archive file identifier.

*strid* **— specified by user, fullword integer**
Structure identifier.

*order* **— specified by user, fullword integer**
Path order (`1=TOPFIRST, 2=BOTTOMFIRST`).

*depth* **— specified by user, fullword integer**
Path depth (>=0).

*start* **— specified by user, fullword integer**
Starting member of the list of paths (>=1).

*number* **— specified by user, fullword integer**
Number of paths requested (>=0).

*buflen* **— specified by user, fullword integer**
Length, in bytes, of the data area specified by the *data* parameter into which the graPHIGS API returns the paths.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**7** FUNCTION REQUIRES STATE AROP

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**552** PATH ORDER IS INVALID

**558** PATH DEPTH < ZERO

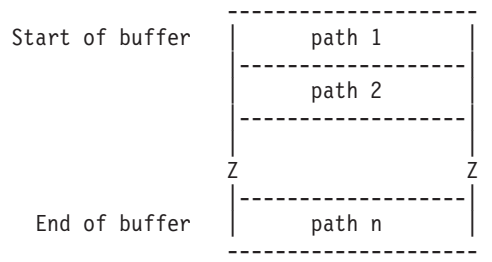
**577** BUFFER LENGTH IS < ZERO

*actnum* **— returned by the graPHIGS API, fullword integer**
Total number of paths returned.

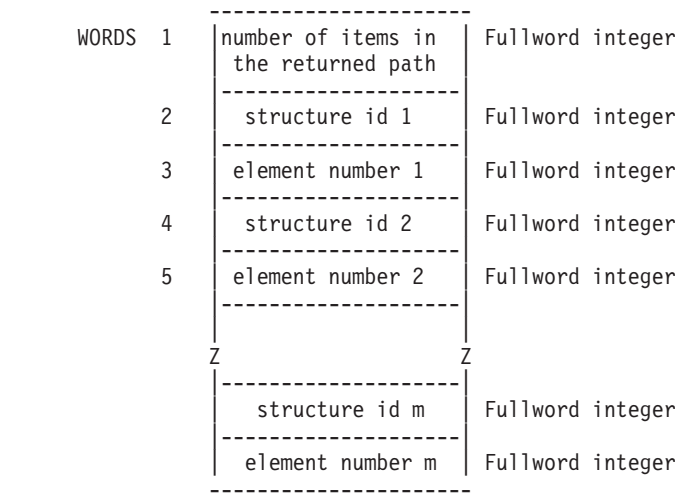
*actlen* **— returned by the graPHIGS API, fullword integer**
Total length, in bytes, of the paths that the graPHIGS API returns in the *data* parameter.

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of complete paths available for the specified structure identifier.

*data* **— returned by the graPHIGS API, variable data**
The data buffer into which the graPHIGS API returns the paths. The format of the data is as follows:




where each path has the following format:




*termcond* **— returned by the graPHIGS API, fullword integer**
Termination condition. The graPHIGS API terminated the list of paths due to one of the following reasons:

**1- Count Exhausted**
The graPHIGS API returned the requested number of paths.

**2- Buffer Overflow**
The graPHIGS API could not return the requested number of paths because there was not enough room in the area provided. *actnum* contains the actual number returned.

**3- End of the Paths**
No more paths exist. This condition supercedes the Count Exhausted condition (if that

condition was in effect). The total number of paths returned may or may not be equal to the number of returned paths you requested. Check *actnum* to find the actual number of paths returned.

**4- Large Path**
The next path would not fit into the inbound buffer between the nucleus and the shell. *actnum* contains the number of paths excluding the one that would not fit. This number of paths is also in the *data* parameter.

### Error Codes

None

### Related Subroutines

**GPRAS**
Retrieve Ancestors to Structures

### RCP code

201347595 (X'0C00520B')

## GPRISN - Retrieve Identifiers of Structures in Network

**GPRISN (*arid*, *strid*, *start*, *number*, *errind*, *totnum*, *istrid*)**

**Note:** This subroutine is an Archive inquiry. For an overview, see "Archive Inquiries."

### Purpose

Use **GPRISN** to retrieve a list of structure identifiers in a specified structure network in the specified open archive file.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*arid* **— specified by user, fullword integer**
Archive file identifier.

*strid* **— specified by user, fullword integer**
Structure identifier of the root structure.

*start* **— specified by user, fullword integer**
Starting member of the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**7** FUNCTION REQUIRES STATE AROP

**122** STRUCTURE IDENTIFIER DOES NOT EXIST

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of structures in the specified structure network.

*istrid* **— returned by the graPHIGS API, fullword integer**
List of structure identifiers. In a complete list of structure identifiers in the network, the first structure identifier is always the root structure. No structure identifiers are duplicated in the list (e.g., if a structure is referenced in the network more than once, then it only appears once in the list).

**Error Codes**

None

**Related Subroutines**

**GPRSTI**
Retrieve Structure Identifiers

**RCP code**

201347598 (X'0C00520E')

## GPRSTI - Retrieve Structure Identifiers

**GPRSTI (*arid*, *start*, *number*, *errind*, *totnum*, *istrid*)**

**Note:** This subroutine is an Archive inquiry. For an overview, see "Archive Inquiries."

**Purpose**

Use **GPRSTI** to retrieve a list of structure identifiers in the specified open archive file.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*arid* **— specified by user, fullword integer**
Archive file identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of structure identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of entries requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**7** FUNCTION REQUIRES STATE AROP

**220** SPECIFIED ARCHIVE FILE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of structures in the specified archive file.

*istrid* **— returned by the graPHIGS API, fullword integer**
List of structure identifiers.

**Error Codes**

None

**Related Subroutines**

None

**RCP code**

201347598 (X'0C00520E')

# Chapter 18. Compatibility Subroutines

This chapter contains the subroutines that have been replaced by an extended version of the subroutine. The new extended subroutine performs the same function as was supported in prior releases, plus it does some additional function.

The subroutines found in this chapter will continue to be supported for compatibility; however, it is recommended that the new applications use the new extended version of the subroutine and that existing applications migrate over time to the new extended subroutines.

Under each subroutine in this chapter, there is a "Related Subroutine" section which contains the name of its new extended subroutine.

## GPCR - Set Color Representation

**GPCR (*wsid*, *index*, *number*, *colors*)**

### Purpose

Use **GPCR** to set the specified color values starting at the specified color table entry of the workstation's default color table. The color values are interpreted according to the workstation's current color model.

The workstation's default color table may be either its rendering color table or its display color table. If its display color table is modifiable, then it is the default, otherwise the rendering color table is modifiable, and thus, the default. (Use the Inquire Color Facilities [**GPQCF**] subroutine [page GPQCF - Inquire Color Facilities] to determine the characteristics of the workstation's color table).

If the color model is Hue-Saturation-Value (HSV), the first color component (hue) is specified as a fraction of the total range available (that is, zero to one is used to represent zero degrees to 360 degrees). If the second color component is zero, then the first color component is ignored.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Entry at which to begin setting the default color table (>=0).

*number* **— specified by user, fullword integer**
Number of entries to set (>=0).

*colors* **— specified by user, array of 3 short floating-point numbers**
The color table values to be set into the default color table (0<=component<=1).

The array is assumed to be in row order, such as `RED1, GREEN1, BLUE1, RED2, GREEN2, BLUE2,` etc... .

### Error Codes

| | |
|---|---|
| **25** | SPECIFIED WORKSTATION DOES NOT EXIST |
| **35** | WORKSTATION HAS ONLY INPUT CAPABILITIES |
| **49** | COLOR INDEX EXCEEDS WORKSTATION TABLE CAPACITY |
| **92** | COLOR INDEX < ZERO |

| | |
|---|---|
| **96** | COLOR PARAMETER OUT OF RANGE FOR CURRENT COLOR MODEL |
| **517** | NUMBER OF INDEXES < ONE |

### Related Subroutines

**GPXCR** Set Extended Color Representation

### RCP code

201329414 (X'0C000B06')

## GPEPLB - Set Element Pointer at Label

**GPEPLB (*label*)**

### Purpose

Use **GPEPLB** to set the element pointer to the specified label element within the open structure.

Starting at the position following the element pointer, the graPHIGS API searches for the first occurrence of the specified *label*. If the end of the structure is reached, the system continues the search at element 1. If the specified *label* is not found, an error is generated.

> **Note:** This subroutine is not processed by the nucleus right away. It is processed by the nucleus when the next subroutine call is issued that forces requests to be sent to the nucleus. See "Controlling the System" section in the *The graPHIGS Programming Interface: Understanding Concepts* for a discussion on when requests are sent to the nucleus for processing.

### Parameters

*label* — **specified by user, fullword integer**
Label identifier.

### Error Codes

**4** FUNCTION REQUIRES STATE STOP

**130** LABEL IDENTIFIER CANNOT BE FOUND IN THE OPEN STRUCTURE

### Related Subroutines

**GPEPLG**
Generalized Set Element Pointer at Label

**GPINLB**
Insert Label

### RCP code

201332227 (X'0C001603')

## GPEPPK - Set Element Pointer at Pick Identifier

**GPEPPK (*pickid*)**

### Purpose

Use **GPEPPK** to set the element pointer to the specified Set Pick Identifier structure element within the open structure.

Starting with the position following the element pointer, the graPHIGS API searches for the first occurrence of the specified *pickid*. If the end of the structure is reached, the system continues the search at element 1. If the specified *pickid* is not found, an error is generated.

> **Note:** This subroutine is not processed by the nucleus right away. It is processed by the nucleus when the next subroutine call is issued that forces requests to be sent to the nucleus. See *The graPHIGS Programming Interface: Understanding Concepts* for a discussion on when requests are sent to the nucleus for processing.

### Parameters

*pickid* **— specified by user, fullword integer**
Pick identifier.

### Error Codes

**4** FUNCTION REQUIRES STATE STOP

**566** PICK IDENTIFIER DOES NOT EXIST IN THE OPEN STRUCTURE

### Related Subroutines

**GPEPPG**
Generalized Set Element Pointer at Pick Identifier

**GPPKID**
Set Pick Identifier

### RCP code

201332228 (X'0C001604')

## GPER - Set Edge Representation

**GPER (*wsid*, *index*, *edgefg*, *edgelt*, *edgesf*, *ecol*)**

### Purpose

Use **GPER** to set the given attribute values in the specified entry of the specified workstation.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Edge bundle table index (>=1).

*edgefg* **— specified by user, fullword integer**
Edge flag (`1=OFF, 2=ON, 3=GEOMETRY_ONLY`).

*edgelt* **— specified by user, fullword integer**
Edge line type table index. Specifies an index into the workstation's edge line type table. The table size and specific entries supported are workstation dependent. Use the Inquire Edge Facilities (**GPQEF**) subroutine to locate the supported line types on your workstation. The default edge line type table for supported entries is defined with the following line types: (1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE. Any entry may be changed by the Set Linetype Representation (**GPLTR**) subroutine except entry 1).

*edgesf* **— specified by user, short floating-point number.**
Edge scale factor, specified as a fraction of the nominal width of a line. A line width scale factor of 1.0, which is the default, generates a nominal size line on any workstation. Any other value is mapped to the closest available line width on the workstation.

*ecol* **— specified by user, fullword integer**
Edge color index (>=0).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**63** LINETYPE VALUE < ONE

**64** SPECIFIED LINETYPE NOT AVAILABLE ON WORKSTATION

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**311** EDGE FLAG VALUE IS INVALID

**Related Subroutines**

**GPXER**
Set Extended Edge Representation

**RCP code**

201329412 (X'0C000B04')

## GPIR - Set Interior Representation

**GPIR (*wsid*, *index*, *style*, *sindex*, *icol*)**

**Purpose**

Use **GPIR** to set the given attribute values in the specified table entry of the specified workstation.

The empty and hollow interior styles display nothing for the interior. If the edge flag is 1=OFF and the style is 5=EMPTY, then no visual output is generated. The interior is detectable when a primitive with empty interior style is encountered and it is eligible for picking as determined by it visibility and detectability.

If the edge flag is 1=OFF and the style is 1=HOLLOW, then the boundary is drawn. When a primitive with hollow interior style is encountered only the boundary of the primitive is eligible for picking as determined by its visibility and detectability.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Interior bundle table index (>=1). Index of bundle table entry to be loaded.

*style* **— specified by user, fullword integer**
Interior style (1=HOLLOW, 2=SOLID, 3=PATTERN, 4=HATCH, 5=EMPTY).

*sindex* **— specified by user, fullword integer**
Interior style index (>=1). This index points into either the pattern or the hatch table, depending on the interior style of the polygon.

*icol* **— specified by user, fullword integer**
Interior color index (>=0). For interiors of style solid or hatch, the specified color is used to draw the interior. For interior style hollow, with the edge set to 1=OFF, the specified color is used to draw the perimeter of the polygon.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**83** INTERIOR STYLE NOT AVAILABLE ON WORKSTATION

**84** INTERIOR STYLE INDEX VALUE < ONE

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**310** INTERIOR STYLE VALUE IS INVALID

**Related Subroutines**

**GPXIR**
Set Extended Interior Representation

**RCP code**

201329416 (X'0C000B08')

## GPPLR - Set Polyline Representation

**GPPLR (*wsid*, *index*, *ltype*, *lwidth*, *color*)**

**Purpose**

Use **GPPLR** to set the given attribute values in the specified entry of the polyline bundle table.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Polyline bundle table index (>=1) Index of polyline bundle table entry to be loaded.

*ltype* **— specified by user, fullword integer**
Specifies an index into the workstation's line type table. The table size and specific entries supported are workstation dependent. Use the Inquire Polyline Facilities (**GPQPLF**) subroutine to locate the supported line types on your workstation. The default line type table for supported entries is defined with the following line types: (1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE. Any entry may be changed by the Set Linetype Representation (**GPLTR**) subroutine except entry 1).

*lwidth* **— specified by user, short floating-point number**
Line width scale factor is specified as a fraction of the nominal width of a line. A line width scale factor of 1.0, which is the default, generates a nominal size line on any workstation. Any other value is mapped to the closest line width available on the workstation.

*color* **— specified by user, fullword integer**
Polyline color index (>=0)

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**63** LINETYPE VALUE < ONE

**64** SPECIFIED LINETYPE NOT AVAILABLE ON WORKSTATION

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**Related Subroutines**

**GPXPLR**
Set Extended Polyline Representation

**RCP code**

201329409 (X'0C000B01')

## GPPMR - Set Polymarker Representation

**GPPMR (*wsid*, *index*, *mtype*, *msize*, *color*)**

**Purpose**

Use **GPPMR** to set the given attribute values in the specified entry of the polymarker bundle table.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Polymarker bundle table index (>=1). Index of table entry to be loaded.

*mtype* **— specified by user, fullword integer**
Specifies an index into the marker type table of the workstation. The table size and specific entries supported are workstation dependent. Use the Inquire Polymarker Facilities (**GPQPMF**) subroutine to locate the supported marker types on your workstation. The default marker type table for supported entries is defined with the following marker types: (`1=DOT, 2=PLUS_SIGN, 3=ASTERISK, 4=CIRCLE, 5=DIAGONAL_CROSS, 6-n=ASTERISK`. Any entry may be changed by the Set Marker Type Representation (**GPMTR**) subroutine except entry 3).

*msize* **— specified by user, short floating-point number**
Marker size scale factor is specified as a fraction of the nominal marker size. A marker size scale factor of 1.0, which is the default, generates a nominal size marker on any workstation. Any other value is mapped to the closest available marker size on the workstation.

*color* **— specified by user, fullword integer**
Polymarker color index (>=0).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**69** MARKER TYPE VALUE < ONE

**70** SPECIFIED MARKER TYPE NOT AVAILABLE ON WORKSTATION

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**Related Subroutines**

**GPXPMR**
Set Extended Polymarker Representation

**RCP code**

201329410 (X'0C000B02')

## GPQABK - Inquire Actual Break Capabilities

**GPQABK (*wsid*, *start*, *number*, *errind*, *ntrigs*, *ltrigs*)**

**Purpose**

Use **GPQABK** to inquire the break action capabilities of the given workstation. A list of available triggers for a break action is returned.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of available triggers (>=1).

*number* **— specified by user, fullword integer**
Number of triggers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**572** WORKSTATION DOES NOT SUPPORT PROGRAMMABLE BREAK ACTION

*ntrigs* **— returned by the graPHIGS API, fullword integer**
Total number of entries in the list of available triggers.

*ltrigs* **— returned by the graPHIGS API, array of fullword integers**
List of available triggers as specified by the start and number parameters. The list is an array of trigger descriptors, consisting of three fullword integers each, in which a descriptor consists of a triplet containing the trigger type, low trigger qualifier, and high trigger qualifier. Positive integers as trigger types are choice device numbers. The trigger qualifier for a choice device is the choice number. The parameter *ntrigs* identifies the total number of triplets in the available trigger list. The actual number returned depends on the setting of the *start* and *number* parameters.

### Error Codes

None

### Related Subroutines

**GPQBK**
Inquire Break Capabilities

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201336844 (X'0C00280C')

## GPQACF - Inquire Actual Color Facilities

**GPQACF (*wsid*, *errind*, *model*, *ncolor*, *avcolor*)**

### Purpose

Use **GPQACF** to inquire the actual color capabilities of the specified workstation.

The graPHIGS API returns the default color model, the maximum number of colors available, and data indicating if color is supported on the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**3** FUNCTION REQUIRES STATE WSOP

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*model* **— returned by the graPHIGS API, fullword integer**
Color model (`1=RGB, 2=HSV, 3=CMY`).

