GDDM-GKS
Programming Guide and Reference

SC33-0334-00

GDDM-GKS
Programming Guide and Reference

SC33-0334-00

First Edition (March 1987)

This edition applies to Release 1 of the IBM GDDM-GKS licensed program, 5668-802, and to any subsequent releases and modifications unless otherwise indicated in new editions or Technical Newsletters. Changes are made periodically to the information herein; before you use this publication in connection with the operation of IBM systems, consult the latest *IBM System/370, 30xx, and 4300 Processors Bibliography*, GC20-0001, for the editions that are applicable and current.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's program may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the addresses given below. Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, address comments to either:

International Business Machines Corporation, Department 6R1H, 180 Kost Road, Mechanicsburg, PA 17055, U.S.A.

or

IBM United Kingdom Laboratories, Information Development, Mail Point 95, Hursley Park, Winchester, Hampshire, England SO21 2JN.

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

No part of this manual may be reproduced in any form or by any means, including storing in a data processing machine, without permission in writing from IBM. Permission is hereby granted to copy and store the sample programs into a data processing system and to use the stored programs for study and instruction only. No permission is granted to use the sample programs for any other purpose.

© **Copyright International Business Machines Corporation 1984, 1987. All rights reserved.**
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

© Copyright IBM Corp. 1984, 1987

Preface

What this book is about

This manual provides information about writing and running application programs that use GDDM-GKS, the Graphical Data Display Manager Graphical Kernel System licensed program.

Who this book is for

The material is intended primarily for application programmers, although some of the information is also applicable to system programmers.

What you need to know

This book includes introductory information to let you start using GDDM-GKS to perform simple graphics programming tasks. To advance to more complicated GKS programming, you are recommended to read the various application programming guides that are available, in which techniques for GKS programming are described. Some specific techniques are described in this book. (You should be aware that the *GDDM Base Application Programming Guide* does **not** give guidance that is specific to GDDM-GKS.)

You may also need to use the *GDDM Base Application Programming Reference* book for general guidance and support information.

How to use this book

The first four chapters can be read sequentially, to give an overview of GDDM-GKS. The remainder of the book is reference material, to be read as you need it.

Latest GDDM information

For up-to-date information on GDDM products, check our Home Page on the Internet at the following URL:

`http://www.hursley.ibm.com/gddm/`

You might also like to look at the IBM Software Home Page at:

`http://www.software.ibm.com`

© Copyright IBM Corp. 1984, 1987

GDDM publications

GDDM Base	*GDDM Base Application Programming Guide*, SC33-0867 *GDDM Base Application Programming Reference*, SC33-0868 *GDDM Diagnosis*, SC33-0870 *GDDM General Information*, GC33-0866 *GDDM/MVS Program Directory*, GC33-1801 *GDDM/VM Program Directory*, GC33-1802 *GDDM/VSE Program Directory*, GC33-1803 *GDDM Messages*, SC33-0869 *GDDM Series Licensed Program Specifications*, GC33-0876 *GDDM System Customization and Administration*, SC33-0871 *GDDM User's Guide*, SC33-0875 *GDDM Using the Image Symbol Editor*, SC33-0920
GDDM-GKS	*GDDM-GKS Programming Guide and Reference*, SC33-0334
GDDM-IMD	*GDDM Interactive Map Definition*, SC33-0338
GDDM-IVU	*GDDM Image View Utility*, SC33-0479
GDDM-PGF	*GDDM-PGF Application Programming Guide*, SC33-0913 *GDDM-PGF Programming Reference*, SC33-0333 *GDDM-PGF Interactive Chart Utility*, SC33-0328 *GDDM-PGF Vector Symbol Editor*, SC33-0330 *GDDM-PGF OPS User's Guide*, SC33-1776

GDDM/MVS is an element of OS/390. GDDM-REXX/MVS and GDDM-PGF are optional features of OS/390. For a complete list of the publications associated with OS/390, see the *OS/390 Information Roadmap*, GC28-1727.

Books from related libraries

Information on the IBM 3279 Color Display Station can be found in:

IBM 3279 Color Display Station: Operator's Guide, GA33-3057.

Information on the IBM 3270-PC/G, /GX, AT/G, and AT/GX can be found in:

Introducing the IBM 3270 Personal Computer/G and /GX Ranges of Work Stations, GA33-3157.

Valuable supplementary information about the use of color and programmed symbols on the IBM 3279 Color Display Station and the IBM 3287 Printer is contained in:

IBM 3270 Information Display System: Color and Programmed Symbols, GA33-3056.

Further information about the terminals of the IBM 3270 Information Display System is given in:

IBM 3270 Information Display System: 3274 Control Unit Description and Programmer's Guide, GA23-0061.

The following manuals may be useful:

VS APL for CMS: Terminal User's Guide, SH20-9067

VS APL for OS/TSO: Terminal User's Guide, SH20-9180

APL2 Programming: System Services Reference, SH20-9218.

Book structure

Chapter 1. Introduction to GKS

Graphical Kernel System (GKS), an International Standard (ISO 7942), of the International Organization for Standardization (ISO), is a basic graphics system supporting the definition and representation of two-dimensional pictures on a wide variety of graphics devices.

GKS allows you to code a wide range of graphics applications, from simple passive output to complex interactive exchanges. Once coded, applications can be made to run on many different graphics devices without the need to modify the program structure.

This book describes the GDDM-GKS implementation of GKS.

If you have not used GKS before, try running the ROOM program provided with GDDM-GKS. (Instructions are given below; no knowledge of GKS is required.) In the next chapter, parts of this program are used to introduce some of the GKS functions.

The ROOM program

Running the ROOM program helps you to verify that the system is correctly installed. The program also demonstrates some of the basic features of GKS. The program uses a graphics monitor as an output device. ROOM is available only if you have FORTRAN on your system.

Using ROOM under VM/CMS

To use the ROOM program if you run under VM/CMS, you must first compile and load the program. The program has been written so that it can be compiled and run using one of the FORTRAN compilers:

VS FORTRAN
FORTRAN IV G
FORTRAN IV H

If you are using the VS FORTRAN 4.1 compiler:

1. Compile the program:

```
FORTVS ADMJROOM
```

© Copyright IBM Corp. 1984, 1987

2. Make the GDDM-GKS and FORTRAN libraries available:

```
SET LDRTBLS 6
GLOBAL LOADLIB VFLODLIB
GLOBAL TXTLIB VFORTLIB CMSLIB ADMNLIB ADMKLIB ADMGLIB
```

3. Load and run the program:

```
LOAD ADMJROOM ADMJB77
START *
```

If you are using a FORTRAN IV compiler, you must use the FORTRAN IV compilation command and programming libraries. Also, when loading the program, you must use the GDDM-GKS FORTRAN IV binding routine, ADMJBIV, and not the VS FORTRAN binding routine, ADMJB77, in step 3 above.