*ncolor* **— returned by the graPHIGS API, fullword integer**
Number of colors available (the total color palette size).

*avcolor* **— returned by the graPHIGS API, fullword integer**
Color availability (`1=MONOCHROME, 2=COLOR`).

**Error Codes**

None

**Related Subroutines**

**GPQCF**
Inquire Color Facilities

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336577 (X'0C002701')

## GPQADS - Inquire Actual Maximum Display Surface Size

**GPQADS (*wsid*, *errind*, *units*, *csize*, *asize*)**

**Purpose**

Use **GPQADS** to inquire the actual display surface size.

The graPHIGS API returns the values in Device Coordinate (DC) units and address units (for example, raster units) for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*units* **— returned by the graPHIGS API, fullword integer**
Device coordinate units (`1=METERS, 2=OTHER`).

*csize* **— returned by the graPHIGS API, 3 short floating-point numbers**
Actual display surface size in device coordinate units.

*asize* **— returned by the graPHIGS API, 3 fullword integers**
Actual display surface size in address units.

**Error Codes**

None

**Related Subroutines**

**GPQDS**
Inquire Maximum Display Surface Size

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336578 (X'0C002702')

## GPQAEF - Inquire Actual Edge Facilities

**GPQAEF (*wsid*, *start*, *number*, *errind*, *ntype*, *eltype*, *nelwidth*, *elwidth*, *minelw*, *maxelw*)**

**Purpose**

Use **GPQAEF** to inquire the actual edge facilities of the specified workstation.

The graPHIGS API returns values indicating the identifiers of the linetypes; the maximum quantity available; and the nominal, minimum, and maximum line width values. The line widths are returned in Device Coordinates (DC) for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the

available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of edge line types (>=1).

*number* **— specified by user, fullword integer**
Number of edge line types requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*ntype* **— returned by the graPHIGS API, fullword integer**
Total number of available edge line types.

*eltype* **— returned by the graPHIGS API, array of fullword integers**
Specifies an index into the workstation's edge line type table. The table size and specific entries supported are workstation dependent. The default edge line type table for supported entries is defined with the following line types: (`1=SOLID_LINE`, `2=DASHED`, `3=DOTTED`, `4=DASH_DOT`, `5=LONG_DASH`, `6=DOUBLE_DOT`, `7=DASH_DOUBLE_DOT`, `8-n=SOLID_LINE`. Any entry may be changed by the Set Linetype Representation (**GPLTR**) subroutine except entry 1.)

The output array must be large enough to contain the requested data.

*nelwidth* **— returned by the graPHIGS API, fullword integer**
Total number of available line widths of edges. (Zero means that the workstation supports a continuous range of line widths of edges).

*elwidth* **— returned by the graPHIGS API, short floating-point number (DC)**
Nominal line width of edge.

*minelw* **— returned by the graPHIGS API, short floating-point number (DC)**
Minimum line width of edge.

*maxelw* **— returned by the graPHIGS API, short floating-point number (DC)**
Maximum line width of edge.

**Error Codes**

None

**Related Subroutines**

**GPQEF**
Inquire Edge Facilities

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336596 (X'0C002714')

## GPQAES - Inquire List of Actual Available Escape Subroutines

**GPQAES (*wsid*, *start*, *number*, *errind*, *nids*, *idlist*)**

### Purpose

Use **GPQAES** to inquire the list of escape subroutine identifiers supported by a given workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member in the list of escape identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of escape identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*nids* **— returned by the graPHIGS API, fullword integer**
Total number of available escape identifiers.

*idlist* **— returned by the graPHIGS API, array of fullword integers**
List of available escape identifiers:

- 1001 = Sound Alarm
- 1002 = Enable/Disable Link Switch Notification
- 1003 = GDF/CGM Plot Size
- 1004 = Initialize Pick Correlation State
- 1005 = Set Pick Selection Criteria
- 1006 = Set Input Echo Color
- 1007 = Read Frame Buffer

- 1008 = Geometric Text Culling
- 1009 = Window Resize Notification Control
- 1010 = Inquire Mapped Display Surface Size
- 1011 = Window Exposure Notification Control
- 1012 = Window Deletion Notification Control
- 1014 = Workstation-Dependent Output
- 1015 = Convert Coordinate Values

**Error Codes**

None

**Related Subroutines**

**GPQES**
Inquire List of Available Escape Subroutines

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201340168 (X'0C003508')

## GPQAFC - Inquire Annotation Font Characteristics

**GPQAFC (*wsid*, *csid*, *font*, *start*, *num*, *errind*, *prec*, *nomh*, *nahsf*, *lahsf*, *lnfac*, *lmnfac*, *lmxfac*)**

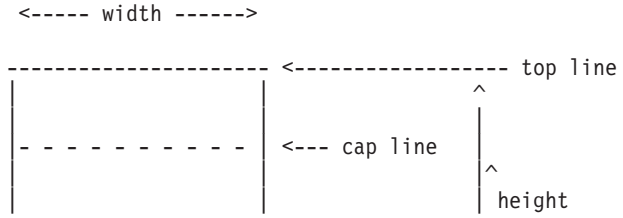
**Purpose**

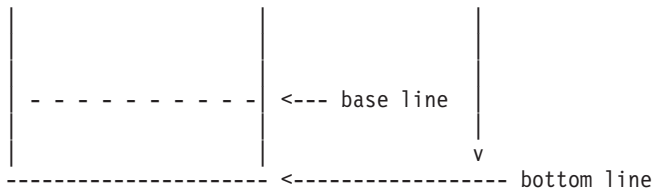
Use **GPQAFC** to inquire information concerning the annotation text capabilities for the specified character set and font on the specified workstation.

The information consists of the highest text precision supported by the specified workstation for the specified *csid*/ *font*, the number of supported annotation height scale factors, the nominal character height in Device Coordinates, a list indicating exact supported height scale factors and the number of character expansion factors, minimum expansion factor and maximum expansion factor, corresponding to each height scale factor.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

Annotation text capabilities are defined in terms of the following character box description:






Height is defined as the distance between the top line and bottom line.

> **Note:** This subroutine exists to maintain compatibility with previous graPHIGS API releases. Values of returned data are calculated with a different definition of character height than is used by the Inquire Extended Annotation Font Characteristics (**GPQXAF**) subroutine, and by the Inquire Font Characteristics (**GPQFCH**) subroutine.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*csid* — **specified by user, fullword integer**
Character set identifier.

See Appendix A. "Character Set and Font Identifiers" for more information.

*font* — **specified by user, fullword integer**
Font identifier (>=1).

*start* — **specified by user, fullword integer**
Starting member of the list of supported height scale factors (>=1).

*num* — **specified by user, fullword integer**
Number of list elements requested (>=0) This number refers to the number of list elements that are returned.

> **Note:** The four output arrays must be large enough to hold the requested number of elements.

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**75** TEXT FONT VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**542** CHARACTER SET IDENTIFIER IS INVALID

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**562** CHARACTER SET/FONT COMBINATION IS NOT AVAILABLE FOR ANNOTATION

**571** INQUIRED INFORMATION IS NOT AVAILABLE

*prec* — **returned by the graPHIGS API, fullword integer**
Highest available precision for the corresponding *csid*/*font* (`1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC`). If the highest precision supported is stroke, then the following parameters refer to the workstation's capabilities for character precision with annotation text.

*nomh* **— returned by the graPHIGS API, short floating-point number (DC)**
The bottom to top size, (top - bottom), in Device Coordinates (DC) for the nominal height font is returned in this field.

*nahsf* **— returned by the graPHIGS API, fullword integer**
The total number of annotation sizes that are supported. This also defines the number of entries in each of the next four parameter lists except when this value is zero. Zero means that a continuous range of annotation sizes is supported and each of the lists contains two entries.

*lahsf* **— returned by the graPHIGS API, array of short floating-point numbers**
List of the annotation height scale factors that are supported. The first entry is $1.0 \times 10^{-30}$ and the second entry is $9.9 \times 10^{30}$ if the device supports a continuous range of annotation heights.

> **Note:** This list has one entry for each annotation size supported. Its length is determined by the *nahsf* field except when that contains 0. In this case, this list has two entries. The first entry contains information for the minimum annotation height scale factor. The second contains information for the maximum.

*lnfac* **— returned by the graPHIGS API, array of fullword integers**
List of numbers of expansion factors that are supported. There is one entry for each of the supported annotation height scale factors. Zero means that a continuous range of expansion factors is supported for the font size.

> **Note:** The length of this list is determined by the *nahsf* field except when that contains 0. In this case, there are two entries in this list. The first entry contains information for the minimum annotation height scale factor. The second contains information for the maximum.

*lmnfac* **— returned by the graPHIGS API, array of short floating-point numbers**
List of minimum expansion factors that are supported. There is one entry for each of the supported annotation height scale factors. For devices which support all possible expansion factors, there is one entry for each of the supported annotation height scale factors. For devices which support all possible expansion factors, this parameter is a very small positive number (e.g. $10^{-30}$).

> **Note:** The length of this list is determined by the *nahsf* field except when that contains 0. In this case, there are two entries in this list. The first entry contains information for the minimum annotation height scale factor. The second contains information for the maximum.

> **Note:** The expansion factor is computed as follows: expf = (W/H) / generic where 'W' and 'H' are the width and height for the font with the corresponding scale factor and generic is the (W/H) ratio for a generic font and has the value 0.86667 (=65/75)

*lmxfac* **— returned by the graPHIGS API, array of short floating-point numbers**
List of maximum expansion factors that are supported. There is one entry for each of the supported annotation height scale factors. For devices which support all possible expansion factors, this parameter should contain a very large positive number (e.g. $9.9 \times 10^{30}$).

> **Note:** The length of this list is determined by the *nahsf* field except when that contains 0. In this case, there are two entries in this list. The first entry contains information for the minimum annotation height scale factor. The second contains information for the maximum.

> **Note:** The expansion factor is computed as follows: expf = (W/H) / generic where 'W' and 'H' are the width and height for the font with the corresponding scale factor and generic is the (W/H) ratio for a generic font and has the value 0.86667 (=65/75).

**Error Codes**

None

### Related Subroutines

**GPQXAF**
Inquire Extended Annotation Font Characteristics

### RCP code

201336589 (X'0C00270D')

## GPQAFP - Inquire Actual Font Pool Size

**GPQAFP (*wsid*, *errind*, *poolsize*)**

### Purpose

Use **GPQAFP** to inquire the actual font pool size for the specified workstation.

This represents the maximum number of fonts that can be simultaneously active on this workstation after it is open. The size is specified as the number of fonts.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*poolsize* **— returned by the graPHIGS API, fullword integer**
Font pool size for this workstation.

### Error Codes

None

### Related Subroutines

**GPQFP**
Inquire Font Pool Size

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201336599 (X'0C002717')

## GPQAGD - Inquire List of Actual Generalized Drawing Primitives

**GPQAGD (*wsid*, *start*, *number*, *errind*, *ngdp*, *lgdp*)**

**Purpose**

Use **GPQAGD** to inquire the actual generalized drawing primitives (GDP) available on the specified workstation.

The graPHIGS API returns the number of primitives available, and their GDP identifiers.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of GDP identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of GDP identifiers requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*ngdp* **— returned by the graPHIGS API, fullword integer**
Total number of supported GDP identifiers (>=1).

*lgdp* **— returned by the graPHIGS API, array of fullword integers.**
List of GDP identifiers. GDP identifiers are as follows:

- 1001 - Pixel 3
- 1002 - Pixel 2
- 1003 - Disjoint Polyline 3
- 1004 - Disjoint Polyline 2
- 1005 - Circle 2
- 1006 - Circular Arc 2
- 1007 - Ellipse 2
- 1008 - Ellipse 3
- 1009 - Elliptical Arc 2
- 1010 - Elliptical Arc 3
- 1014 - Polyline Set 3 With Data
- 1016 - Polygon 3 With Data
- 1017 - Polygon 2 With Data

- 1020 - Marker Grid 3
- 1021 - Marker Grid 2
- 1022 - Line Grid 3
- 1023 - Line Grid 2
- 1027 - Composite Fill Area 2
- 1029 - Triangle Strip 3
- 1031 - Quadrilateral Mesh 3
- 1033 - Non-Uniform B-Spline Curve 3
- 1034 - Non-Uniform B-Spline Curve 2
- 1035 - Non-Uniform B-Surface
- 1036 - Trimmed Non-Uniform B-Spline Surface
- 1037 - Polyhedron Edge
- 1039 - Character Line 2
- 1046 - Polysphere

**Note:** The output array must be large enough to contain the requested data.

### Error Codes

None

### Related Subroutines

**GPQGD**
Inquire List of Generalized Drawing Primitives

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201336579 (X'0C002703')

## GPQAIF - Inquire Actual Interior Facilities

**GPQAIF (*wsid*, *startp*, *nump*, *starth*, *numh*, *errind*, *intnum*, *interiors*, *hatnum*, *hatch*)**

### Purpose

Use **GPQAIF** to inquire the actual interior facilities of a particular workstation. The graPHIGS API returns values indicating the maximum quantity of interior styles and their identifiers and the maximum quantity of hatch styles and their identifiers at the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*startp* — **specified by user, fullword integer**
Starting member of the list of interior styles (>=1).

*nump* — **specified by user, fullword integer**
Number of interior styles requested (>=0).

*starth* — **specified by user, fullword integer**
Starting member of the list of hatch styles (>=1).

*numh* — **specified by user, fullword integer**
Number of hatch styles requested (>=0).

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*intnum* — **returned by the graPHIGS API, fullword integer**
Total number of available interior styles.

*interiors* — **returned by the graPHIGS API, array of fullword integers**
List of available interior styles (`1=HOLLOW, 2=SOLID, 3=PATTERN, 4=HATCH, 5=EMPTY`) The output array must be large enough to contain the requested data.

*hatnum* — **returned by the graPHIGS API, fullword integer**
Total number of available hatch styles.

*hatch* — **returned by the graPHIGS API, array of fullword integers.**
List of available hatch styles. The output array must be large enough to contain the requested data.

See GPISI - Set Interior Style Index for the default hatch styles in the workstation's hatch table.

**Error Codes**

None

**Related Subroutines**

**GPQIF**
Inquire Interior Facilities

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336582 (X'0C002706')

## GPQAIS - Inquire Actual Input Character Set Facilities

**GPQAIS (*wsid*, *class*, *device*, *start*, *number*, *errind*, *ncsid*, *csid*)**

### Purpose

Use **GPQAIS** to inquire the actual input character sets that are available for a specified input device.

The values returned by the graPHIGS API indicate the total quantity of character sets available and their identifiers.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*class* **— specified by user, fullword integer**
Device class requested (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*device* **— specified by user, fullword integer**
Device number.

*start* **— specified by user, fullword integer**
Starting member of the list of character set identifiers (>=1).

*number* **— specified by user, fullword integer**
Number of character set identifiers requested (>=0). Number refers to the number of *csid* values the graPHIGS API returns.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*ncsid* **— returned by the graPHIGS API, fullword integer**
Total number of character set identifiers.

*csid* **— returned by the graPHIGS API, array of fullword integers.**
List of character set identifiers.

See Appendix A. "Character Set and Font Identifiers" for more information.

The output array must be large enough to contain the requested data.

### Error Codes

None

### Related Subroutines

**GPQISF**
Inquire Input Character Set Facilities

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201336592 (X'0C002710')

## GPQAIT - Inquire Actual Input Trigger Capabilities

**GPQAIT (*wsid*, *class*, *devnum*, *start*, *number*, *errind*, *ntrigs*, *ltrigs*)**

### Purpose

Use **GPQAIT** to inquire the input device trigger capabilities of a specified device on a given workstation. If the triggers are programmable, a list of available triggers is returned. The returned list corresponds to the available triggers for all triggers of the given input device.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*class* **— specified by user, fullword integer**
Input device class (1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING).

*devnum* **— specified by user, fullword integer**
Input device number (>=1).

*start* **— specified by user, fullword integer**
Starting member in the list of available trigger types (>=1).

*number* **— specified by user, fullword integer**
Number of triggers requested from the list (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**140** DEVICE NUMBER < ONE OR DEVICE NOT AVAILABLE

**328** INPUT CLASS VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**569** DEVICE DOES NOT SUPPORT PROGRAMMABLE TRIGGERS

*ntrigs* **— returned by the graPHIGS API, fullword integer**
Total number of entries in the list of available triggers.

*ltrigs* **— returned by the graPHIGS API, array of fullword integers**
List of trigger descriptor triplets. The list is an array of trigger descriptors in which each descriptor consists of three fullword integers designating the trigger type, low trigger qualifier, and high trigger qualifier. The trigger type field has the following meanings:

| Type | Meaning |
|---|---|
| **>0** | Identifier of physical device within the button category.<br><br>The trigger qualifiers for this trigger type are a range of choice numbers generated by the physical device. |
| **-1** | Change of the measure of the logical input device<br><br>The trigger qualifier fields are ignored. |

The parameter *ntrigs* identifies the total number of triplets in the available trigger list. The actual number returned depends on the setting of the *start* and *number* parameters.

### Error Codes

None

### Related Subroutines

**GPQIT**
Inquire Input Trigger Capabilities

**GPQRCT**
Inquire Realized Connection Type

### RCP code

201336845 (X'0C00280D')

## GPQALF - Inquire Actual Polyline Facilities

**GPQALF (*wsid*, *start*, *number*, *errind*, *ntype*, *ltype*, *nlwidth*, *lwidth*, *minlw*, *maxlw*)**

### Purpose

Use **GPQALF** to inquire the actual polyline facilities for the specific workstation.

The values returned by the graPHIGS API indicate the total quantity of available linetypes, the specific supported linetypes, and the total quantity of available line widths and their minimum and maximum values. The returned widths of the line are in Device Coordinates for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of linetypes (>=1).

*number* **— specified by user, fullword integer**
Number of linetypes requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
If the error indicator is zero, the request was completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*ntype* **— returned by the graPHIGS API, fullword integer**
Total number of available linetypes.

*ltype* **— returned by the graPHIGS API, array of fullword integers.**
Specifies an index into the workstation's available line type table. The table size and specific entries are workstation dependent. The default available line type table for supported entries is defined with the following line types: (`1=SOLID_LINE`, `2=DASHED`, `3=DOTTED`, `4=DASH_DOT`, `5=LONG_DASH`, `6=DOUBLE_DOT`, `7=DASH_DOUBLE_DOT`, `8-n=SOLID_LINE`. Any entry may be changed by the Set Linetype Representation (**GPLTR**) subroutine except entry 1).

The output array must be large enough to contain the requested data.

*nlwidth* **— returned by the graPHIGS API, fullword integer**
Number of available line widths. (Zero means that the workstation supports a continuous range of line widths.)

*lwidth* **— returned by the graPHIGS API, short floating-point number (DC)**
Nominal line width.

*minlw* **— returned by the graPHIGS API, short floating-point number (DC)**
Minimum line width.

*maxlw* **— returned by the graPHIGS API, short floating-point number (DC)**
Maximum line width.

**Error Codes**

None

**Related Subroutines**

**GPQPLF**
Inquire Polyline Facilities

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336583 (X'0C002707')

# GPQALI - Inquire List of Actual Logical Input Devices

**GPQALI (*wsid*, *class*, *start*, *number*, *errind*, *ndev*, *dev*)**

### Purpose

Use **GPQALI** to inquire the available input devices at the specified workstation.

For the specified device class, the graPHIGS API returns a number indicating the quantity of logical devices and their numbers for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*class* **— specified by user, fullword integer**
Device class (`1=LOCATOR, 2=STROKE, 3=VALUATOR, 4=CHOICE, 5=PICK, 6=STRING`).

*start* **— specified by user, fullword integer**
Starting member of the list of input devices (>=1).

*number* **— specified by user, fullword integer**
Number of input device numbers requested (>=0) This refers to the total quantity of device numbers the graPHIGS API returns.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**38** WORKSTATION HAS ONLY OUTPUT CAPABILITIES

**328** INPUT CLASS VALUE IS INVALID

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*ndev* **— returned by the graPHIGS API, fullword integer**
Total quantity of logical devices in the device class.

*dev* **— returned by the graPHIGS API, array of fullword integers**
List of device numbers in the device class. The output array must be large enough to contain the requested data.

### Error Codes

None

**Related Subroutines**

**GPQLI**
Inquire List of Logical Input Devices

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336580 (X'0C002704')

## GPQALW - Inquire Actual Length of Workstation State Tables

**GPQALW (*wsid*, *errind*, *ltable*, *mtable*, *ttable*, *itable*, *etable*, *pttable*, *ctable*)**

**Purpose**

Use **GPQALW** to inquire the actual length of the workstation tables in the WSL.

The graPHIGS API returns values indicating the maximum number of entries in the bundle and color tables for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*ltable* **— returned by the graPHIGS API, fullword integer**
Total number of polyline bundle table entries.

*mtable* **— returned by the graPHIGS API, fullword integer**
Total number of polymarker bundle table entries.

*ttable* **— returned by the graPHIGS API, fullword integer**
Total number of text bundle table entries.

*itable* **— returned by the graPHIGS API, fullword integer**
Total number of interior bundle table entries.

*etable* **— returned by the graPHIGS API, fullword integer**
Total number of edge bundle table entries.

*pttable* **— returned by the graPHIGS API, fullword integer**
Total number of pattern indexes.

*ctable* **— returned by the graPHIGS API, fullword integer**
Total number of default color table indexes.

**Error Codes**

None

**Related Subroutines**

**GPQLW**
Inquire Length of Workstation State Tables

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336586 (X'0C00270A')

## GPQAMF - Inquire Actual Polymarker Facilities

**GPQAMF (*wsid*, *start*, *number*, *errind*, *ntype*, *mtype*, *nsize*, *size*, *minms*, *maxms*)**

**Purpose**

Use **GPQAMF** to inquire the quantity of polymarker types and polymarker sizes on the specified workstation.

The graPHIGS API returns a number that indicates the total quantity of polymarker types and their identifiers and data indicating the total quantity of nominal, minimum, and maximum polymarker sizes for the specified workstation. The marker sizes are returned in Device Coordinates (DC) If the number of available marker sizes is returned as zero, the workstation supports a continuous range of marker sizes within the minimum and maximum sizes indicated.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Starting member of the list of marker types (>=1).

*number* **— specified by user, fullword integer**
Number of marker types requested (>=0).

This number refers to the quantity of marker type values the graPHIGS API returns.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*ntype* **— returned by the graPHIGS API, fullword integer**
Total number of available marker types.

*mtype* **— returned by the graPHIGS API, array of fullword integers.**
Specifies an index into the marker type table of the workstation. The table size and specific entries supported are workstation dependent. The default marker type table for supported entries is defined with the following marker types: (1=DOT, 2=PLUS_SIGN, 3=ASTERISK, 4=CIRCLE, 5=DIAGONAL_CROSS, 6-n=ASTERISK. Any entry may be changed by the Set Marker Type Representation (**GPMTR**) subroutine except entry 3. Any output array must be large enough to contain the requested data.

*nsize* **— returned by the graPHIGS API, fullword integer**
Number of available marker sizes. (Zero means that the workstation supports a continuous range of marker sizes).

*size* **— returned by the graPHIGS API, short floating-point number (DC)**
Nominal marker size.

*minms* **— returned by the graPHIGS API, short floating-point number (DC)**
Minimum marker size.

*maxms* **— returned by the graPHIGS API, short floating-point number (DC)**
Maximum marker size.

**Error Codes**

None

**Related Subroutines**

**GPQPMF**
Inquire Polymarker Facilities

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336581 (X'0C002705')

## GPQANV - Inquire Actual Number of Definable View Table Entries

**GPQANV (*wsid*, *errind*, *number*)**

**Purpose**

Use **GPQANV** to inquire the quantity of definable view table entries for the specified workstation. (View zero is not included because it cannot be modified).

The number the graPHIGS API returns indicates the quantity of view table entries that can be defined for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*number* **— returned by the graPHIGS API, fullword integer**
Number of definable view table indexes. (Entry 0 may not be changed.)

**Error Codes**

None

**Related Subroutines**

**GPQNV**
Inquire Number of Definable View Table Entries

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336587 (X'0C00270B')

## GPQAPF - Inquire Actual Pattern Facilities

**GPQAPF (*wsid*, *errind*, *maxrow*, *maxcol*)**

**Purpose**

Use **GPQAPF** to inquire the maximum pattern dimensions supported by the specified workstation.

The values the graPHIGS API returns indicate the maximum quantity of rows and columns that can be placed in a pattern array for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*maxrow* **— returned by the graPHIGS API, fullword integer**
Maximum number of rows in the pattern array.

*maxcol* **— returned by the graPHIGS API, fullword integer**
Maximum number of columns in the pattern array.

**Error Codes**

None

**Related Subroutines**

**GPQPAF**
Inquire Pattern Facilities

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336584 (X'0C002708')

## GPQAPS - Inquire Actual Primary Character Set

**GPQAPS (*wsid*, *errind*, *csid*)**

**Purpose**

Use **GPQAPS** to inquire the primary character set on the specified workstation.

The graPHIGS API returns the primary Character Set Identifier (*csid*) for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*csid* **— returned by the graPHIGS API, fullword integer**
Character set identifier. See Appendix A. "Character Set and Font Identifiers" for more information.

**Error Codes**

None

**Related Subroutines**

**GPQPCS**
Inquire Primary Character Set

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336593 (X'0C002711')

## GPQAVF - Inquire Actual View Facilities

**GPQAVF (*wsid*, *errind*, *shield*)**

**Purpose**

Use **GPQAVF** to inquire the availability of view shielding for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

*shield* **— returned by the graPHIGS API, fullword integer**
Shielding available (`1=NOT_AVAILABLE, 2=AVAILABLE`).

**Error Codes**

None

**Related Subroutines**

**GPQVF**
Inquire View Facilities

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336585 (X'0C002709')

## GPQAWC - Inquire Actual Workstation Category

**GPQAWC (*wsid*, *errind*, *type*)**

**Purpose**

Use **GPQAWC** to inquire the actual workstation category for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

*type* **— returned by the graPHIGS API, fullword integer**
Workstation category (`1=OUTPUT, 2=INPUT, 3=OUTIN`).

**Error Codes**

None

**Related Subroutines**

**GPQWC**
Inquire Workstation Category

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336588 (X'0C00270C')

## GPQAWD - Inquire Actual Workstation Display Classification

**GPQAWD (*wsid*, *errind*, *type*)**

**Purpose**

Use **GPQAWD** to inquire the actual type of display technology for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

*type* **— returned by the graPHIGS API, fullword integer**
Display type (1=VECTOR, 2=RASTER, 3=OTHERS).

**Error Codes**

None

**Related Subroutines**

**GPQWD**
Inquire Workstation Display Classification

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336595 (X'0C002713')

## GPQCR - Inquire Color Representation

**GPQCR (*wsid*, *start*, *number*, *type*, *errind*, *colors*)**

**Purpose**

Use **GPQCR** to inquire the current color values in the specified workstation's default color table.

If the workstation's display color table is modifiable, then the display color table is the workstation's default. Otherwise the rendering color table is the workstation's default color table.

Use Inquire Extended Color Facilities (**GPQXCF**) subroutine to inquire the characteristics of the workstation's color table.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

Use Inquire Actual Length of Workstation Tables (**GPQALW**) subroutine to determine the actual size of the workstation's color table.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*start* **— specified by user, fullword integer**
Index specifying an entry into the default color table to start returning the requested color values (>=0).

*number* **— specified by user, fullword integer**
Number of color table entries requested (>=0).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**534** TYPE VALUE IS INVALID

**539** REQUESTED NUMBER < ZERO

**544** START VALUE < ZERO

**551** START VALUE EXCEEDS COLOR TABLE SIZE

*colors* **— returned by the graPHIGS API, array of three short floating-point numbers**
Color components are to be interpreted by the current color model. The array contains a list of color table entries ordered by row. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPCR** Set Color Representation

**GPQXCR**
Inquire Extended Color Representation

**RCP code**

201339137 (X'0C003101')

## GPQCVX - Inquire Current Viewing Transformation

**GPQCVX (*wsid*, *view*, *errind*, *matrix*, *window*, *viewpt*, *viewt*, *refpt*, *dist*, *near*, *far*, *wincp*, *nearcp*, *farcp*, *shield*, *shldci*, *border*, *brdrci*, *viewact*)**

**Purpose**

Use **GPQCVX** to inquire the current view information for the specified view on the specified workstation.

If your application uses direct color to set the shielding or border color of the specified view, then you must use the Inquire Current View Representation (**GPQCVR**) subroutine to inquire the view information.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Current viewing transformation matrix. For the transformation matrix, the elements are returned in the following order:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24|---> (m11,m12,m13,m14,m21...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

*window* **— returned by the graPHIGS API, 4 short floating-point numbers (VC)**
Current window (Umin, Umax, Vmin, Vmax).

*viewpt* **— returned by the graPHIGS API, 6 short floating-point numbers (NPC)**
Current viewport (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax)

*viewt* **— returned by the graPHIGS API, fullword integer**
Current view type (`1=PARALLEL, 2=PERSPECTIVE`)

*refpt* **— returned by the graPHIGS API, 3 short floating-point numbers (VC)**
Current projection reference point (*u*, *v*, *n*)

*dist* **— returned by the graPHIGS API, short floating-point number (VC)**
Current distance of view plane from the view reference point along the n-axis.

*near* **— returned by the graPHIGS API, short floating-point number (VC)**
Current distance of near clipping plane from the view reference point along the n-axis.

*far* **— returned by the graPHIGS API, short floating-point number (VC)**
Current distance of far clipping plane from the view reference point along the n-axis.

*wincp* **— returned by the graPHIGS API, fullword integer**
Current window clipping indicator (`1=NOCLIP, 2=CLIP`)

*nearcp* **— returned by the graPHIGS API, fullword integer**
Current near clipping indicator (1=NOCLIP, 2=CLIP).

*farcp* **— returned by the graPHIGS API, fullword integer**
Current far clipping indicator (1=NOCLIP, 2=CLIP).

*shield* **— returned by the graPHIGS API, fullword integer**
Current shielding indicator (1=OFF, 2=ON).

*shldci* **— returned by the graPHIGS API, fullword integer**
Current shielding color index.

**Note:** Direct shielding color returns error indicator 526.

*border* **— returned by the graPHIGS API, fullword integer**
Current view border indicator (1=OFF, 2=ON).

*brdrci* **— returned by the graPHIGS API, fullword integer**
Current view border color index.

**Note:** Direct border color returns error indicator 526.

*viewact* **— returned by the graPHIGS API, fullword integer**
Current view active indicator for output (1=INACTIVE, 2=ACTIVE).

**Error Codes**

None

**Related Subroutines**

**GPQCVR**
Inquire Current View Representation

**RCP code**

201336835 (X'0C002803')

## GPQE - Inquire Element Content

**GPQE (*start*, *number*, *errind*, *ndata*, *data*)**

**Purpose**

Use **GPQE** to retrieve the current structure element content that is indicated by the element pointer. It will only return the data if the element is an element supported by Version 1 of the graPHIGS API.

This subroutine returns the data contained in the current element. The number of bytes in the element and the element contents are returned. The element type can be determined by using the Inquire List of Element Headers (**GPQEHD**) subroutine (or, to maintain compatibility with previous releases, the Inquire Element Type and Size [**GPQETS**] [page GPQETS - Inquire Element Type and Size] subroutine).

To execute this subroutine, a structure must be open.

For information on the format of the data, see *The graPHIGS Programming Interface: Technical Reference*.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

> **Note:** This subroutine exists to maintain compatibility with previous graPHIGS API releases. The format of the data returned is different from that returned with the Inquire List of Element Data (**GPQED**) subroutine. This subroutine can not return the element content for all the new structure elements that were not supported in graPHIGS API Version 1. For the new structure elements, the Inquire List of Element Data (**GPQED**) subroutine must be used to get the element content.

**Parameters**

*start* **— specified by user, fullword integer**
Starting byte offset in the element content (>=1).

*number* **— specified by user, fullword integer**
Number of bytes of the element content requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**4** FUNCTION REQUIRES STATE STOP

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

**535** CURRENT ELEMENT POINTER IS ZERO

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*ndata* **— returned by the graPHIGS API, fullword integer**
Total number of bytes in the element record.

*data* **— returned by the graPHIGS API, variable length data**
Structure element content. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPQED**
Inquire List of Element Data

**GPQEHD**
Inquire List of Element Headers

**GPQETS**
Inquire Element Type and Size

**RCP code**

201337090 (X'0C002902')

# GPQER - Inquire Edge Representation

**GPQER (*wsid*, *index*, *type*, *errind*, *edgefg*, *edgelt*, *edgesf*, *ecol*)**

### Purpose

Use **GPQER** to inquire the current attribute values in the specified entry in the edge bundle table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Edge bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

**534** TYPE VALUE IS INVALID

*edgefg* **— returned by the graPHIGS API, fullword integer**
Edge flag in the specified edge table entry (1=OFF, 2=ON, 3=GEOMETRY_ONLY).

*edgelt* **— returned by the graPHIGS API, fullword integer**
Specifies an index into the workstation's edge line type table. The table size and specified entries supported are workstation dependent. The default edge line type table for supported entries is defined with the following line types: 1=SOLID_LINE, 2=DASHED, 3=DOTTED, 4=DASH_DOT, 5=LONG_DASH, 6=DOUBLE_DOT, 7=DASH_DOUBLE_DOT, 8-n=SOLID_LINE. Any entry may be changed by the Set Linetype Representation (**GPLTR**) subroutine except entry 1.

*edgesf* **— returned by the graPHIGS API, short floating-point number**
Edge scale factor in the specified edge table entry.

*ecol* **— returned by the graPHIGS API, fullword integer**
Edge color index in the specified edge table entry.

**Note:** Direct edge color returns error indicator 526.

### Error Codes

None

**Related Subroutines**

**GPQXER**
Inquire Extended Edge Representation

**RCP code**

201339138 (X'0C003102')

## GPQETS - Inquire Element Type and Size

**GPQETS (*errind*, *type*, *size*)**

**Purpose**

Use **GPQETS** to inquire the type and size of the current element. It will only return the data if the current element is an element supported by Version 1 of the graPHIGS API.

The subroutine returns the element type and size in bytes. To use this subroutine a structure must be open. If the element pointer is currently zero, a value of zero is returned in the *type* parameter. To retrieve the element contents, use the Inquire List of Element Data (**GPQED**) subroutine or, to maintain compatibility with previous releases, use Inquire Element Content (**GPQE**) subroutine.