Go to the section Running the ROOM program, below.

Using ROOM under TSO

If your system programmer has given you instructions for running the ROOM program, follow those instructions. Otherwise, your system programmer has probably set up a CLIST to allow you to run ROOM. The name of the CLIST is "GKSROOM." Try entering:

```
GKSROOM
```

and see if the ROOM program starts. If it does not start, see your system programmer for assistance.

Running the ROOM program

To run the program:

1. The initial screen appears with an empty room layout. Two pieces of furniture are on the right-hand side of the screen (the storage area), and a selection menu is on the lower part of the screen.
2. Press PF1 on your keyboard. A graphics cursor (+) appears in the lower right side of the room.
3. Using the cursor-control arrow keys (or a mouse or puck), move the cursor over one of the pieces of furniture in the storage area. Press Enter (or a mouse or puck button) to pick the item.
4. Press PF2. The graphics cursor reappears at the center of the room.
5. Move the cursor to the position in the room where you intend to place the selected item, and press Enter. The furniture appears at that location with the same orientation as it had in the storage area.
6. You can reposition the item by pressing PF2 again and repeating step 5.

7. Press PF3 to rotate the piece of furniture. The prompt “angle:” appears below the selection menu. Enter the number of degrees you need to rotate the piece of furniture counterclockwise from its *original* position.
8. You can rotate a piece of furniture again by pressing PF3 and repeating step 7. The rotation is always relative to the original position of the furniture in the storage area.
9. Repeat steps 2 through 7 to include as many pieces as the room will hold, and arrange them as you want.
10. Press PF4 to remove a piece of furniture from the room. When the cursor appears, move it over the item you want to delete. The item disappears when you press Enter.
11. Press PF5 to end the program at any time. The screen is refreshed with the final layout, and then the program ends.

Chapter 2. Using GKS

This chapter contains:

- An overview of the concepts of the Graphical Kernel System (GKS)
- An introduction to some of the GKS functions
- A step-by-step analysis of the ROOM program.

Notes:

1. GKS is a standard having several different levels, each higher level providing more functions. In this chapter, the term GKS is used when generic GKS functions are discussed, but the term GDDM-GKS is used when the functions provided specifically by this implementation of GKS are described.
2. In this chapter, programming examples are given in a generic, **pseudocode** format. Function calls use the GKS function names, rather than the FORTRAN binding names, to make the purpose of the example clearer. If you wish to code one of the examples, you will need to substitute calls to the binding names, given in Chapter 5, “GDDM-GKS functions” on page 63.

Concepts of GKS

The concepts of GKS described in this chapter are:

- Workstations
- Graphical output
- Coordinate systems
- Segments
- Graphical input
- Metafiles
- Operating states
- Data structures.

Workstations

GKS **workstations** are abstract devices providing logical interfaces to physical devices. GKS identifies its workstations by means of workstation identifiers that you assign, and by workstation types.

In this book, the term workstation is also used to refer generally to the physical devices used with GDDM-GKS. GKS supports three categories of physical-device workstations:

Output Output only
Input Input only
Outin Output and input.

Examples of physical-device workstations are graphics terminals, plotters, and printers.

© Copyright IBM Corp. 1984, 1987

This implementation of GKS also includes segment capabilities, providing a fourth category of workstation for segment storage:

WISS Workstation independent segment storage.

The WISS is a virtual device that allows pictures to be stored independently of any particular physical workstation.

There are also two GKS metafile workstation categories:

MI Metafile input
MO Metafile output.

(GKS metafiles are described in a later section.)

GDDM-GKS allows a maximum of five workstations to be open at any one time.

Graphical output

GKS output to graphics devices is performed using **output primitives**. These are basic graphics elements used to construct a graphics image. For example, a **polyline** is an output primitive that draws a line.

Output primitives have associated **attributes** that determine particular properties of the graphics image produced. For example, *linetype* is a polyline attribute that selects the sort of line that is drawn.

Coordinate systems

There are three types of coordinate system in GKS. The GKS **transformation** functions perform mappings between the coordinate systems.

A **world coordinate** (**WC**) system is a device-independent Cartesian coordinate system. You build your graphics image, and input its coordinate points to GKS, in WC. Several WC systems can be used in a GKS program.

Device coordinate (**DC**) systems are defined in terms of individual display devices and their built-in methods of measuring display space.

The **normalized device coordinate** (**NDC**) system is a standardized, **virtual** plane, onto which all world coordinate systems are mapped.

Segments

You can group output primitives and attributes together in **segments**. For example, the four lines forming a square could be drawn using four polylines. By grouping these in a segment, you can then treat the four lines as one unit. For example, you can then move the square about, make it invisible, or highlight it.

Graphical input

GKS input from an operator is obtained through logical input devices, each identified by a workstation identifier, an **input class**, and a device number. The input class identifies the type of data to be received; for example, a *valuator* input yields a real number.

GKS logical input devices can be operated in several **operating modes**; GDDM-GKS implements only one of these: *request* mode.

GKS metafiles

GKS provides a way of sending and retrieving data (pictures or non-graphical) for storage on external disk-storage devices. **GKS metafiles (GKSMs)** are logical workstations. The MO (metafile output) workstation writes information to a sequential metafile, and the MI (metafile input) workstation retrieves it.