> **Note:** This subroutine exists to maintain compatibility with previous graPHIGS API releases. The format of the data returned is different from that returned with **GPQED**. This subroutine can not return the element header information (type and size) for all the new structure elements that were not supported in graPHIGS API Version 1. For the new structure elements, use the Inquire List of Element Headers (**GPQEHD**) subroutine which must be used to get the element header information.

**Parameters**

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**4** FUNCTION REQUIRES STATE STOP

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

*type* **— returned by the graPHIGS API, fullword integer**
Element type.

A value of zero indicates that the current element pointer is zero. Otherwise, current element types are as follows:

- 4352 - Polyline 3
- 4353 - Polyline 2
- 4354 - Polymarker 3
- 4355 - Polymarker 2
- 4356 - Text 3
- 4357 - Text 2
- 4358 - Polygon 3
- 4359 - Polygon 2

- 4366 - Annotation 3
- 4367 - Annotation 2
- 4368 - Pixel 3 (GDP Identifier = 1001)
- 4369 - Pixel 2 (GDP Identifier = 1002)
- 4370 - Disjoint Polyline 3 (GDP Identifier = 1003)
- 4371 - Disjoint Polyline 2 (GDP Identifier = 1004)
- 4372 - Circle 2 (GDP Identifier = 1005)
- 4373 - Circular Arc 2 (GDP Identifier = 1006)
- 4374 - Ellipse 2 (GDP Identifier = 1007)
- 4375 - Ellipse 3 (GDP Identifier = 1008)
- 4376 - Elliptical Arc 2 (GDP Identifier = 1009)
- 4377 - Elliptical Arc 3 (GDP Identifier = 1010)
- 4608 - Set Polyline Index
- 4609 - Set Polymarker Index
- 4610 - Set Text Index
- 4611 - Set Edge Index
- 4612 - Set Interior Index
- 4864 - Set Linetype
- 4865 - Set Linewidth Scale Factor
- 4866 - Set Polyline Color Index
- 4867 - Set Marker Type
- 4868 - Set Marker Size Scale Factor
- 4869 - Set Polymarker Color Index
- 4870 - Set Text Font
- 4871 - Set Text Precision
- 4872 - Set Character Expansion Factor
- 4873 - Set Character Spacing
- 4874 - Set Text Color Index
- 4875 - Set Character Height
- 4876 - Set Character Up Vector
- 4877 - Set Text Path
- 4880 - Set Text Alignment
- 4881 - Set Interior Style
- 4882 - Set Interior Style Index
- 4883 - Set Interior Color Index
- 4884 - Set Edge Flag
- 4885 - Set Edge Linetype
- 4886 - Set Edge Color Index
- 4887 - Set Edge Scale Factor
- 4894 - Set Aspect Source Flag Setting
- 4895 - Set Annotation Height Scale Factor
- 4897 - Set Polyline End Type
- 5120 - Set Modeling Transformation 3
- 5121 - Set Modeling Transformation 2
- 5122 - Set Global Transformation 3

- 5123 - Set Global Transformation 2
- 5376 - Label
- 5377 - Execute Structure
- 5378 - Application Data
- 5379 - Add Class Name to Set
- 5380 - Remove Class Name from Set
- 5381 - Set Pick Id
- 5382 - Set Highlighting Color Index

*size* **— returned by the graPHIGS API, fullword integer**
Element size in bytes.

> **Note:** To retrieve the element's contents, the value of this parameter can be used as input to the Inquire Element Content (**GPQE**) subroutine. The value of this parameter is an estimate and may be larger than the actual element size.

### Error Codes

None

### Related Subroutines

**GPQE** Inquire Element Content

**GPQED**
Inquire List of Element Data

**GPQEHD**
Inquire List of Element Headers

### RCP code

201337092 (X'0C002904')

## GPQIR - Inquire Interior Representation

**GPQIR (*wsid*, *index*, *type*, *errind*, *style*, *sindex*, *icol*)**

### Purpose

Use **GPQIR** to inquire the current attribute values in the specified entry in the interior bundle table for the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Interior bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

**534** TYPE VALUE IS INVALID

*style* **— returned by the graPHIGS API, fullword integer**
Interior style for the specified entry (1=HOLLOW, 2=SOLID, 3=PATTERN, 4=HATCH, 5=EMPTY).

*sindex* **— returned by the graPHIGS API, fullword integer**
Interior style index for the specified entry.

*icol* **— returned by the graPHIGS API, fullword integer**
Interior color index for the specified entry.

**Note:** Direct interior color returns error indicator 526.

**Error Codes**

None

**Related Subroutines**

**GPQXIR**
Inquire Extended Interior Representation

**GPXIR**
Set Extended Interior Representation

**RCP code**

201339139 (X'0C003103')

## GPQLR - Inquire Polyline Representation

**GPQLR (*wsid*, *index*, *type*, *errind*, *ltype*, *lwidth*, *color*)**

**Purpose**

Use **GPQLR** to inquire the current attribute values in the specified entry in the polyline bundle table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*index* — **specified by user, fullword integer**
Polyline bundle table index (>=1).

*type* — **specified by user, fullword integer**
Type of returned values (1=SET).

*errind* — **returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

**534** TYPE VALUE IS INVALID

*ltype* — **returned by the graPHIGS API, fullword integer**
Line type in the specified entry.

*lwidth* — **returned by the graPHIGS API, short floating-point number**
Line width scale factor in the specified entry.

*color* — **returned by the graPHIGS API, fullword integer**
Polyline color index in the specified entry.

**Note:** Direct polyline color returns error indicator 526.

**Error Codes**

None

**Related Subroutines**

**GPXPLR**
Set Extended Polyline Representation

**RCP code**

201339140 (X'0C003104')

## GPQMR - Inquire Polymarker Representation

**GPQMR (*wsid*, *index*, *type*, *errind*, *mtype*, *msize*, *color*)**

**Purpose**

Use **GPQMR** to inquire the current attribute values in the specified entry in the polymarker bundle table of the specified workstation.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Polymarker bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

**534** TYPE VALUE IS INVALID

*mtype* **— returned by the graPHIGS API, fullword integer**
Marker type in the specified entry.

*msize* **— returned by the graPHIGS API, short floating-point number**
Marker size scale factor in the specified entry.

*color* **— returned by the graPHIGS API, fullword integer**
Polymarker color index in the specified entry.

**Note:** Direct polymarker color returns error indicator 526.

**Error Codes**

None

**Related Subroutines**

**GPQXMR**
Inquire Extended Polymarker Representation

**GPXPMR**
Set Extended Polymarker Representation

**RCP code**

201339141 (X'0C003105')

# GPQRVX - Inquire Requested Viewing Transformation

**GPQRVX (*wsid*, *view*, *errind*, *matrix*, *window*, *viewpt*, *viewt*, *refpt*, *dist*, *near*, *far*, *wincp*, *nearcp*, *farcp*, *shield*, *shldci*, *border*, *brdrci*, *viewact*)**

### Purpose

Use **GPQRVX** to inquire the viewing parameters of a specified view in the specified workstation view table. These values are not current, if the values have been passed to the graPHIGS API but the workstation has not been updated. If your application uses direct color to set the shielding or border color of the specified view, then you must use the Inquire Requested View Representation (**GPQRVR**) subroutine to inquire the view information.

The graPHIGS API returns the requested viewing transformation matrix, the requested viewport, and the view characteristics.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

*matrix* **— returned by the graPHIGS API, 16 short floating-point numbers**
Requested viewing transformation matrix. For the transformation matrix, the elements are returned in the following order:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24|---> (m11,m12,m13,m14,m21....m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

*window* **— returned by the graPHIGS API, 4 short floating-point numbers (VC)**
Requested window (Umin, Umax, Vmin, Vmax).

*viewpt* **— returned by the graPHIGS API, 6 short floating-point numbers (NPC)**
Requested viewport (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*viewt* **— returned by the graPHIGS API, fullword integer**
Requested view type (`1=PARALLEL, 2=PERSPECTIVE`).

*refpt* **— returned by the graPHIGS API, 3 short floating-point numbers (VC)**
Requested projection reference point (*u*, *v*, *n*).

*dist* **— returned by the graPHIGS API, short floating-point number (VC)**
Requested distance of the view plane from the view reference point along the n-axis.

*near* **— returned by the graPHIGS API, short floating-point number (VC)**
Requested distance of the near clipping plane from the view reference point along the n-axis.

*far* **— returned by the graPHIGS API, short floating-point number (VC)**
Requested distance of the far clipping plane from the view reference point along the n-axis.

*wincp* **— returned by the graPHIGS API, fullword integer**
Requested window clipping indicator (`1=NOCLIP, 2=CLIP`).

*nearcp* **— returned by the graPHIGS API, fullword integer**
Requested near clipping indicator (`1=NOCLIP, 2=CLIP`).

*farcp* **— returned by the graPHIGS API, fullword integer**
Requested far clipping indicator (`1=NOCLIP, 2=CLIP`).

*shield* **— returned by the graPHIGS API, fullword integer**
Requested shielding indicator (`1=OFF, 2=ON`).

*shldci* **— returned by the graPHIGS API, fullword integer**
Requested shielding color index.

**Note:** Direct shielding color returns error indicator 526.

*border* **— returned by the graPHIGS API, fullword integer**
Requested view border indicator (`1=OFF, 2=ON`).

*brdrci* **— returned by the graPHIGS API, fullword integer**
Requested view border color index.

**Note:** Direct border color returns error indicator 526.

*viewact* **— returned by the graPHIGS API, fullword integer**
Requested view active indicator for output (`1=INACTIVE, 2=ACTIVE`).

### Error Codes

None

### Related Subroutines

**GPQRVR**
Inquire Requested View Representation

### RCP code

201336840 (X'0C002808')

## GPQSTE - Inquire Structure Existence

**GPQSTE (*strid*, *flag*)**

### Purpose

Use **GPQSTE** to inquire whether the specified structure exists in the currently selected structure store.

If there is no structure store currently selected, that is, the current structure state is Structure Store Close (SSCL) or Structure Store Open (SSOP), the structure existence indicator returns 1=NON_EXISTENT.

**Parameters**

*strid* **— specified by user, fullword integer**
Structure identifier.

*flag* **— returned by the graPHIGS API, fullword integer**
Structure existence indicator (1=NON_EXISTENT, 2=EXISTENT).

**Error Codes**

None

**Related Subroutines**

**GPQSTS**
Inquire Structure Status

**RCP code**

201337096 (X'0C002908')

## GPQTR - Inquire Text Representation

**GPQTR (*wsid*, *index*, *type*, *errind*, *font*, *prec*, *factor*, *space*, *color*)**

**Purpose**

Use **GPQTR** to inquire the current text attributes contained in the specified entry in the text bundle table of the specified workstation.

The graPHIGS API returns data indicating the text font and precision, character expansion factor and spacing, and text color of the specified entry.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Text bundle table index (>=1).

*type* **— specified by user, fullword integer**
Type of returned values (1=SET).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**526** REQUESTED DATA NOT AVAILABLE FOR THIS FUNCTION

**534** TYPE VALUE IS INVALID

*font* **— returned by the graPHIGS API, fullword integer**
Text font corresponding to the specified entry.

*prec* **— returned by the graPHIGS API, fullword integer**
Text precision corresponding to the specified entry (1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC).

*factor* **— returned by the graPHIGS API, short floating-point number**
Character expansion factor corresponding to the specified entry is specified as the deviation from the nominal width-to-height ratio for the font.

*space* **— returned by the graPHIGS API, short floating-point number**
Character spacing corresponding to the specified entry is specified as a fraction of the font-nominal character height.

*color* **— returned by the graPHIGS API, fullword integer**
Text color index corresponding to the specified entry.

**Note:** Direct text color returns error indicator 526.

### Error Codes

None

### Related Subroutines

**GPQXTR**
Inquire Extended Text Representation

### RCP code

201339143 (X'0C003107')

## GPQTXF - Inquire Text Facilities

**GPQTXF (*wstype*, *errind*, *npred*)**

### Purpose

Use **GPQTXF** to inquire the number of predefined text bundle table entries in the WDT for the specified workstation type.

The graPHIGS API returns a value indicating the number of predefined text indexes for the bundle table of the specified workstation type.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

### Parameters

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*npred* **— returned by the graPHIGS API, fullword integer**
Number of predefined text bundle table entries.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**GPQXTX**
Inquire Extended Text Facilities

**RCP code**

201339656 (X'0C003308')

## GPQWCT - Inquire Workstation Connection and Type

**GPQWCT (*wsid*, *errind*, *connid*, *wstype*)**

**Purpose**

Use **GPQWCT** to inquire the connection identifier and generic workstation type of the specified workstation identifier. The generic workstation type is the type that was developed from the Open Workstation (**GPOPWS**) subroutine or Create Workstation (**GPCRWS**) subroutine or the type that resulted from Nickname Default processing.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**533** INQUIRY DATA EXCEEDS AREA. OUTPUT TRUNCATED

*connid* **— returned by the graPHIGS API, 8-byte character string**
Connection identifier.

**Note:** It the connection identifier is >8 characters, then it is truncated and the *errind* parameter is set to 533.

*wstype* **— returned by the graPHIGS API, 8-byte character string**
Generic workstation type.

**Error Codes**

None

**Related Subroutines**

**GPQRCT**
Inquire Realized Connection Type

**RCP code**

201336841 (X'0C002809')

## GPQWCV - Inquire Workstation Configuration Variability

**GPQWCV (*wstype*, *start*, *number*, *errind*, *totnum*, *flist*)**

**Purpose**

Use **GPQWCV** to inquire the items that may vary based on the configuration of the specified workstation type.

The graPHIGS API returns a value indicating the total number of workstation dependent features and a list of their identifiers.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*wstype* **— specified by user, 8-byte character string**
Workstation type.

*start* **— specified by user, fullword integer**
Starting member of the list of workstation variable features (>=1).

*number* **— specified by user, fullword integer**
Number of requested feature identifiers (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**23** SPECIFIED WORKSTATION TYPE DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

**548** SPECIFIED WORKSTATION TYPE CANNOT BE LOADED

*totnum* **— returned by the graPHIGS API, fullword integer**
Total number of configuration-dependent features.

*flist* **— returned by the graPHIGS API, array of fullword integers.**
List of the configuration-dependent features. The output array must be large enough to contain the requested data.

The following items represent possible variable features:

**01** Workstation category

**02** Maximum display surface size in device coordinates

**03** Maximum display surface size in addressable units

**04** Number of definable view table entries

**05** Polyline linetypes

**06** Polyline line widths

**07** Number of polyline bundle table entries

**08** Polymarker marker types

**09** Polymarker marker sizes

**10** Number of polymarker bundle table entries

**11** Workstation character set facilities

**12** Number of text bundle table entries

**13** Interior style indexes

**14** Number of interior bundle table entries

**15** Number of edge bundle table entries

**16** Line widths of edges

**17** Number of edge bundle table entries

**18** Pattern facilities

**19** Hatch facilities

**20** Number of color table entries

**21** Color availability

**22** Color model

**23** Generalized drawing primitives

**24** Dynamic modification accepted

**25** Locator input devices

**26** Stroke input devices

**27** Valuator input devices

**28** Choice input devices

**29** Pick input devices

**30** String input devices

**31** Maximum pattern array dimensions

**32** Workstation display classification

**33** (*reserved*)

**34** Available break action

**35** Default break action

**36** Supported escape identifiers

**37** Available triggers for locator devices

**38** Available triggers for stroke devices

**39** Available triggers for valuator devices

**40** Available triggers for choice devices

**41** Available triggers for pick devices

**42** Available triggers for string devices

**43** Default triggers for locator devices

**44** Default triggers for stroke devices

**45** Default triggers for valuator devices

**46** Default triggers for choice devices

**47** Default triggers for pick devices

**48** Default triggers for string devices

**49** Maximum number of choice alternatives

**50** Available triggers on the workstation.

### Error Codes

None

### Related Subroutines

None

### RCP code

201339398 (X'0C003206')

## GPQWST - Inquire List of Available Workstation Types

**GPQWST (*start*, *number*, *errind*, *maxopen*, *nwstype*, *wstype*)**

### Purpose

Use **GPQWST** to inquire a list of available generic workstation types that can be open on a nucleus with nucleus identifier=1.

The graPHIGS API returns a value indicating the number of workstation types and a list of those types.

If the inquired information is available, then the graPHIGS API sets the error indicator to zero and returns the values in the output parameters. If the error indicator is 543 (the start value exceeds the extent of the available data), then only the total number (*totnum*) parameter is set. If the inquired information is unavailable, then the error indicator (*errind*) contains an error number indicating the reason, and the values returned in the output parameters are unpredictable.

**Parameters**

*start* **— specified by user, fullword integer**
Starting member of the list of workstation types (>=1).

*number* **— specified by user, fullword integer**
Number of workstation types requested (>=0).

*errind* **— returned by the graPHIGS API, fullword integer**
Error indicator. If the error indicator is zero, the request has been completed. Otherwise, one of the following errors exists:

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**538** START VALUE < ONE

**539** REQUESTED NUMBER < ZERO

**543** START EXCEEDS DATA EXTENT. TOTAL NUMBER AVAILABLE RETURNED

*maxopen* **— returned by the graPHIGS API, fullword integer**
Maximum number of workstations to which a structure store can be simultaneously associated.

*nwstype* **— returned by the graPHIGS API, fullword integer**
Total number of available generic workstation types.

*wstype* **— returned by the graPHIGS API, array of 8-byte character strings**
List of available generic workstation types. The output array must be large enough to contain the requested data.

**Error Codes**

None

**Related Subroutines**

**GPQWTN**
Inquire List of Available Workstation Types on Nucleus

**RCP code**

201336325 (X'0C002605')

## GPTXR - Set Text Representation

**GPTXR (*wsid*, *index*, *font*, *prec*, *factor*, *space*, *color*)**

**Purpose**

Use **GPTXR** to set the given attribute values into the specified entry of the text bundle table.

If a precision is specified that is not supported by the workstation, that workstation substitutes the font's highest available precision.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*index* **— specified by user, fullword integer**
Text bundle table index (>=1). Index of table entry to be loaded.

*font* **— specified by user, fullword integer**
Text font (>=1).

*prec* **— specified by user, fullword integer**
Text precision (1=STRING_PREC, 2=CHAR_PREC, 3=STROKE_PREC).

*factor* **— specified by user, short floating-point number**
Character expansion factor (>0). Specifies the deviation of the character's width-to-height ratio from the ratio specified by the font designer.

*space* **— specified by user, short floating-point number**
Character spacing between two adjacent characters. This is specified as a fraction of the character's height.

*color* **— specified by user, fullword integer**
Text color index (>=0).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**35** WORKSTATION HAS ONLY INPUT CAPABILITIES

**43** BUNDLE INDEX EXCEEDS WORKSTATION TABLE CAPACITY

**60** BUNDLE INDEX VALUE < ONE

**75** TEXT FONT VALUE IS INVALID

**77** CHARACTER EXPANSION FACTOR <= ZERO

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**305** TEXT PRECISION VALUE IS INVALID

**Related Subroutines**

**GPXTXR**
Set Extended Text Representation

**RCP code**

201329411 (X'0C000B03')

## GPVCH - Set View Characteristics

**GPVCH (*wsid*, *view*, *window*, *near*, *far*, *shield*, *shldci*, *border*, *brdrci*, *active*)**

### Purpose

Use **GPVCH** to set the characteristics of the specified view. These characteristics include: clipping indicators, appearance of the viewport, and a value indicating whether or not the view is displayed.

The values specified are stored in the requested view table entries. The corresponding current values in the view table entry are set to the requested values when the workstation is updated.

The clipping indicators determine to which boundaries the contents of the view are clipped. The shielding indicator determines whether the content of lower priority views may be displayed within the boundaries of the specified view. The border indicator and color specify whether a border is to be drawn around the viewport, and if it is, its color. The view active flag determines whether the view and its contents are displayed.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
View index (>=1). Index of view table entry to be loaded.

*window* **— specified by user, fullword integer**
Window clipping indicator (`1=NOCLIP, 2=CLIP`).

*near* **— specified by user, fullword integer**
Near clipping indicator (`1=NOCLIP, 2=CLIP`).

*far* **— specified by user, fullword integer**
Far clipping indicator (`1=NOCLIP, 2=CLIP`).

*shield* **— specified by user, fullword integer**
Shielding indicator (`1=OFF, 2=ON`).

*shldci* **— specified by user, fullword integer**
Shielding color index (>=0).

*border* **— specified by user, fullword integer**
View border indicator (`OFF, 2=ON`).

*brdrci* **— specified by user, fullword integer**
Border color index (>=0).

*active* **— specified by user, fullword integer**
View active flag for output (`1=INACTIVE, 2=ACTIVE`).

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**92** COLOR INDEX < ZERO

**93** COLOR INDEX VALUE(S) EXCEED WORKSTATION TABLE CAPACITY

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**332** CLIP INDICATOR VALUE IS INVALID

**507** SHIELDING INDICATOR VALUE IS INVALID

**508** VIEW ACTIVE FLAG VALUE IS INVALID

**518** VIEW ZERO CANNOT BE MODIFIED

**547** VIEW BORDER INDICATOR IS INVALID

**Related Subroutines**

**GPXVR**
Set Extended View Representation

**RCP code**

201330433 (X'0C000F01')

## GPVMP2 - Set View Mapping 2

**GPVMP2 (*wsid*, *view*, *window*, *viewpt*)**

**Purpose**

Use **GPVMP2** to set the requested window and viewport values (for *x* and *y*) to the specified values for the given view.

Other viewing parameters are automatically set with the **GPVMP2** subroutine and are calculated as follows:

- The *z* extents for the viewport are set to the *z* extents of the NPC range.
- The projection type is set to parallel.
- The projection reference point is placed on a line perpendicular to the center of the specified window. The *z* value of the projection reference point is set to one half of the maximum of the Umax-Umin and Vmax-Vmin.
- The view plane distance is set to zero.
- The far clipping plane is set to the negative of one-half the maximum of Umax-Umin and Vmax-Vmin.
- The near clipping plane is set to one-half of the maximum of Umax-Umin and Vmax-Vmin.

All current view mapping values are set to the requested values when the workstation is updated.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
Index of view table entry to be modified (>=1).

*window* **— specified by user, 4 short floating-point numbers (VC)**
Window limits (Umin, Umax, Vmin, Vmax).

*viewpt* **— specified by user, 4 short floating-point numbers (NPC)**
Viewport limits (Xmin, Xmax, Ymin, Ymax).

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**44** INVALID WINDOW DEFINITION

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**330** INVALID VIEWPORT

**518** VIEW ZERO CANNOT BE MODIFIED

**Related Subroutines**

**GPXVR**
Set Extended View Representation

**RCP code**

201330178 (X'0C000E02')

## GPVMP3 - Set View Mapping 3

**GPVMP3 (*wsid*, *view*, *window*, *viewpt*, *type*, *point*, *dist*, *near*, *far*)**

**Purpose**

Use **GPVMP3** to set the requested viewing parameters to the given values for the specified view. The current values are set to the requested values when the workstation is updated.

**Parameters**

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
Index of view table entry to be modified (>=1).

*window* **— specified by user, 4 short floating-point numbers (VC)**
Window limits (Umin, Umax, Vmin, Vmax).

*viewpt* **— specified by user, 6 short floating-point numbers (NPC)**
Viewport limits (Xmin, Xmax, Ymin, Ymax, Zmin, Zmax).

*type* **— specified by user, fullword integer**
Projection type (`1=PARALLEL, 2=PERSPECTIVE`).

*point* **— specified by user, 3 short floating-point numbers (VC)**
Projection reference point (*u*, *v*, *n*).

*dist* **— specified by user, short floating-point number (VC)**
Distance of view plane from view reference point along the n-axis.

*near* **— specified by user, short floating-point number (VC)**
Distance of near plane from view reference point along the n-axis.

*far* **— specified by user, short floating-point number (VC)**
Distance of far plane from view reference point along the n-axis.

**Error Codes**

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**44** INVALID WINDOW DEFINITION

**55** PRP IS POSITIONED ON THE VIEW PLANE

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**330** INVALID VIEWPORT

**331** PROJECTION TYPE IS INVALID

**336** FAR CLIPPING PLANE IN FRONT OF NEAR CLIPPING PLANE

**518** VIEW ZERO CANNOT BE MODIFIED

### Related Subroutines

**GPXVR**
Set Extended View Representation

### RCP code

201330177 (X'0C000E01')

## GPVMT2 - Set View Matrix 2

**GPVMT2 (*wsid*, *view*, *matrix*)**

### Purpose

Use **GPVMT2** to load the requested view matrix into the specified view table entry.

The 3x3 matrix is expanded by the graPHIGS API into a 4x4 matrix as follows:

```
|a b c|         |a b 0 c|
|d e f|  ---->  |d e 0 f|
|g h i|         |0 0 1 0|
                |g h 0 i|
```

The current value is set to the requested value when the workstation is updated.

When inquired, the matrix returned is the expanded 4 x 4 matrix.

### Parameters

*wsid* **— specified by user, fullword integer**
Workstation identifier.

*view* **— specified by user, fullword integer**
Index of view table entry to be loaded (>=1).

*matrix* **— specified by user, 9 short floating-point numbers**
View matrix.

For the input view matrix, the elements must be in the following order:

```
|m11 m12 m13|
|m21 m22 m23|---> (m11,m12,m13,m21.....m33)
|m31 m32 m33|
```

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**518** VIEW ZERO CANNOT BE MODIFIED

**522** VIEW MATRIX IS SINGULAR

### Related Subroutines

**GPXVR**
Set Extended View Representation

### RCP code

201329922 (X'0C000D02')

## GPVMT3 - Set View Matrix 3

**GPVMT3 (*wsid*, *view*, *matrix*)**

### Purpose

Use **GPVMT3** to load the requested view matrix into the specified view table entry. For the specified workstation, the current value is set to the requested value when the workstation is updated.

### Parameters

*wsid* — **specified by user, fullword integer**
Workstation identifier.

*view* — **specified by user, fullword integer**
Index of view table entry to be loaded (>=1).

*matrix* — **specified by user, 16 short floating-point numbers**
View matrix.

For the input view matrix, the elements must be in the following order:

```
|m11 m12 m13 m14|
|m21 m22 m23 m24|---> (m11,m12,m13,m14,m21,m22,...m44)
|m31 m32 m33 m34|
|m41 m42 m43 m44|
```

### Error Codes

**25** SPECIFIED WORKSTATION DOES NOT EXIST

**59** VIEW INDEX VALUE < ZERO

**323** VIEW INDEX EXCEEDS VIEW TABLE CAPACITY

**518** VIEW ZERO CANNOT BE MODIFIED

**522** VIEW MATRIX IS SINGULAR

### Related Subroutines

**GPXVR**
Set Extended View Representation

### RCP code

201329921 (X'0C000D01')

# Chapter 19. Distributed Application Processing (DAP)

The subroutines in this section initiate and terminate a distributed application process.

## GPEXAP - Execute Application Process

**GPEXAP (*apid*, *ncid*, *size*, *namel*, *name*, *parmt*, *parml*, *parm*, *xferflag*, *useridl*, *userid*, *passwordl*, *password*)**

### Purpose

Use **GPEXAP** to execute a Distributed Application Process (DAP) through a connected nucleus on a local or remote node. Depending upon the value of the transfer and/or execute parameter (*xferflag*), the DAP specified through the *name* parameter either is executed if the DAP already resides on the node or is transferred to the nucleus's node and then is executed.

If the *xferflag* parameter is set to `2=TRANSFER_EXECUTE`, then the *name* parameter is interpreted to be the path and filename of the DAP object code on the shell's file system. DAPs transferred to the 6095 nucleus's file system are loaded into memory. DAPs transferred to the RS/6000® nucleus's file system are placed in a unique temporary directory and erased when the DAP is terminated. By default this will be the **/tmp/.gP/ *xxxxx*** directory where *xxxxx* is the concatenation of the nucleus identifier and the DAP resource identifier. This will insure that the path to the DAP is unique for all DAPs on a RS/6000®. In order to provide the user control of the size of the directory that will contain the downloaded DAPs, you may override **/tmp/.gP/** with your own path by specifying the `DAPPATH` default in the External Defaults File (EDF) or by using the **-p** option of the gPinit command (see *The graPHIGS Programming Interface: Technical Reference* for information on the gPinit command).

If the *xferflag* parameter is set to `1=EXECUTE_ONLY`, then the *name* parameter is interpreted to be the path and filename of the DAP object code on the nucleus's file system.

For DAPs targeted for the RS/6000®, where the *xferflag* is set to either `1=EXECUTE_ONLY` or `2=TRANSFER_EXECUTE`, a *userid* and *password* on the nucleus's node must be provided for the operation to be performed. The DAPs must be prelinked with any required libraries.

For DAPs targeted for the IBM 6095, the *xferflag*, *userid*, and *password* parameters are ignored and the DAP is transferred and executed in a manner equivalent to the Initiate Application Process (**GPINAP**) subroutine. The DAPs must be prelinked with any required nonshared libraries. System memory with the specified size is allocated, the specified program module is loaded into the memory and the module is started with the passing of the specified data as its parameter. The program module must reside on a disk that is accessible from the shell as a single file with a format that can be loaded by the control program in the 6090.

As with the **GPINAP** subroutine, DAPs started with the **GPEXAP** subroutine can be terminated with the Terminate Application Process (**GPTMAP**) subroutine.

### Parameters

*apid* **— specified by user, fullword integer**
Application process identifier.

*ncid* **— specified by user, fullword integer**
Nucleus identifier.

*size* **— specified by user, fullword integer**
Size of the memory allocated for the application process in bytes (>=0) The size must be large

enough to contain the program module itself and all storage used by the application process. This parameter is ignored if the nucleus is running on a RS/6000®.

*namel* **— specified by user, fullword integer**
Length of the file name in bytes (0<=*namel*<=80).

*name* **— specified by user, variable length character string**
Name of the file that contains the program module of the application process. If the *xferflag* parameter is set to `1=EXECUTE_ONLY`, then *name* is interpreted to be the path and filename of the DAP object code on the nucleus's file system. If the *xferflag* parameter is set to `2=TRANSFER_EXECUTE`, then *name* is interpreted to be the path and filename of the DAP object code on the shell's file system. This parameter looks like a Unix file descriptor which consists of a **[/path/]filename[.extension]**. Path is the route of directories through the file system. It is optional and is ignored for MVS and VM.

Following is an example of a full file descriptor:

**/dap/appl1.o**

- path = **/dap/** which says go from the root directory to directory **dap**
- filename = **appl1**
- extension = **.o**

If the xferflag parameter is set to `1=EXECUTE_ONLY`, then the name is passed to the nucleus as specified by the application. If the xferflag parameter is set to `2=TRANSFER_EXECUTE`, then the following rules apply to the *name* parameter depending on which system the shell is running:

- AIX®
  - `filename` - The filename must be supplied.
  - `extension` - The extension is optional.
  - `path` - The path is optional. If the path is not included, then the default directory at the time of the execution of the function will be used.
- MVS, MVS/XA
  - `filename` - The filename must be supplied and it is a member name within a partition data set if the extension is included. If the extension is not included, then the filename is the DD-name of the partition data set.
  - `extension` - The extension is optional. If specified it is the DD-name of the partition data set including the member filename.
  - the recommended file format for DAP object files targeted for AIX® is `V` or `VB` and for the 6095 is `F` or `FB`.
  - the recommended record length for DAP object files targeted for AIX® is 2052 and for the 6095 is 80.
- VM/CMS
  - `filename` - The filename must be supplied.
  - `extension` - The extension is optional. If it is not included, then an extension of `.COF` is used.
  - the `filename.extension` will be folded to upper case.
  - the recommended file format for DAP object files targeted for AIX® is `V` and for the 6095 is `F`.
  - the recommended record length for DAP object files targeted for AIX® is 8192 and for the 6095 is 80.

*parmt* **— specified by user, fullword integer**
Format of the parameter data and method of passing it to the main entry point of the application process. The following options are available:

**0 =** No parameters are to be passed to the application process. Parameters *parml* and *parm* are ignored.

**1 =** The parameter data consists of a single string. This string will be split into strings which contains no white space (blanks, newline, tabs, etc.) before being passed to the application process. The application entry point will receive two parameters. The first parameter will contain the number of strings resulting from the parsing operation described above. The second parameter will contain a list of pointers to each of the strings. The main entry declaration could be coded in "C" as follows:

```
void main( argc, argv )
   int    argc;
   char * argv[[default]
```

On a RS/6000®, the path and filename of the executing DAP is:

```
arg[o]
```

The arguments specified by the *parm* parameter are:

```
arg[1..n]
```

On an IBM 6095 system, the arguments specified by the *parm* parameter are:

```
arg[0.. n]
```

**2 =** The parameter data contains arbitrary information which has content that is only understood by the application processes. A pointer to the data and the amount of data is passed to the application process when it is started. The main entry declaration could be coded in "C" as follows:

```
void main( parmlen, parmptr )
   int    parmlen;
   char * parmptr;
```

**Note:** This parameter is **not** supported by a nucleus running on a RS/6000® and if it is specified an error will be generated. For this option, it is the application's responsibility to perform character code, floating-point, and byte order conversion, if needed, between the two environments in which the application processes are executing.

*parml* **— specified by user, fullword integer**
Length of data to be passed to the process as its invocation parameter in bytes (>=0).

*parm* **— specified by user, variable length character or byte string**
Data to be passed to the process as its invocation parameter. The format of the data contained in this parameter is defined by *parmt*.

*xferflag* **— specified by user, fullword integer**
Transfer and/or execute flag (`1=EXECUTE_ONLY`, `2=TRANSFER_EXECUTE`). Only option `2=TRANSFER_EXECUTE` is valid if the nucleus is running on a IBM 6095 system.

*useridl* **— specified by user, fullword integer**
Length of *userid* string. This parameter is ignored if the nucleus is running on a IBM 6095 system.

*userid* **— specified by user, variable length character string**
Userid of a user on the nucleus's node. This parameter is ignored if the nucleus is running on a IBM 6095 system.

**Notes:**

1. The nucleus targeted to execute the DAP must be a remote nucleus and be enabled to support DAP execution (see gPinit in *The graPHIGS Programming Interface: Technical Reference*.
2. The userid specified though the *userid* parameter, does ***not*** have to be the userid under which the nucleus is running.
3. The DAP will be executed under the specified userid.

*passwordl* **— specified by user, fullword integer**
Length of *password* string. This parameter is ignored if the nucleus is running on a IBM 6095 system.

*password* **— specified by user, variable length character string**
The password corresponding to the userid supplied through the *userid* parameter. This parameter is ignored if the nucleus is running on an IBM 6095 system.