GKS operating states

GKS exists in one of the following five operating states:

GKCL GKS closed

GKOP GKS open

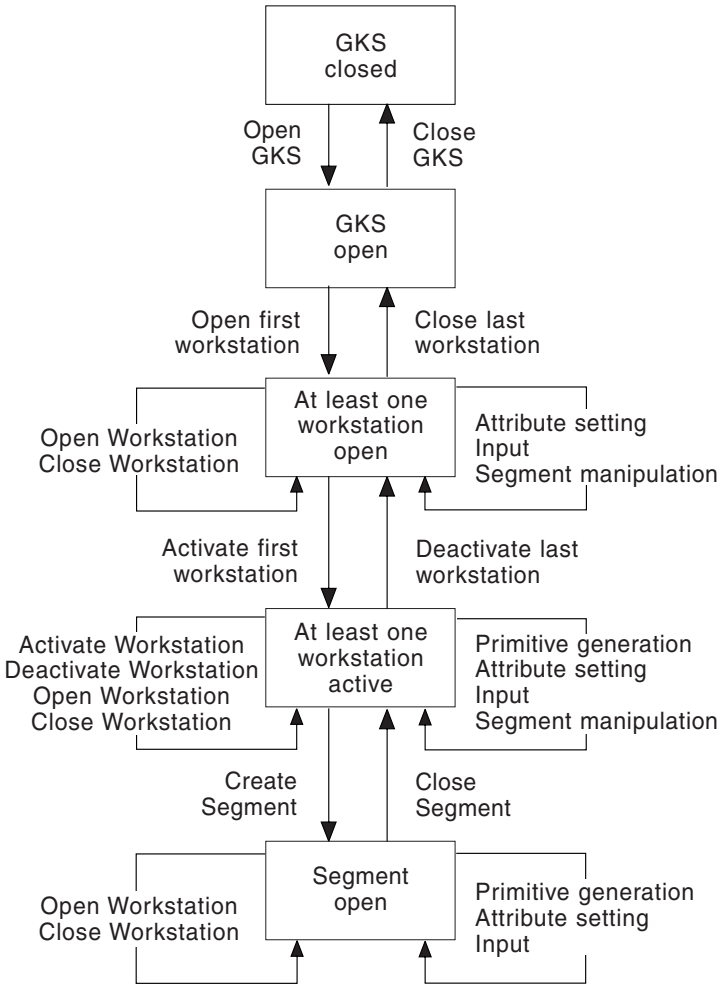
WSOP At least one workstation open

WSAC At least one workstation active

SGOP Segment open.

Each GKS function requires the system to be in a specific operating state. If you make a call in the wrong operating state, you will receive an error code in the error logging file indicating that GKS is not in the correct state.

This chart illustrates the five GKS operating states and the types of functions that are allowed in each state.



GKS data structures

The overall state of GKS is defined by a set of state variables having specific values. These state variables allow a complete description of the effects of GKS functions. The total set of GKS state variables includes these subsets:

- Operating state
- GKS state list
- GKS description table
- Workstation state list for every open workstation
- Workstation description table for every workstation type
- Segment state list
- GKS error state list.

These state lists appear in Appendix A, “GKS data structures” on page 359.

The state subsets are initialized with default values. Some default values for workstation state lists are taken from the workstation description tables. The variables of the state subsets are modified and inquired by calls to GKS functions.

GKS functions

GKS provides a set of **functions** for generating graphics images. In GDDM-GKS, each function is implemented as a subroutine; you include calls to the subroutines in your program. GDDM-GKS can support applications written in several programming languages, including FORTRAN, PL/I, and COBOL. The external references generated when you compile your program are resolved when you link-edit or load your application using the GDDM-GKS libraries.

The calling conventions and parameter sequences for each programming language are described in Chapter 4, “Using GDDM-GKS” on page 43.

GKS function types

The GKS functions are separated into ten categories according to the type of task each one performs. A “List of functions by function type” follows the “Alphabetical list of functions” in Chapter 5, “GDDM-GKS functions” on page 63.

The GKS function types are:

Control	Affect the state of the system or of individual workstations
Output	Display basic graphic images: **primitives**
Attribute	Modify the appearance of graphic primitives
Transformation	Translate primitives between coordinate planes
Segment	Create, delete, and manipulate functional groups of related primitives: **segments**
Input	Accept input data, allowing interactive applications to be used
Metafile	Store or retrieve graphical data from disk storage

Utility	Pack or unpack data records and compose segment transformation matrixes
Inquiry	Return data to the application, including information about workstations, logical input devices, and current attribute settings
Error handling	Assist the application program in logging and responding to errors.

Control functions

GKS **control functions** initialize and terminate GKS, and provide you with workstation control functions.

In your application, you must call Open GKS (GOPKS) before any other GKS function. It sets the system to the operating state GKOP. The GKS state list is initialized with default values. The GKS description table and the workstation description tables are made available for inquiry functions. You specify a number to identify the file to be used by GDDM-GKS for error logging.

Open workstation (GOPWK) sets the system to the operating state WSOP. A workstation state list is initialized with default values for each workstation you open. You assign the workstation an identifying number for use in other functions. You specify a number that identifies the workstation type (physical devices, WISS, metafile and so on); to open the GDDM-GKS user console, you use workstation type 1. The connection identifier identifies a particular device.

Activate workstation (GACWK) sets the system to the operating state WSAC. Only *output*, *outin*, *WISS*, and *MO* workstations can be activated. This function causes subsequent output primitives to go to the device.

For more information on the state lists and description tables, see Appendix A, “GKS data structures” on page 359.

Output and attribute functions

Output **primitives** are the basic building blocks of a graphics image. GKS primitives include:

- Polyline
- Polymarker
- Fill area
- Text
- Cell array
- Generalized drawing primitive.

Each primitive has a number of **attributes** that you set to control its appearance.

Specifying attributes

Attributes affecting the **aspect** of output primitives are of two types:

- **Geometric** - affecting the shape or size of the primitive
- **Non-geometric** - merely affecting the appearance.

Geometric attributes (for example, *character height* for Text) are workstation-independent, and each attribute is set separately for each primitive associated with it.

The non-geometric aspects of primitives are workstation-dependent, and can be specified in either of two ways:

- By an individual attribute (for example, *linetype* for polylines)
- Via a **bundle** consisting of tables of attribute values.

A GKS function allows you to set an **aspect source flag (ASF)** for each primitive. The function (Set aspect source flags (GSASF)) has two valid input values, *individual* or *bundled*, to select the way in which the attributes for each primitive are to be specified. The GDDM-GKS default is individual.

To use a bundle, each primitive has an associated function to let you specify an attribute called a primitive **index** (for example, Set polyline index (GSPLI)). This is an index into a **bundle table**, containing all the non-geometric aspects for a particular primitive. Each primitive (except for generalized drawing primitive, and cell array) has a bundle table for each workstation. The set of entries for each primitive is called a **representation**. Each primitive has an associated function (for example, Set text representation (GSTXR)) to allow you to build the bundle tables.

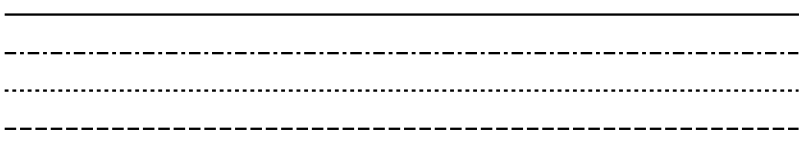
Polyline

The **polyline** is the fundamental line-drawing primitive. When you enter a series of points to the Polyline (GPL) output function, GKS draws straight lines between the points in the sequence in which you entered them. You can set the following **polyline attributes**:

- Polyline type (Set linetype (GSLN))
- Polyline color (Set polyline color index (GSPLCI))
- Polyline width scale factor (Set linewidth scale factor (GSLWSC)).

You choose the polyline color from among the predefined colors your display device supports, or from color representations you define using the Set color representation (GSCR) function. You can find out the number of line types and line width scale factors your device supports by calling Inquire polyline facilities (GQPLF). If the number of line widths returned is 0, the device supports a continuous range of line widths.

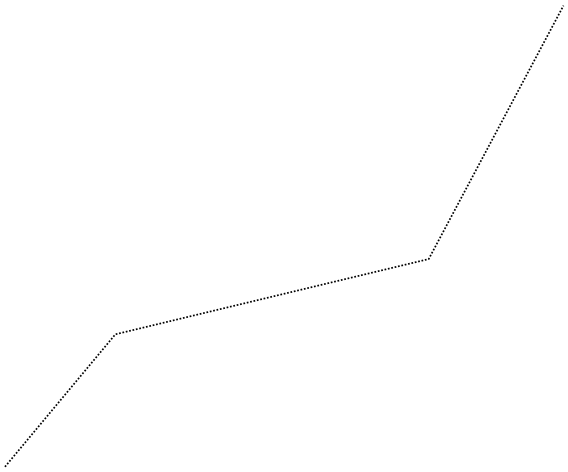
GKS supports four line types:



This example draws a polyline by putting the coordinate information in arrays and calling the Polyline (GPL) output function to connect the points.

```
X array = (10.0,25.0,65.0,85.0)
Y array = (30.0,45.0,50.0,80.0)
SET LINETYPE (3)
POLYLINE (4, X array, Y array)
```

This sequence produces the following output:



Polymarker

Polymarkers are symbols such as asterisks (*) or crosses (+) that you can use to mark any point on the display surface. You can set the following **polymarker attributes**:

- Polymarker type (Set marker type (GSMK))
- Polymarker color index (Set polymarker color index (GSPMCI))
- Polymarker size scale factor (Set marker size scale factor (GSMKSC)).

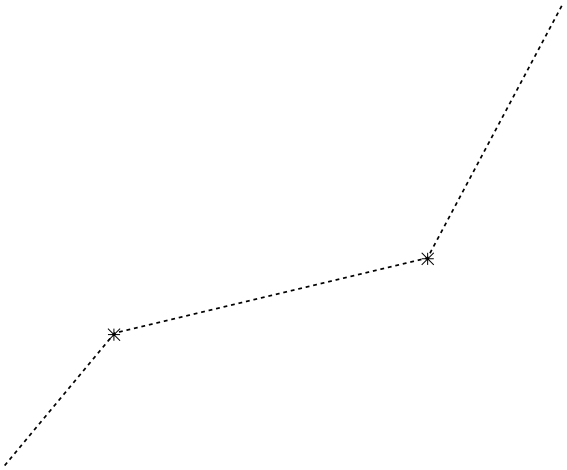
You choose the polymarker color from among the predefined colors your display device supports or from color representations you define in the Set color representation (GSCR) function. You can find out the number of marker types and marker sizes your device supports by calling Inquire polymarker facilities (GQPMF). If the number of marker sizes returned is 0, the device supports a continuous range of marker sizes.

GKS supports five polymarker types:

This example marks the points (65.0,50.0) and (25.0,45.0) on the previous example by putting the points in arrays and calling the Polymarker (GPM) output function to mark the points.

```
X array = (65.0,25.0)
Y array = (50.0,45.0)
SET MARKER TYPE (3)
POLYMARKER (2, X array, Y array)
```

This sequence produces the following output:

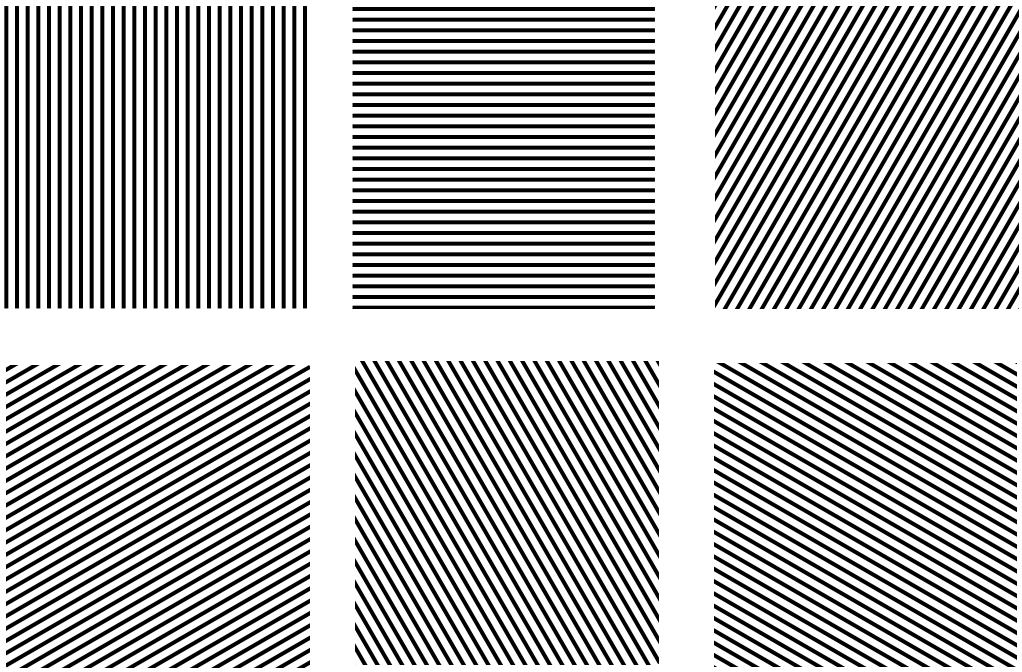


Fill area

A **fill area** is a polygon that can be hollow or painted with a color, pattern, or hatch style. You define a fill area by specifying the vertices of the polygon that encloses it, and calling the Fill area (GFA) output function. You can set the following **fill area attributes**:

- Fill area color index (Set fill area color index (GSFACI))
- Fill area interior style (Set fill area interior style (GSFAIS))
- Fill area style index (Set fill area style index (GSFASI))

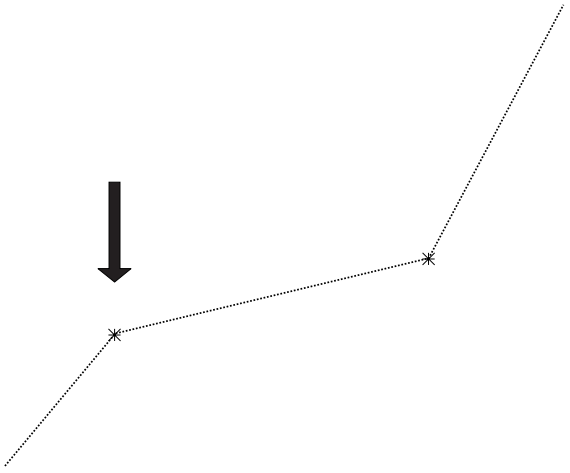
You choose the fill area color from among the predefined colors your display device supports, or from color representations you define in the Set color representation (GSCR) function. When you choose an **interior style**, you decide whether the area is to be outlined (hollow style), filled solidly, or painted with a pattern or hatch style. When you choose a **style index**, you are choosing among the available device-dependent patterns or from the range of six hatch styles provided by GDDM-GKS:



The following example draws an arrow, fills it with solid color, and adds it to the previous example. The points of the three vertices are put into arrays, and the Fill area (GFA) function is called.

```
X array = (26.0,24.0,24.0,22.0,25.0,28.0,26.0,26.0)
Y array = (63.0,63.0,53.0,53.0,50.0,53.0,53.0,63.0)
SET FILL AREA INTERIOR STYLE (1)
FILL AREA (8, X array, Y array)
```

This sequence produces the following output:



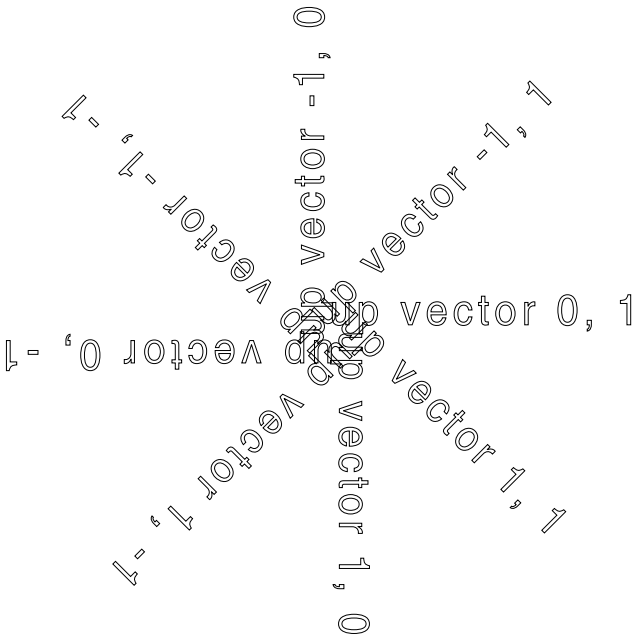
Text

A **text** primitive is a string of characters. You provide the characters to be used, and the starting point in world coordinates. You can set the following **text attributes**:

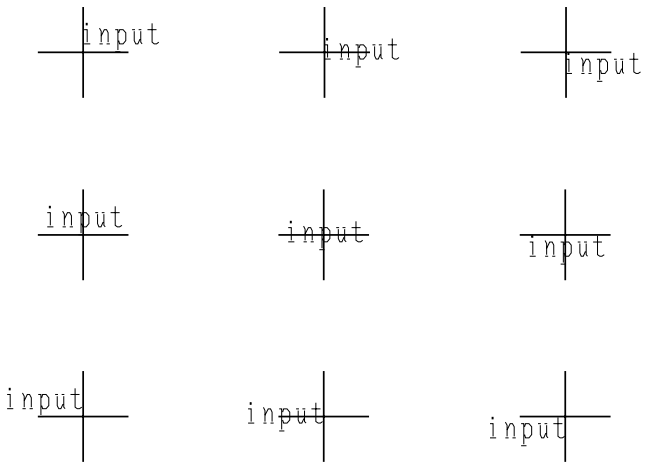
- Text color index (Set text color index (GSTXCI))
- Text font and precision (Set text font and precision (GSTXFP))
- Character height (Set character height (GSCHH))
- Character up vector (Set character up vector (GSCHUP))
- Text alignment (Set text alignment (GSTXAL))
- Text path (Set text path (GSTXP)).

You choose the text color from among the predefined colors your display device supports, or from color representations you define in the Set color representation (GSCR) function. When you choose a **text font**, you choose among device-dependent alternatives. You can find the number of available text fonts, and the minimum and maximum **character heights** supported by your device, by calling the Inquire text facilities (GQTXF) function.

The character up vector is a line from the origin (0,0) to a point in world coordinates, and it establishes an up direction for text characters. You set this attribute with the function Set character up vector (GSCHUP). The following illustration shows the text strings displayed with eight different character up vectors.