> **Note:** Operating system passwords within the AFS system cannot be verified and will result in error 210. In AFS, only locally defined passwords are accessible to the graPHIGS nucleus for validation.

**Error Codes**

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**210** RESOURCE CREATION DETECTED AN INVALID USERID/PASSWORD

**217** RESOURCE CREATION REQUEST EXCEEDS NUCLEUS TABLE CAPACITY

**1132** RESOURCE CREATION AFS USERID/PASSWORD VALIDATION SUBSYSTEM TIMEOUT

**1133** RESOURCE CREATION REQUIRED AN AFS TOKEN THAT DOES NOT EXIST

**1301** SPECIFIED APPLICATION PROCESS ID ALREADY IN USE

**1303** SIZE OF APPLICATION PROCESS REGION IS TOO LARGE

**1304** APPLICATION PROCESS REQUEST EXCEEDS NUCLEUS CAPACITY

**1305** LENGTH OF APPLICATION MODULE NAME IS INVALID

**1307** APPLICATION MODULE HAS UNRESOLVED EXTERNAL REFERENCE *a1*

**1308** PARAMETER TYPE IS INVALID

**1309** PARAMETER LENGTH < ZERO

**1310** APPLICATION MODULE SIZE > REGION SIZE

**1311** APPLICATION LOAD MODULE IS INVALID

**1316** APPLICATION PROCESS ID=*n1* EXITED WITH CODE=*n2*

**1317** FLAG PARAMETER IS INVALID

**Related Subroutines**

**GPQNCE**
Inquire Nucleus Environment

**RCP code**

201342468 (X'0C003E04')

## GPINAP - Initiate Application Process

**GPINAP (*apid*, *ncid*, *size*, *namel*, *name*, *parmt*, *parml*, *parm*)**

**Purpose**

Use **GPINAP** to create an application process on a 6090.

System memory of the specified size is allocated, the specified program module is loaded into the memory and the module is started with the specified parameter data being passed.

This subroutine is applicable only to a nucleus running on a 6090. The program module must reside on a disk that is accessible from the shell as a single file with a format that can be loaded by the control program in the 6090. For details of the file format, see *The graPHIGS Programming Interface: Technical Reference*.

**Parameters**

*apid* **— specified by user, fullword integer**
Application process identifier.

*ncid* **— specified by user, fullword integer**
Nucleus identifier. The specified nucleus must be running on a 6090.

*size* **— specified by user, fullword integer**
Size of the memory allocated for the application process in bytes (>0). The size must be large enough to contain the program module itself and all storage used by the application process.

*namel* **— specified by user, fullword integer**
Length of the file name in bytes (0<*namel*<=80).

*name* **— specified by user, character string**
Name of the file that contains the program module of the application process. This parameter looks like a Unix file descriptor which consists of a **[path]/filename[extension]**. **Path** is the route of directories through the file system. It is optional and is ignored for MVS and VM.

Following is an example of a full file descriptor:

**/dap/appl1.o**

- path = **/dap** which says go from the root directory to directory **dap**
- filename = **appl1**
- extension = **.o**

The following rules apply to the *name* parameter depending on which system the shell is running:

- AIX®

  If you do not specify the path, then the graPHIGS API uses the default directory at the time of the execution of the subroutine.
- MVS, MVS/XA
  - `filename` - The filename must be supplied and it is a member name within a partition data set if the extension is included. If the extension is not included, then the filename is the DD-name of the partition data set.
  - `extension` - The extension is optional. If specified it is the DD-name of the partition data set including the member filename.
- VM/CMS
  - `filename` - The filename must be supplied.
  - `extension` - The extension is optional. If it is not included an extension of .COF will be used.
  - the `filename.extension` will be folded to upper case.

*parmt* **— specified by user, fullword integer**
Format of the parameter data and method of passing it to the 'main' entry point of the application process. The following options are available:

**0 =** No parameters are to be passed to the application process. Parameters *parml* and *parm* are ignored.

**1 =** The parameter data consists of a single string. This string will be split into strings which

contain no white space (blanks, newline, tabs, etc.) before being passed to the application process. The application entry point will receive two parameters. The first parameter will contain the number of strings resulting from the parsing operation described above. The second parameter will contain a list of pointers to each of the strings. The main entry declaration could be coded in "C" as follows:

```
void main( argc, argv )
   int    argc;
   char * argv[[default]
```

**2 =** The parameter data contains arbitrary information which has content that is only understood by the application processes. A pointer to the data and the amount of data is passed to the application process when it is started. The main entry declaration could be coded in "C" as follows:

```
void main( parmlen, parmptr )
   int    parmlen;
   char * parmptr;
```

**Note:** For this option, it is the application's responsibility to perform character code, floating-point, and byte order conversion between the two environments in which the application processes are executing.

*parml* **— specified by user, fullword integer**
Length of data to be passed to the process as its invocation parameter in bytes (>=0).

*parm* **— specified by user, variable length character or byte string**
Data to be passed to the process as its invocation parameter. The format of the data contained in this parameter is defined by *parmt*.

**Error Codes**

**202** SPECIFIED NUCLEUS DOES NOT EXIST

**1301** SPECIFIED APPLICATION PROCESS ID ALREADY IN USE

**1303** SIZE OF APPLICATION PROCESS REGION IS TOO LARGE

**1304** APPLICATION PROCESS REQUEST EXCEEDS NUCLEUS CAPACITY

**1305** LENGTH OF APPLICATION MODULE NAME IS INVALID

**1307** APPLICATION MODULE HAS UNRESOLVED EXTERNAL REFERENCE *a1*

**1308** PARAMETER TYPE IS INVALID

**1309** PARAMETER LENGTH < ZERO

**1310** APPLICATION MODULE SIZE > REGION SIZE

**1311** APPLICATION LOAD MODULE IS INVALID

**1316** APPLICATION PROCESS ID=*n1* EXITED WITH CODE=*n2*

**Related Subroutines**

**GPEXAP**
Execute Application Process

**GPTMAP**
Terminate Application Process

**GPQNCE**
Inquire Nucleus Environment

### RCP code

201342465 (X'0C003E01')

## GPTMAP - Terminate Application Process

**GPTMAP (*apid*)**

### Purpose

Use **GPTMAP** to terminate an application process.

The specified application process is terminated (if still running) and all system resources used by the application process are reclaimed.

### Parameters

*apid* **— specified by user, fullword integer**
Application process identifier.

### Error Codes

**1302** SPECIFIED APPLICATION PROCESS ID DOES NOT EXIST

### Related Subroutines

**GPEXAP**
Execute Application Process

**GPINAP**
Initiate Application Process

### RCP code

201342466 (X'0C003E02')

# Appendix A. Character Set and Font Identifiers

**Character Set Identifiers are defined as follows:**

| | |
|---|---|
| **1-100** | Reserved for IBM-defined single-byte character codes. |
| | **1-** US ENGLISH |
| | **2-** UK ENGLISH |
| | **3-** GERMAN |
| | **4-** FRENCH |
| | **5-** ITALIAN |
| | **6-** KATAKANA |
| | **7-** SWEDISH |
| | **8-** MULTINATIONAL |
| | **10-** ISO 8859-1 (LATIN-1) |
| | **11-** ISO 8859-2 (CZECH) |
| | **12-** ISO 8859-5 (CYRILLIC) |
| **101-127** | Reserved for your use for single-byte character sets (only for geometric text). |
| **128-228** | Reserved for IBM defined double-byte character codes. |
| **128** | KANJI (IBM-932 encoding) |
| **129** | HANGUL |
| **130** | TRADITIONAL CHINESE |
| **131** | UNICODE |
| **132** | SIMPLIFIED CHINESE |
| **134** | KANJI (IBM-943 encoding) |
| **229-255** | Reserved for your use for double-byte character sets (only for geometric text). |

**Font Identifiers**

| | |
|---|---|
| **1-127** | Reserved for IBM defined fonts within IBM-reserved character sets; available for your use in user-reserved character sets. |
| **128-255** | Reserved for your use (only for geometric text) |

For more information, see *The graPHIGS Programming Interface: Understanding Concepts* and *The graPHIGS Programming Interface: Technical Reference*.

# Appendix B. Error Processing

The specific errors detected by the graPHIGS API subroutines are listed with the appropriate subroutine syntax.

The graPHIGS API error numbers are divided into the following ranges:

**ERRORS 1-899 BASE graPHIGS API**
Device-independent warnings and errors.

**ERRORS 900-999 DEVICE SUPPORT**
Workstation specific warnings and errors.

**ERRORS 1000-1299 SYSTEM SERVICE**
System service warnings and errors.

**ERRORS 1300-1399 DAP**
Distributed application processing warnings and errors.

**ERRORS 2000-2999 DEVICE DRIVER**
Device driver warning and errors.

The timing of notification of the error is controlled by several factors, such as deferral mode or availability of system resources. Although errors are listed with each subroutine, the error may actually be reported by the processing of another subroutine. For example:

- Errors detected within the nucleus may be reported to the application on later, unrelated subroutines.
- Errors in view specifications (such as error 522 when a view matrix specified on the Set View Mapping 3 [**GPVMP3**] [page GPVMP3 - Set View Mapping 3] subroutine is singular) may not be detected and reported to the application until the incorrect data is used by the Update Workstation (**GPUPWS**) [page GPUPWS - Update Workstation] subroutine.
- I/O errors may be reported on subroutines that do not appear to perform I/O, or may perform I/O that is unrelated to the reported I/O error.
- Since deferral mode affects the timing of workstation updates, error conditions in some subroutines may not be detected until later subroutines force an update. This can cause errors from the former subroutines to be reported to the application on the later, unrelated subroutines.

In addition, there are certain errors that, although possible on any graPHIGS API subroutine, are not listed with each subroutine. These errors are:

| | |
|---|---|
| **2** | FUNCTION REQUIRES STATE PHOP |
| **300** | STORAGE REQUEST FAILED |
| **520** | ERROR QUEUE HAS OVERFLOWED |
| **525** | FUNCTION CANNOT BE CALLED IN ERROR STATE |
| **556** | ELEMENT EXCEEDS MAXIMUM ALLOWED SIZE |
| **594** | DATA EXCEEDS CONNECTION BUFFER SIZE |
| **603** | INTERNAL COMMUNICATIONS PROTOCOL ERROR |
| **607** | NUCLEUS IS DOWN LEVEL. VERSION @A1, RELEASE @A2.@A3 IS REQUIRED |

Messages from the workstation service routines (Messages 2000-2999) and/or system services (Messages 1000-1299) are possible on many subroutines. For additional information on errors and error processing states, see *The graPHIGS Programming Interface: Understanding Concepts*.

# Appendix C. Error-to-Subroutine Reference

The following table presents the error numbers listed in this reference manual, followed by a list of the possible subroutines that can generate a given error:

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 1 | GPOPPH |
| 3 | GPBGTR GPCARR GPCPVP GPCRT GPCVW GPDMR GPDRT GPDRVW GPIDMO GPQACF GPSRT |
| 4 | GPCEDT GPCLST GPCPER GPCPST GPDELB GPDLE GPDLEG GPDLER GPEP GPEPCD GPEPLB GPEPLG GPEPPG GPEPPK GPMVER GPOEP GPQDMR GPQE GPQED GPQEHD GPQEP GPQETS GPRNBS GPRTNS |
| 5 | GPAAL GPADCN GPAH GPAHSC GPAID GPAN2 GPAN3 GPANR2 GPANR3 GPAPT GPAS GPASF GPAUP GPBBLF GPBDFM GPBDMF GPBDMI GPBDM2 GPBICD GPBICI GPBISM GPBLF GPBRMO GPBSCD GPBSCI GPBSPR GPBTCO GPCAC GPCEXS GPCFA2 GPCHH GPCHLS GPCHL2 GPCHPM GPCHSP GPCHUB GPCHUP GPCHXP GPCOND GPCPI GPCR2 GPCRA2 GPCRET GPDCI GPDFM GPDMF GPDPL2 GPDMF GPDMI GPDM2 GPDPL3 GPECD GPECI GPEF GPEI GPELA2 GPELA3 GPELT GPEL2 GPEL3 GPESC GPEXST GPFBC GPFBM GPFLM GPFDMO GPGLX2 GPGLX3 GPHID GPHLCD GPHLCI GPICD GPICI GPII GPINAD GPINLB GPIS GPISI GPISM GPLG2 GPLG3 GPLLCD GPLLCI GPLMO GPLSS GPLT GPLWSC GPMCI GPMCV2 GPMCV3 GPMG2 GPMG3 GPMLX2 GPMLX3 GPMSSC GPMT GPNBC2 GPNBC3 GPNBS GPPGC GPPGD2 GPPGD3 GPPG2 GPPG3 GPPHE GPPHEC GPPKID GPPLCD GPPLCI GPPLD3 GPPLET GPPLI GPPL2 GPPL3 GPPLSM GPPMCD GPPMCI GPPMI GPPM2 GPPM3 GPPSC GPPXL2 GPPXL3 GPQM3 GPRCN GPRMCV GPRMO GPSAC GPSCD GPSCI GPSPH GPSPR GPTCAC GPTCO GPTEX2 GPTEX3 GPTNBS GPTS3 GPTXAL GPTXCD GPTXCI GPTXFO GPTXI GPTXPR GPTXPT GPTX2 GPTX3 GPVMF GPVWI GPVMF GPWDO GPZBM |
| 7 | GPARAS GPARSN GPARST GPCLAR GPDASA GPDSAR GPDSNA GPQACA GPQCNA GPRAS GPRDS GPRISN GPRSTI GPRVAS GPRVSN GPRVST |
| 11 | GPBGST GPOPST |
| 12 | GPARAS GPARSN GPARST GPARW GPCSI GPCSIR GPCSRS GPDARW GPDAST GPDLNC GPDLNT GPDLST GPDRAV GPDRW GPELS GPEST GPRVAS GPRVSN GPRVST GPQACA GPQACS GPQCNA GPQCSN GPQEDA GPQEHA GPQEXS GPQISN GPQPAS GPQPDS GPQRST GPQSTI GPQSTS GPQWSA GPTAST GPTST |
| 13 | GPSSS |
| 14 | GPBGST GPDRI2 GPENTR |
| 15 | GPATS GPENST GPPSTS GPPTS GPTE |
| 16 | GPDRI2 GPENTR |
| 21 | GPCRWS GPOPWS |
| 23 | GPCRWS GPOPWS GPQAAF GPQAMO GPQANF GPQART GPQBK GPQCDF GPQCF GPQCPF GPQCQM GPQCSF GPQCUF GPQDBK GPQDCF GPQDCH GPQDDV GPQDIT GPQDLC GPQDPK GPQDS GPQDSK GPQDST GPQDVL GPQEF GPQES GPQFBC GPQFP GPQGD GPQGDP GPQGSE GPQHD GPQHF GPQHMO GPQIDF GPQIF GPQIMF GPQISF GPQIT GPQLCF GPQLI GPQLNR GPQLSF GPQLTF GPQLW GPQMTF GPQNCN GPQNSP GPQNST GPQNV GPQPAF GPQPCR GPQPCS GPQPDC GPQPER GPQPIR GPQPKT GPQPLF GPQPLR GPQPMF GPQPMR GPQPPR GPQPTR GPQRCM GPQSDF GPQSPD GPQTDF GPQTMO GPQTXF GPQVF GPQWC GPQWCV GPQWD GPQWDT GPQXCF GPQXTX |
| 24 | GPATR GPCRWS GPOPWS |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 25 | GPACFO GPAFDW GPARV GPARW GPASSW GPBGTR GPBKAC GPCAI GPCARR GPCCV GPCHMO GPCIM2 GPCIM3 GPCLWS GPCML GPCPR GPCPVP GPCR GPCRC GPCRT GPCSR GPCUR GPCUS GPCVW GPDAFO GPDARW GPDCMM GPDCR GPDF GPDFI GPDIM GPDLC GPDMR GPDRAV GPDRT GPDRV GPDRVW GPDRW GPDTR GPEAV GPEPD GPER GPEV GPFLEV GPFWEV GPGTXC GPHLF GPHR GPICS GPIDMO GPIEC GPINCH GPINLC GPINPK GPINSK GPINST GPINVL GPIPKC GPIR GPIT GPIVF GPLCMO GPLNR GPLSR GPLTR GPMSG GPMTR GPPAR GPPDMO GPPKAP GPPKF GPPKMO GPPKSC GPPLR GPPMR GPPW GPQABK GPQACF GPQADS GPQAEF GPQAES GPQAFC GPQAFP GPQAGD GPQAIF GPQAIS GPQAIT GPQALF GPQALI GPQALW GPQAMF GPQANV GPQAPF GPQAPS GPQAR GPQAVF GPQAWC GPQAWD GPQBKS GPQCCH GPQCH GPQCID GPQCML GPQCPR GPQCR GPQCSR GPQCVE GPQCVO GPQCVR GPQCVX GPQDCR GPQDMR GPQDV GPQER GPQFO GPQGFC GPQHLF GPQHR GPQICH GPQICS GPQID GPQIMC GPQIMI GPQIMV GPQIMW GPQIR GPQITS GPQIVF GPQIW GPQLC GPQLR GPQLSR GPQLTR GPQMDS GPQMR GPQMTR GPQNCR GPQPAR GPQPK GPQPKA GPQRCT GPQRV GPQRVE GPQRVO GPQRVR GPQRVX GPQSK GPQST GPQTR GPQVL GPQVR GPQWCT GPQWSU GPQWSX GPQWTO GPQXAF GPQXCR GPQXER GPQXIR GPQXLR GPQXMR GPQXTR GPRAST GPRDFB GPRQCH GPRQLC GPRQPK GPRQSK GPRQST GPRQVL GPRQXP GPSDAL GPSKMO GPSMCH GPSMLC GPSMPK GPSMSK GPSMST GPSMVL GPSMXP GPSRT GPSTMO GPTXR GPUPWA GPUPWS GPVCH GPVIP GPVLMO GPVMP2 GPVMP3 GPVMT2 GPVMT3 GPVOP GPVP GPWSX2 GPWSX3 GPXCR GPXER GPXIR GPXPLR GPXPMR GPXTXR GPXVCH GPXVR |
| 26 | GPCRWS GPOPWS |
| 35 | GPACFO GPAFDW GPARV GPARW GPASSW GPCAI GPCCV GPCIM2 GPCIM3 GPCML GPCPR GPCR GPCRC GPCSR GPDAFO GPDARW GPDCMM GPDCR GPDF GPDFI GPDIM GPDMR GPDRAV GPDRV GPDRW GPEAV GPER GPEV GPGTXC GPHR GPIR GPLNR GPLSR GPLTR GPMTR GPPAR GPPLR GPPMR GPQAAF GPQACF GPQADS GPQAEF GPQAFC GPQAFP GPQAGD GPQAIF GPQALF GPQALW GPQAMF GPQAMO GPQANF GPQANV GPQAPF GPQAPS GPQAR GPQAVF GPQCCH GPQCDF GPQCF GPQCID GPQCML GPQCPF GPQCPR GPQCQM GPQCR GPQCSF GPQCSR GPQCVO GPQDCF GPQDCR GPQDDV GPQDMR GPQDS GPQDV GPQEF GPQER GPQFBC GPQFP GPQGD GPQGDP GPQGFC GPQGSE GPQHD GPQHF GPQHLF GPQHMO GPQHR GPQICH GPQIDF GPQIF GPQIMC GPQIMF GPQIMI GPQIMV GPQIMW GPQIR GPQIVF GPQIW GPQLCF GPQLNR GPQLR GPQLSF GPQLSR GPQLTF GPQLTR GPQLW GPQMDS GPQMR GPQMTF GPQMTR GPQNSP GPQNV GPQPAF GPQPAR GPQPCR GPQPER GPQPIR GPQPLF GPQPLR GPQPMF GPQPMR GPQPPR GPQPTR GPQRCM GPQRV GPQRVO GPQSDF GPQTDF GPQTMO GPQTR GPQTXF GPQVF GPQVR GPQWSX GPQXAF GPQXCF GPQXCR GPQXER GPQXIR GPQXLR GPQXMR GPQXTR GPQXTX GPRAST GPTXR GPUPWA GPUPWS GPVOP GPXCR GPXER GPXIR GPXPLR GPXPMR GPXTXR |
| 37 | GPCUR GPCUS GPQCUF GPQID GPQPK GPQPKA |
| 38 | GPBKAC GPEPD GPICS GPIPKC GPIT GPPDMO GPPKAP GPPKSC GPQABK GPQAIS GPQAIT GPQALI GPQBK GPQBKS GPQCH GPQDBK GPQDCH GPQDIT GPQDLC GPQDPK GPQDSK GPQDST GPQDVL GPQICS GPQID GPQISF GPQIT GPQITS GPQLC GPQLI GPQNST GPQPKT GPQSK GPQSPD GPQST GPQVL GPVIP |
| 39 | GPBGTR GPCARR GPCPVP GPCRT GPCVW GPDRT GPDRVW GPSRT |
| 41 | GPQGDP |
| 43 | GPER GPIR GPPLR GPPMR GPQER GPQIR GPQLR GPQMR GPQPER GPQPIR GPQPLR GPQPMR GPQPTR GPQTR GPQXER GPQXIR GPQXLR GPQXMR GPQXTR GPTXR GPXER GPXIR GPXPLR GPXPMR GPXTXR |
| 44 | GPEVM2 GPEVM3 GPVMP2 GPVMP3 GPWSX2 GPWSX3 GPXVR |
| 47 | GPHR |
| 48 | GPPAR GPQPAR GPQPPR |
| 49 | GPCR GPXCR |
| 50 | GPDMR GPQDMR |
| 55 | GPEVM3 GPVMP3 GPXVR |
| 56 | GPVPLN GPVUP |
| 58 | GPCVMT GPDFCO |
| 59 | GPARV GPBGTR GPCIM2 GPCIM3 GPCPVP GPCVW GPDRV GPDRVW GPEV GPINLC GPINSK GPQCVR GPQCVX GPQIMV GPQRV GPQRVR GPQRVX GPVCH GPVIP GPVMP2 GPVMP3 GPVMT2 GPVMT3 GPVOP GPVP GPVWI GPXVCH GPXVR |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 60 | GPEI GPER GPII GPIR GPPLI GPPLR GPPMI GPPMR GPQER GPQIR GPQLR GPQMR GPQTR GPQXER GPQXIR GPQXLR GPQXMR GPQXTR GPQPER GPQPIR GPQPLR GPQPMR GPQPTR GPTXI GPTXR GPXER GPXIR GPXPLR GPXPMR GPXTXR |
| 61 | GPCRWS GPBDMF GPDMF GPHR GPMTR GPOPAR GPVMF GPVMF |
| 62 | GPLTR |
| 63 | GPELT GPER GPLNR GPLT GPLTR GPPLR GPQLTR GPXER GPXPLR |
| 64 | GPER GPLNR GPLTR GPPLR GPQLTR GPXER GPXPLR |
| 65 | GPCAC GPTCAC |
| 66 | GPCAC GPSAC GPTCAC |
| 67 | GPPHEC |
| 69 | GPMT GPMTR GPPMR GPQMTR GPXPMR |
| 70 | GPMTR GPPMR GPQMTR GPXPMR |
| 71 | GPHR GPLNR GPMTR |
| 75 | GPACFO GPDAFO GPDLFO GPLDFO GPQAFC GPQFAR GPQFCH GPQGFC GPQXAF GPTXFO GPTXR GPXTXR |
| 76 | GPCHLS |
| 77 | GPCHXP GPTXR GPXTXR |
| 78 | GPAH GPCHH GPCHL2 |
| 79 | GPAUP GPCHUP |
| 80 | GPCHUB |
| 81 | GPCHPM |
| 82 | GPAS |
| 83 | GPIR GPXIR |
| 84 | GPIR GPISI GPQHR GPXIR |
| 85 | GPPAR GPQPAR GPQPPR |
| 86 | GPSAC |
| 87 | GPPGC |
| 88 | GPFDMO |
| 89 | GPHR |
| 90 | GPQPAR GPQPPR |
| 91 | GPPAR GPPXL2 GPPXL3 GPQPAR GPQPPR |
| 92 | GPBICI GPBSCI GPCR GPDCR GPECI GPER GPHLCI GPICI GPIR GPLLCI GPLSR GPPAR GPPLCI GPPLR GPPMCI GPPMR GPSCI GPTXCI GPTXR GPVCH GPXCR GPXER GPXIR GPXPLR GPXPMR GPXTXR GPXVR |
| 93 | GPDCR GPER GPIR GPLSR GPPAR GPPLR GPPMR GPTXR GPVCH GPXER GPXIR GPXPLR GPXPMR GPXTXR GPXVR |
| 94 | GPEMO |
| 95 | GPPLSM |
| 96 | GPBICD GPBSCD GPCR GPDCR GPDMR GPECD GPHLCD GPICD GPIEC GPLLCD GPLSR GPPGD2 GPPGD3 GPPLCD GPPLD3 GPPMCD GPQM3 GPSCD GPTS3 GPTXCD GPXCR GPXER GPXPLR GPXPMR GPXTXR GPXVR |
| 97 | GPDCR GPLSR GPXER GPXIR GPXPLR GPXPMR GPXTXR GPXVR |
| 98 | GPCCM |
| 100 | GPCFA2 GPDPL2 GPDPL3 GPPL2 GPPL3 GPPLD3 GPPM2 GPPM3 GPTS3 |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 101 | GPSPH |
| 102 | GPPSC |
| 103 | GPPSC |
| 105 | GPGTXC |
| 106 | GPGTXC |
| 107 | GPCFA2 GPELA2 GPELA3 GPEL2 GPEL3 GPLG2 GPLG3 GPMG2 GPMG3 GPTX3 |
| 108 | GPAN2 GPAN3 GPANR2 GPANR3 GPPREC GPQFAR GPTX2 GPTX3 |
| 110 | GPBRMO GPRMO |
| 111 | GPBSPR GPSPR |
| 112 | GPBSPR GPSPR |
| 113 | GPBSPR GPSPR |
| 114 | GPBSPR GPSPR |
| 115 | GPBSPR GPBTCO GPDMR GPPGD2 GPPGD3 GPQM3 GPSPR GPTCO GPTS3 |
| 116 | GPLSS |
| 118 | GPFBC |
| 119 | GPRDFB |
| 120 | GPARSN GPARST GPRVSN GPRVST GPTST |
| 121 | GPEDMO |
| 122 | GPATS GPELS GPQCNA GPQCSN GPQEDA GPQEHA GPQEXS GPQISN GPQPAS GPQPDS GPQRST GPQVR GPQWSA GPRAS GPRDS GPRISN GPTE |
| 123 | GPCEXS GPCRET |
| 124 | GPCEXS |
| 125 | GPCEXS GPCPER GPCPST GPEXST |
| 126 | GPARV GPCIM2 GPCIM3 |
| 127 | GPCNRS GPTAST GPTST |
| 128 | GPARAS GPARSN GPARST GPRVAS GPRVSN GPRVST GPTAST GPTST |
| 129 | GPCSI GPCSIR GPCSRS |
| 130 | GPDELB GPDLEG GPEPLB GPEPLG |
| 132 | GPEPCD |
| 133 | GPDLEG |
| 134 | GPARSN GPARST GPDSAR GPDSNA GPRVSN GPRVST GPTST |
| 135 | GPQCNA GPQCSN |
| 136 | GPEPLG GPEPPG |
| 137 | GPCEDT |
| 138 | GPATS GPTE |
| 139 | GPATS GPNLER GPTE |
| 140 | GPCHMO GPEPD GPFLEV GPICS GPIDMO GPINCH GPINLC GPINPK GPINSK GPINST GPINVL GPIPKC GPIT GPLCMO GPPDMO GPPKAP GPPKF GPPKMO GPPKSC GPQAIS GPQAIT GPQCH GPQDCH GPQDIT GPQDLC GPQDPK GPQDSK GPQDST GPQDVL GPQICS GPQID GPQISF GPQIT GPQITS GPQLC GPQNST GPQPDC GPQPK GPQPKA GPQPKT GPQSK GPQSPD GPQST GPQVL GPRQCH GPRQLC GPRQPK GPRQSK GPRQST GPRQVL GPRQXP GPSKMO GPSMCH GPSMLC GPSMPK GPSMSK GPSMST GPSMVL GPSTMO GPVLMO |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 141 | GPICS GPINCH GPINLC GPINPK GPINSK GPINST GPINVL GPIPKC GPIT GPPKAP GPPKSC GPRQCH GPRQLC GPRQPK GPRQSK GPRQST GPRQVL GPRQXP |
| 142 | GPOPAR |
| 143 | GPNLER |
| 144 | GPINCH GPINLC GPINPK GPINSK GPINST GPINVL |
| 145 | GPINCH GPINLC GPINPK GPINSK GPINST GPINVL |
| 146 | GPCVD GPES GPINCH GPINLC GPINPK GPINSK GPINST GPINVL |
| 147 | GPAWEV GPCLWS GPDTR GPFLEV GPFWEV |
| 148 | GPQIQO |
| 150 | GPGTCH GPGTLC GPGTMS GPGTPK GPGTSK GPGTST GPGTVL GPGTXP GPGWIN |
| 151 | GPAWEV |
| 152 | GPINCH |
| 155 | GPVIP GPVOP GPVP |
| 156 | GPINPK |
| 158 | GPINPK |
| 160 | GPEPD GPPDMO GPQPDC |
| 161 | GPEPD |
| 162 | GPEPD |
| 163 | GPPDMO |
| 164 | GPRQXP GPSMXP |
| 166 | GPIPKC |
| 167 | GPPKSC |
| 168 | GPAWEV GPCHMO GPIDMO GPLCMO GPPKMO GPRQCH GPRQLC GPRQPK GPRQSK GPRQST GPRQVL GPRQXP GPSKMO GPSTMO GPVLMO |
| 169 | GPPDMO |
| 177 | GPCVD |
| 178 | GPCVD |
| 179 | GPCVD |
| 180 | GPCUS |
| 181 | GPCUR |
| 182 | GPCUR GPCUS |
| 183 | GPCUR |
| 197 | GPMSG GPSBMS GPSPMS |
| 198 | GPPGD2 GPPGD3 GPPG2 GPPG3 |
| 199 | GPPGD2 GPPGD3 GPPG2 GPPG3 |
| 201 | GPCNC |
| 202 | GPATR GPCRFD GPCRIB GPCRSS GPCRWS GPDNC GPEXAP GPINAP GPMSPW GPOPAR GPOPWS GPQATR GPQIBF GPQNCE GPQNCS GPQNS GPQOPW GPQPO GPQSH GPQWST GPQWTN GPSBMS GPSPMS GPSYNC |
| 203 | GPCNC |
| 204 | GPCNC |
| 205 | GPSHDF |
| 206 | GPSYNC |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 207 | GPSPMS |
| 208 | GPCNC GPOPPH |
| 209 | GPSHDF |
| 210 | GPEXAP |
| 211 | GPATR GPDTR GPPW GPQATR GPQNCR |
| 212 | GPATR |
| 213 | GPATR GPSPMS |
| 214 | GPPW |
| 215 | GPAFDW GPASSW GPDFI GPQAR GPQVR GPTHPO GPTRCT GPTWPO |
| 216 | GPQICH |
| 217 | GPCRFD GPCRIB GPCRSS GPCRWS GPEXAP GPOPAR GPOPWS |
| 218 | GPOPAR |
| 219 | GPATR GPOPAR |
| 220 | GPARAS GPARSN GPARST GPCLAR GPDASA GPDSAR GPDSNA GPQACA GPQCNA GPQNCR GPRAS GPRDS GPRISN GPRSTI GPRVAS GPRVSN GPRVST |
| 221 | GPATR GPCRSS |
| 222 | GPASSW GPDTR GPPW GPQACS GPQCSN GPQNCR GPSSS GPSSTH GPTAST GPTST |