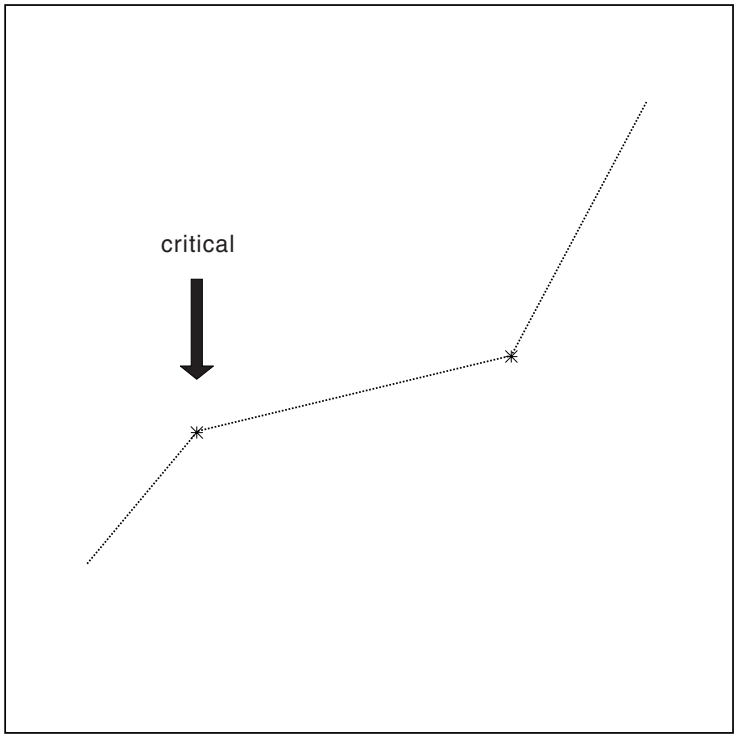
When you set **text alignment**, you specify both horizontal and vertical alignment values. You set this attribute with the function Set text alignment (GSTXAL). Here, the string “input” is displayed at the text alignment positions supported by GDDM-GKS.



This example displays the string “critical” centered at the top of the arrow from the previous example.

```
SET TEXT ALIGNMENT (2,1)
TEXT (25.0, 68.0, 'critical')
```

This sequence produces the following output:



Setting the **text path** specifies the position (right, left, up, or down) of each character in the string relative to the preceding character.

Cell array

A **cell array** is an output primitive that can produce multicolored images. The image consists of a grid of rectangular cells, each cell having its own defined color. The cells need not map one-for-one with pixels on the output device. The output grid is defined by a set of indexes into a color table. The Cell array (GCA) output function generates the output image.

A cell array can be subjected to any transformation. This may yield an array in which the cells are rhomboids (diamond-shaped parallelograms).

Generalized Drawing Primitive

Generalized drawing primitives (GDPs) allow special geometric workstation capabilities such as curve drawing to be provided. GDDM-GKS has no GDPs.

Transformation functions

GKS transformation functions are provided to allow mapping between the three coordinate systems, *world coordinates* (WC), *device coordinates* (DC), and *normalized device coordinates* (NDC).

There are three types of transformations; each involves a different combination of coordinate planes. **Normalization transformations** map WC to NDC. **Workstation transformations** map NDC to DC. **Segment transformations** map geometric primitives from NDC to NDC. (For more information about segment transformations, see “Segment functions” later in this chapter.)

Normalization transformations

When you set a normalization transformation, you are mapping a range of WC to a range of NDC. The **world window**, set with the Set window (GSWN) function, encloses the range of valid WC points containing the graphics picture. The **viewport**, set with the Set viewport (GSVP) function, specifies a range of NDC points to which your world window will be mapped.

The part of NDC space in which the viewport must be located, and which can be viewed at a workstation, is the closed range (0.0,1.0)x(0.0,1.0).

This example sets the range of WC to (0.0,100.0)x(0.0,100.0), and maps that range to an area of NDC space with the same aspect ratio.

```
SET WINDOW (1, 0.0, 100.0, 0.0, 100.0)
SET VIEWPORT (1, 0.0, 1.0, 0.0, 1.0)
SELECT NORMALIZATION TRANSFORMATION (1)
```

Each combination of window and viewport setting defines a separate normalization transformation. In GDDM-GKS, 11 normalization transformations are defined. Transformation 0 is the unity transformation, which maps (0.0,1.0)x(0.0,1.0) in WC to (0.0,1.0)x(0.0,1.0) in NDC; this transformation cannot be changed. Transformations 1 through 10 initially default to the unity transformation, and you can reset them with the Set window (GSWN) and Set viewport (GSVP) functions.

Having defined the normalization transformations you need, you use the Select normalization transformation (GSELNT) function to choose one of them to be the current one; this transformation is then used to transform subsequent output primitives.

Workstation transformations

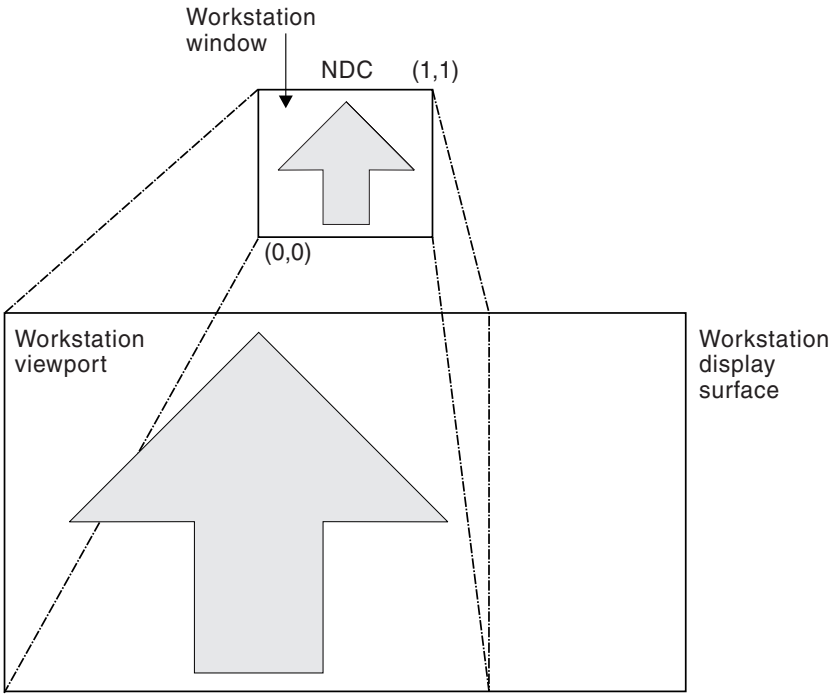
Each workstation has its own workstation transformation. When you set a workstation transformation, you are mapping a range of NDC to a range of DC. The **workstation window**, set with the Set workstation window (GSWKWN) function, encloses the range of NDC points that you want to map to the display surface. (Note that this NDC range is not necessarily the same as a viewport.) The **workstation viewport**, set with the Set workstation viewport (GSWKVP) function, specifies the range of DC points to which your workstation window will be mapped.

The transformation functions provide device-independence and application portability. You can adapt your GKS application to any display device supported by GDDM-GKS. There are two ways you can do this without knowing in advance the specific properties of the device. You can:

- Use the default values for the workstation transformation
- Determine device characteristics using GKS inquiry functions and then set workstation transformations accordingly.

The default workstation window is (0.0,1.0)x(0.0,1.0). The default workstation viewport is the full display surface; only the part that will allow the aspect ratio to be preserved is used.

This illustration shows how the default values for workstation window and workstation viewport preserve the aspect ratio of the picture in NDC.



The largest workstation window that you can set is (0.0,1.0)x(0.0,1.0). The maximum workstation viewport is the full display surface of the workstation.

There is a simple way to ensure that your application makes use of the entire workstation display surface, no matter what kind of output device it may be used with. The function Inquire display space size (GQDSP) returns, among other information, the maximum x and y ranges for DC units. Find the larger of these, and set the aspect ratio of the NDC workstation window equal to the aspect ratio of the display surface. The workstation transformation will then map the graphics picture to the entire display surface.

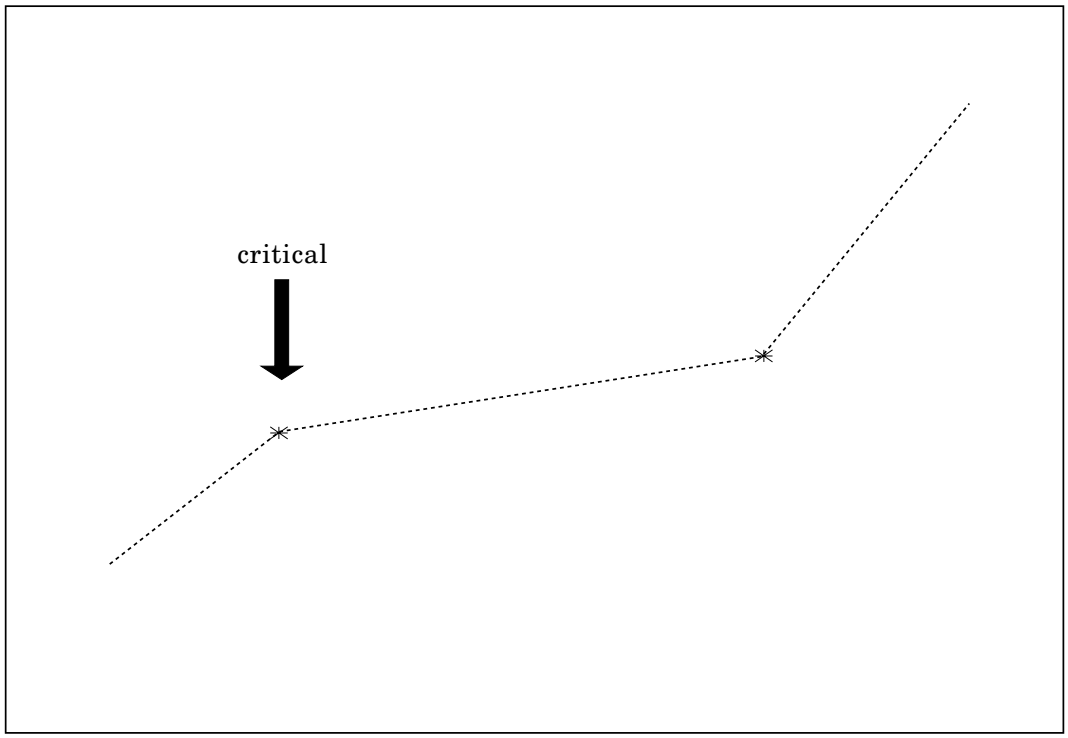
Here is a method for setting the workstation transformation:

```
SET WINDOW (2, 0.0, 100.0, 0.0, 100.0)
INQUIRE DISPLAY SPACE SIZE (wtype, error,
     dcunits, xdevice, ydevice, xraster, yraster)
if xdevice > ydevice
     then scale = xdevice
     else scale = ydevice
xndc = xdevice/scale
yndc = ydevice/scale
SET VIEWPORT (2, 0.0, xndc, 0.0, yndc)
SET WORKSTATION WINDOW (wkid, 0.0, xndc, 0.0,
     yndc)
SET WORKSTATION VIEWPORT (wkid, 0.0, xdevice,
     0.0, ydevice)
```

You can use this procedure to make use of the entire display surface in your application. However, you may lose the one-to-one correspondence with the graphics picture in WC. If after the above procedure, you enter:

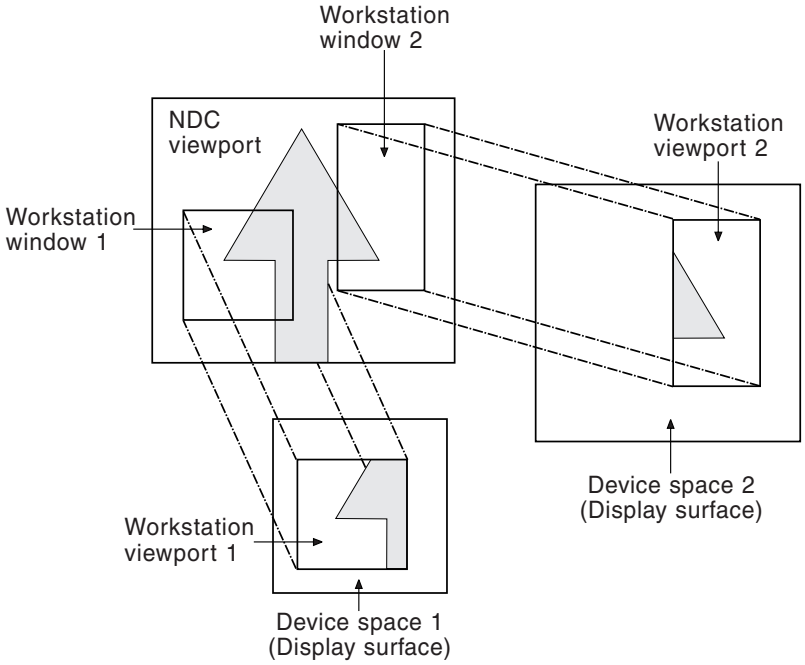
```
SELECT NORMALIZATION TRANSFORMATION (2)
```

and then draw the picture, it looks like this:



When a segment is created, the current normalization transformation is stored with it as a **clipping** rectangle. Therefore subsequent changes in the normalization transformation have no effect on previously created segments. Changing the workstation transformation changes the location of a displayed segment, but not its aspect ratio.