| 223 | GPCRSS |
| 224 | GPARV |
| 225 | GPSSTH |
| 226 | GPASSW |
| 227 | GPQAR GPQVR |
| 231 | GPATR GPCRIB |
| 232 | GPDFI GPDTR GPFRCT GPPW GPQIBC GPQNCR GPRRCT GPTHPO GPTRCT GPTWPO GPWRCT |
| 233 | GPCRIB |
| 234 | GPCRIB |
| 235 | GPCRIB |
| 236 | GPCIM2 GPCIM3 GPDRI2 GPFRCT GPRDFB GPRRCT GPTHPO GPTRCT GPTWPO GPWRCT |
| 237 | GPRDFB GPRRCT GPWRCT |
| 238 | GPTWPO |
| 239 | GPTHPO |
| 240 | GPRDFB GPRRCT GPWRCT |
| 241 | GPATR GPCRFD |
| 242 | GPAFDW GPDLFO GPDTR GPLDFO GPPW GPQNCR |
| 243 | GPCRFD |
| 245 | GPLDFO |
| 250 | GPHID |
| 251 | GPXVR |
| 252 | GPAID |
| 253 | GPXVR |
| 254 | GPLSR GPLSS GPQLSR |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 255 | GPLSR GPQLSR |
| 256 | GPLSS |
| 257 | GPLMO |
| 258 | GPLSR |
| 259 | GPLSR |
| 260 | GPXVR |
| 261 | GPDCI GPDCR GPQDCR |
| 262 | GPDCR GPQDCR |
| 263 | GPDCR |
| 264 | GPDCR |
| 265 | GPCPI GPCPR GPQCPR GPXVR |
| 266 | GPCPR GPQCPR GPXVR |
| 267 | GPCPR |
| 268 | GPCPR |
| 269 | GPCPR |
| 272 | GPDCR GPQCVR GPQDCR GPQLCF GPQRVR GPQWDT GPQXER GPQXIR GPQXLR GPQXMR GPQXTR GPXER GPXIR GPXPLR GPXPMR GPXTXR GPXVR |
| 273 | GPQCVR GPQDCR GPQLCF GPQRVR GPQXER GPQXIR GPQXLR GPQXMR GPQXTR |
| 274 | GPHR GPLNR GPLTR GPMTR GPQHR GPQLTR GPQMTR GPQWSU |
| 275 | GPCPR GPDCR GPDMR GPLNR GPLTR GPMTR |
| 276 | GPHR GPMTR |
| 277 | GPHR GPLTR GPMTR |
| 278 | GPCSR GPQCSR GPTEX2 GPTEX3 |
| 279 | GPCSR GPQCSR |
| 280 | GPCSR |
| 281 | GPDCR |
| 282 | GPCRC GPDLC |
| 283 | GPCRC |
| 284 | GPDFI GPDLC GPQCCH GPQXCR GPXCR |
| 285 | GPCRC |
| 286 | GPCRC |
| 287 | GPCRC |
| 288 | GPCAI GPCIM2 GPCIM3 GPDFI GPDRI2 |
| 289 | GPXCR |
| 290 | GPCIM2 GPCIM3 GPDRI2 GPQICH GPQIMI |
| 291 | GPDFI |
| 292 | GPDFI |
| 293 | GPDFI |
| 294 | GPCIM2 GPCIM3 GPDRI2 |
| 295 | GPQPO |
| 296 | GPQIMC |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 297 | GPLNR |
| 299 | GPFLM |
| 301 | GPRAST |
| 302 | GPUPWS |
| 303 | GPDF |
| 304 | GPDF |
| 305 | GPTXPR GPTXR GPXTXR |
| 306 | GPAPT GPTXPT |
| 309 | GPAAL GPTXAL |
| 310 | GPIR GPIS GPXIR |
| 311 | GPEF GPER GPXER |
| 314 | GPASF GPQAAF |
| 315 | GPASF |
| 318 | GPCML GPDCM |
| 319 | GPMLX2 GPMLX3 |
| 320 | GPADCN GPRCN |
| 321 | GPHLF GPIVF GPPKF |
| 323 | GPARV GPBGTR GPCIM2 GPCIM3 GPCPVP GPCVW GPDRV GPDRVW GPEV GPINLC GPINSK GPQCVR GPQCVX GPQIMV GPQRV GPQRVR GPQRVX GPVCH GPVIP GPVMP2 GPVMP3 GPVMT2 GPVMT3 GPVOP GPVP GPXVCH GPXVR |
| 324 | GPINCH GPINLC GPINPK GPINSK GPINST GPINVL |
| 325 | GPINST |
| 326 | GPCHMO GPLCMO GPPKMO GPSKMO GPSTMO GPVLMO |
| 327 | GPCHMO GPIDMO GPLCMO GPPKMO GPSKMO GPSTMO GPVLMO |
| 328 | GPFLEV GPICS GPIDMO GPIT GPQAIS GPQAIT GPQALI GPQDIT GPQICS GPQID GPQISF GPQIT GPQITS GPQLI GPQNST GPQSPD |
| 329 | GPIDMO |
| 330 | GPEVM2 GPEVM3 GPVMP2 GPVMP3 GPWSX2 GPWSX3 GPXVR |
| 331 | GPCIM2 GPCIM3 GPEVM3 GPVMP3 GPXVR |
| 332 | GPMCI GPVCH GPXVCH GPXVR |
| 333 | GPVIP GPVOP GPVP |
| 334 | GPXVCH GPXVR |
| 336 | GPEVM3 GPVMP3 GPXVR |
| 340 | GPLG2 GPLG3 GPMG2 GPMG3 |
| 341 | GPCFA2 GPNBC2 GPNBC3 GPNBS GPRNBS GPRTNS GPTNBS |
| 342 | GPCFA2 GPNBC2 GPNBC3 GPNBS GPRNBS GPRTNS GPTNBS |
| 343 | GPCFA2 GPNBC2 GPNBC3 GPNBS GPRNBS GPRTNS GPTNBS |
| 345 | GPCFA2 GPNBC2 GPNBC3 GPNBS GPRNBS GPRTNS GPTNBS |
| 347 | GPCFA2 GPNBC2 GPNBC3 GPNBS GPRNBS GPRTNS GPTNBS |
| 348 | GPCFA2 GPDMR GPNBC2 GPNBC3 GPNBS GPRNBS GPRTNS GPTNBS |
| 349 | GPPGD3 GPPHE GPQM3 GPTS3 |
| 351 | GPNBC2 GPNBC3 GPNBS GPPGD2 GPPGD3 GPPLD3 GPQM3 GPRNBS GPRTNS GPSPH GPTNBS GPTS3 |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 352 | GPPGD2 GPPGD3 GPQM3 GPTS3 |
| 353 | GPCFA2 GPRTNS GPTNBS |
| 354 | GPCFA2 GPRTNS GPTNBS |
| 355 | GPCFA2 |
| 356 | GPPLD3 |
| 357 | GPQM3 |
| 361 | GPCFA2 GPRTNS GPTNBS |
| 362 | GPCFA2 GPNBC2 GPNBC3 GPNBS GPTNBS GPRNBS GPRTNS |
| 363 | GPPHE |
| 501 | GPES GPINCH GPINLC GPINSK GPINST GPINVL |
| 502 | GPINCH GPINLC GPINPK GPINSK GPINST GPINVL |
| 505 | GPGTMS GPGTPK GPGTSK GPGTST GPGTXP GPGWIN GPPREC GPQEMS GPQHLF GPQIVF GPRQPK GPRQSK GPRQST GPRQXP GPSMPK GPSMSK GPSMST GPSMXP |
| 506 | GPINPK GPINSK GPINST GPPREC |
| 507 | GPVCH GPXVCH GPXVR |
| 508 | GPVCH GPXVCH GPXVR |
| 509 | GPCVD GPDCMM GPES GPINAD GPINCH GPINLC GPINPK GPINSK GPINST GPINVL GPPGD2 GPPGD3 GPPLD3 GPPREC GPQCH GPQDCH GPQDLC GPQDMR GPQDPK GPQDSK GPQDST GPQDVL GPQLC GPQM3 GPQNCE GPQPK GPQSK GPQST GPQVL GPQWDT GPTS3 GPWDO |
| 511 | GPINVL |
| 512 | GPBISM GPDCMM GPDMR GPISM |
| 513 | GPINPK GPINSK GPINST |
| 514 | GPCRWS GPOPWS |
| 515 | GPINVL |
| 516 | GPDCMM |
| 517 | GPCR GPXCR |
| 518 | GPVCH GPVMP2 GPVMP3 GPVMT2 GPVMT3 GPXVCH GPXVR |
| 519 | GPGTCH GPGTLC GPGTMS GPGTPK GPGTSK GPGTST GPGTVL GPGTXP GPGWIN |
| 521 | GPELOG GPQEMS |
| 522 | GPVMT2 GPVMT3 GPXVR |
| 523 | GPASF |
| 524 | GPQEDA GPQEHA |
| 526 | GPQCVX GPQE GPQER GPQETS GPQIR GPQLR GPQMR GPQNCE GPQRVX GPQTR |
| 527 | GPCCV GPES GPQMDS GPRDFB |
| 528 | GPELS |
| 529 | GPELS |
| 530 | GPADCN GPRCN |
| 531 | GPHLF GPIVF GPPKF |
| 532 | GPAWEV |
| 533 | GPQCH GPQDCH GPQDLC GPQDMR GPQDPK GPQDSK GPQDST GPQDVL GPQHLF GPQIVF GPQLC GPQPAR GPQPDC GPQPK GPQPPR GPQSK GPQST GPQVL GPQWCT GPQWDT |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 534 | GPBGTR GPCPVP GPCRT GPDRT GPQCH GPQCR GPQCSR GPQDCR GPQER GPQIR GPQLC GPQLR GPQLSR GPQMR GPQNCE GPQPAR GPQPK GPQSK GPQST GPQTR GPQVL GPQXCR GPQXER GPQXIR GPQXLR GPQXMR GPQXTR GPSRT |
| 535 | GPQE GPQED GPQEHD |
| 536 | GPQNS GPQRCT GPQWTO |
| 537 | GPPAR GPPXL2 GPPXL3 |
| 538 | GPQAAF GPQABK GPQACA GPQACS GPQAEF GPQAES GPQAFC GPQAGD GPQAI GPQAIF GPQAIS GPQAIT GPQALF GPQALI GPQAMF GPQAMO GPQANF GPQAR GPQARF GPQATR GPQBK GPQCDF GPQCID GPQCMM GPQCNA GPQCNC GPQCQM GPQCSN GPQCUF GPQCVE GPQCVO GPQDCH GPQDIT GPQDLC GPQDPK GPQDSK GPQDST GPQDVL GPQE GPQEDA GPQEF GPQEHA GPQES GPQEXS GPQFO GPQGD GPQGFC GPQGSE GPQHMO GPQIBF GPQIDF GPQIF GPQIMF GPQIMI GPQIMV GPQIMW GPQISF GPQISN GPQIT GPQITS GPQIW GPQLI GPQLNR GPQLSF GPQOPW GPQPAS GPQPCR GPQPDS GPQPLF GPQPMF GPQPO GPQRCM GPQRST GPQRV GPQRVE GPQRVO GPQSDF GPQSID GPQSTI GPQTDF GPQTMO GPQVR GPQWCV GPQWSA GPQWST GPQWTN GPQXAF GPQXCF GPRAS GPRDS GPRISN GPRSTI |
| 539 | GPCCV GPQAAF GPQABK GPQACA GPQACS GPQAEF GPQAES GPQAFC GPQAGD GPQAI GPQAIF GPQAIS GPQAIT GPQALF GPQALI GPQAMF GPQAMO GPQANF GPQAR GPQARF GPQATR GPQBK GPQCDF GPQCID GPQCMM GPQCNA GPQCNC GPQCQM GPQCR GPQCUF GPQCVE GPQCVO GPQDCH GPQDIT GPQDLC GPQDPK GPQDSK GPQDST GPQDVL GPQE GPQED GPQEF GPQES GPQEXS GPQFO GPQGD GPQGFC GPQGSE GPQHMO GPQIBF GPQIDF GPQIF GPQIMF GPQIMI GPQIMV GPQIMW GPQISF GPQISN GPQIT GPQITS GPQIW GPQLI GPQLNR GPQLSF GPQOPW GPQPAS GPQPCR GPQPDC GPQPDS GPQPLF GPQPMF GPQPO GPQRCM GPQRST GPQRV GPQRVE GPQRVO GPQSDF GPQSID GPQSTI GPQTDF GPQTMO GPQVR GPQWCV GPQWSA GPQWST GPQWTN GPQXAF GPQXCF GPQXCR GPRAS GPRDS GPRISN GPRSTI |
| 540 | GPQED GPQEDA GPQEHA GPQEHD |
| 542 | GPACFO GPDAFO GPDLFO GPICS GPLDFO GPQAFC GPQFAR GPQFCH GPQGFC GPTXCS |
| 543 | GPQAAF GPQABK GPQACA GPQACS GPQAEF GPQAES GPQAFC GPQAGD GPQAI GPQAIF GPQAIS GPQAIT GPQALF GPQALI GPQAMF GPQAMO GPQANF GPQAR GPQARF GPQATR GPQBK GPQCDF GPQCID GPQCMM GPQCNA GPQCNC GPQCQM GPQCSN GPQCUF GPQCVE GPQCVO GPQDCH GPQDIT GPQDLC GPQDPK GPQDSK GPQDST GPQDVL GPQE GPQEF GPQES GPQEXS GPQFO GPQGD GPQGFC GPQGSE GPQHMO GPQIBF GPQIDF GPQIF GPQIMF GPQIMI GPQIMV GPQIMW GPQISF GPQISN GPQIT GPQITS GPQIW GPQLI GPQLNR GPQLSF GPQOPW GPQPAS GPQPCR GPQPDS GPQPLF GPQPMF GPQPO GPQRCM GPQRST GPQRV GPQRVE GPQRVO GPQSDF GPQSID GPQSTI GPQTDF GPQTMO GPQVR GPQWCV GPQWSA GPQWST GPQWTN GPQXAF GPQXCF GPRAS GPRDS GPRISN GPRSTI |
| 544 | GPQCR GPQXCR |
| 547 | GPVCH GPXVCH GPXVR |
| 548 | GPQAAF GPQAMO GPQANF GPQART GPQBK GPQCDF GPQCF GPQCPF GPQCQM GPQCSF GPQDBK GPQDCF GPQDCH GPQDDV GPQDIT GPQDLC GPQDPK GPQDS GPQDSK GPQDST GPQDVL GPQEF GPQES GPQFBC GPQFP GPQGD GPQGDP GPQGSE GPQHD GPQHF GPQHMO GPQIDF GPQIF GPQIMF GPQISF GPQIT GPQLCF GPQLI GPQLNR GPQLSF GPQLTF GPQLW GPQMTF GPQNCN GPQNSP GPQNST GPQNV GPQPAF GPQPCR GPQPCS GPQPDC GPQPER GPQPIR GPQPKT GPQPLF GPQPLR GPQPMF GPQPMR GPQPPR GPQPTR GPQRCM GPQSDF GPQSPD GPQTDF GPQTMO GPQTXF GPQVF GPQWC GPQWCV GPQWD GPQWDT GPQXCF |
| 549 | GPPXL2 GPPXL3 |
| 550 | GPICS |
| 551 | GPQCR GPQXCR |
| 552 | GPQPAS GPQPDS GPRAS GPRDS |
| 553 | GPDAFO |
| 554 | GPPKAP |
| 555 | GPDPL2 GPDPL3 |
| 557 | GPCFA2 GPDPL2 GPDPL3 GPINSK GPNBC2 GPNBC3 GPNBS GPPGD2 GPPGD3 GPPG2 GPPG3 GPPLD3 GPPL2 GPPL3 GPPM2 GPPM3 GPPREC GPQM3 GPRNBS GPRTNS GPSPH GPTNBS GPTS3 |
| 558 | GPQPAS GPQPDS GPRAS GPRDS |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 559 | GPACFO |
| 560 | GPQFAR GPQFCH |
| 561 | GPQGFC |
| 562 | GPQAFC GPQXAF |
| 563 | GPACFO GPQGFC |
| 564 | GPQFAR |
| 565 | GPIT |
| 566 | GPEPPG GPEPPK |
| 567 | GPBKAC GPIT |
| 568 | GPBKAC GPIT |
| 569 | GPIT GPQAIT GPQIT |
| 570 | GPIT GPQDIT |
| 571 | GPQAFC GPQCVR GPQNCS GPQRVR GPQWSU GPQXAF GPQXCR |
| 572 | GPBKAC GPQABK GPQBK |
| 574 | GPIT |
| 575 | GPIT |
| 576 | GPIT |
| 577 | GPINSK GPINST GPQARF GPQEDA GPQPAS GPQPDS GPRAS GPRDS |
| 578 | GPINSK GPINST |
| 579 | GPINSK GPINST |
| 580 | GPINSK GPINST |
| 581 | GPCRWS GPOPWS |
| 582 | GPCRA2 GPCR2 GPSPH |
| 583 | GPCRWS GPOPWS |
| 584 | GPPLET |
| 585 | GPCRWS GPOPWS |
| 586 | GPCRWS GPOPWS |
| 587 | GPCRWS GPOPWS |
| 588 | GPCRWS GPOPWS |
| 591 | GPXVCH |
| 592 | GPXVCH |
| 594 | GPARAS GPARSN GPARST GPRVAS GPRVSN GPRVST GPTAST GPTST |
| 595 | GPIT |
| 596 | GPCRWS GPOPWS |
| 597 | GPCRWS GPOPWS |
| 598 | GPCRWS GPOPWS |
| 599 | GPCRWS GPOPWS |
| 600 | GPCRWS GPOPWS |
| 601 | GPCRWS GPOPWS |
| 602 | GPCRWS GPOPWS |
| 608 | GPEVM3 |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 610 | GPEVM3 |
| 612 | GPPSTS |
| 613 | GPPTS |
| 614 | GPARAS GPARSN GPARST GPRVAS GPRVSN GPRVST GPTAST GPTST |
| 624 | GPCCV |
| 625 | GPCCV |
| 627 | GPMCV2 GPMCV3 |
| 628 | GPMCV2 GPMCV3 |
| 629 | GPBBLF GPBLF |
| 630 | GPBDMI GPDMI GPDMR GPQDMR |
| 631 | GPBDFM GPDFM |
| 632 | GPBDFM GPDFM |
| 633 | GPBDM2 GPDM2 |
| 634 | GPDMR |
| 635 | GPDMR |
| 636 | GPPGD2 GPPGD3 GPPLD3 GPTS3 GPQM3 |
| 637 | GPDMR |
| 638 | GPDMR |
| 639 | GPXVR |
| 647 | GPACFO |
| 648 | GPCRWS GPOPWS |
| 649 | GPCRWS GPOPWS |
| 650 | GPCRWS GPOPWS |
| 1113 | GPARAS GPARSN GPARST GPDASA GPDSAR GPDSNA GPOPAR |
| 1132 | GPEXAP |
| 1133 | GPEXAP |
| 1205 | GPOPAR |
| 1206 | GPOPAR |
| 1301 | GPEXAP GPINAP |
| 1302 | GPTMAP |
| 1303 | GPEXAP GPINAP |
| 1304 | GPEXAP GPINAP |
| 1305 | GPEXAP GPINAP |
| 1307 | GPEXAP GPINAP |
| 1308 | GPEXAP GPINAP |
| 1309 | GPEXAP GPINAP |
| 1310 | GPEXAP GPINAP |
| 1311 | GPEXAP GPINAP |
| 1316 | GPEXAP GPINAP |
| 1317 | GPEXAP |
| 2050 | GPES GPWDO |

| ERROR NUMBERS | ASSOCIATED SUBROUTINES |
|---|---|
| 2051 | GPES GPWDO |
| 2052 | GPES GPWDO |

# Appendix D. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Dept. LRAS/Bldg. 003
11400 Burnet Road
Austin, TX 78758-3498
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:
(c) (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. (c) Copyright IBM Corp. _enter the year or years_. All rights reserved.

## Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AIX®
AIXwindows
GDDM®
IBM®
PS/2

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be the trademarks or service marks of others.

Printed in U.S.A.

SC23-6624-00