This illustration shows two different workstation transformations.



Segment functions

When primitives are grouped together in a **segment**, you can perform operations on them as a single object. You form a segment by calling the Create segment (GCRSG) function, and then the output functions to create the primitives for the segment. The Close segment (GCLSG) function is called to define the end of the current segment.

The segment function Create segment (GCRSG) sets the system to the operating state SGOP. Only one segment can be open at a time. A segment state list is initialized with default values for the current segment. Each existing segment has a segment state list associated with it until the segment is deleted from all existing workstations. You assign to each segment a unique identifier. All subsequent output primitives are collected into the named segment until the next Close segment (GCLSG) call is made.

When you create a segment, it is automatically stored at all workstations that are active. If you activate a *workstation independent segment storage* (WISS), you can copy any segment from WISS to any other workstation that is activated after the segment is created.

There are three segment functions that require the presence of an open WISS. You can choose between two of them to display a segment on a workstation that was not active when the segment was created. Associate segment with workstation (GASGWK) stores a segment on a workstation as though it had been active at segment creation. Copy segment to workstation (GCSGWK) displays the primitives in a segment on a workstation, but does not store the segment on the workstation state list. The third function, Insert segment (GINSG), adds the contents of a segment stored in WISS to the current open segment, or into the stream of primitives outside segments if no segment is open.

You can delete a segment from GKS with Delete segment (GDSG), or you can delete it from a specified workstation with Delete segment from workstation (GDSGWK).

Segment attributes

Segments have **segment attributes** that you can set using the segment functions. These attributes include:

- Segment transformations (see the next section)
- Visibility
- Detectability
- Segment priority
- Highlighting.

You control the **visibility** of a segment with the Set visibility (GSVIS) function. **Detectability** is controlled by Set detectability (GSDTEC). A segment must be both **detectable** and **visible** to be available for picking during *pick* input.

You assign **segment priority** to segments with Set segment priority (GSSGP). Segment priority determines the display order of segments if overlapping segments are displayed, and governs the outcome of pick input if overlapping segments are picked.

When a segment is **highlighted**, it is displayed in white (if the output device supports this). Highlighting is controlled by the Set highlighting (GSHLIT) function.

Only primitives within segments can be picked during **pick input**. Every primitive within a segment has the *pick identifier* attribute, which you set with the Set pick identifier (GSPKID) function. During pick input, the pick identifier of the primitive is returned along with the segment name. (For more information on pick input devices, see “Input functions” later in this chapter.)

Segment transformations

Segment transformations scale, rotate, and shift primitives that were created within segments. A **segment transformation matrix** is stored in the **segment state list** for each segment as it is created. (See Appendix A, “GKS data structures” on page 359 for details on the segment state list.) This initial, **identity matrix** contains values that have no effect on the appearance of the primitives within the segment (for instance, zero shift, identity scale, zero radians of rotation). The initial matrix is replaced with a new one when you call Set segment transformation (GSSGT).

These transformations are made with a 2x3 transformation matrix, consisting of a 2x2 scaling and rotation portion, and a 2x1 translation portion. GKS includes two utility functions to help you to compose transformation matrixes for segments. Evaluate transformation matrix (GEVTM) composes a matrix from input values for a fixed point, shift vector, rotation angle, and scale factors. Accumulate transformation matrix (GACTM) composes a new matrix from its input values and a matrix you defined previously. After the matrix is composed, you apply it to a segment with Set segment transformation (GSSGT).

In this example, the current display is a segment containing a rectangle with its lower left-corner at (5.0,5.0). To shift the segment so that the lower-left corner will be located at (20.0,20.0), call Evaluate transformation matrix (GEVTM) with fixed point at (5.0,5.0), shift vector as (15.0,15.0) (which means “add 15 to the x and y

values of the fixed point"), rotation angle as 0.0, and scale factor as 1.0 for both x and y.

```
EVALUATE TRANSFORMATION MATRIX (5.0, 5.0, 15.0,
     15.0, 0.0, 1.0, 1.0, 1, Output Matrix)
```

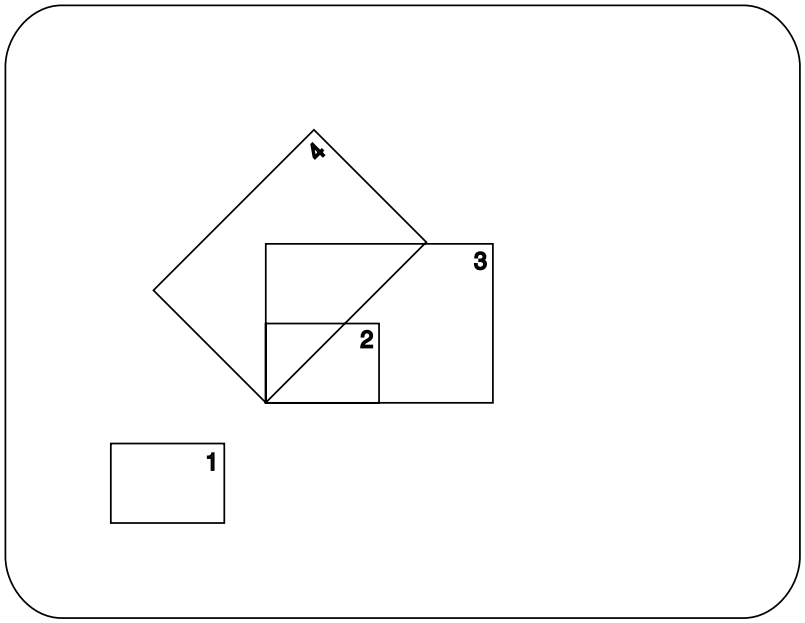
To scale the segment after it is shifted, use the *output matrix* of the previous Evaluate transformation matrix (GEVTM) call as the *input matrix*. Change the fixed point to (20.0, 20.0) and specify the scale factor as 2.0 for both x and y axes.

```
ACCUMULATE TRANSFORMATION MATRIX (Input Matrix,
     20.0, 20.0, 0.0, 0.0, 0.0, 2.0, 2.0, 1,
     Output Matrix)
```

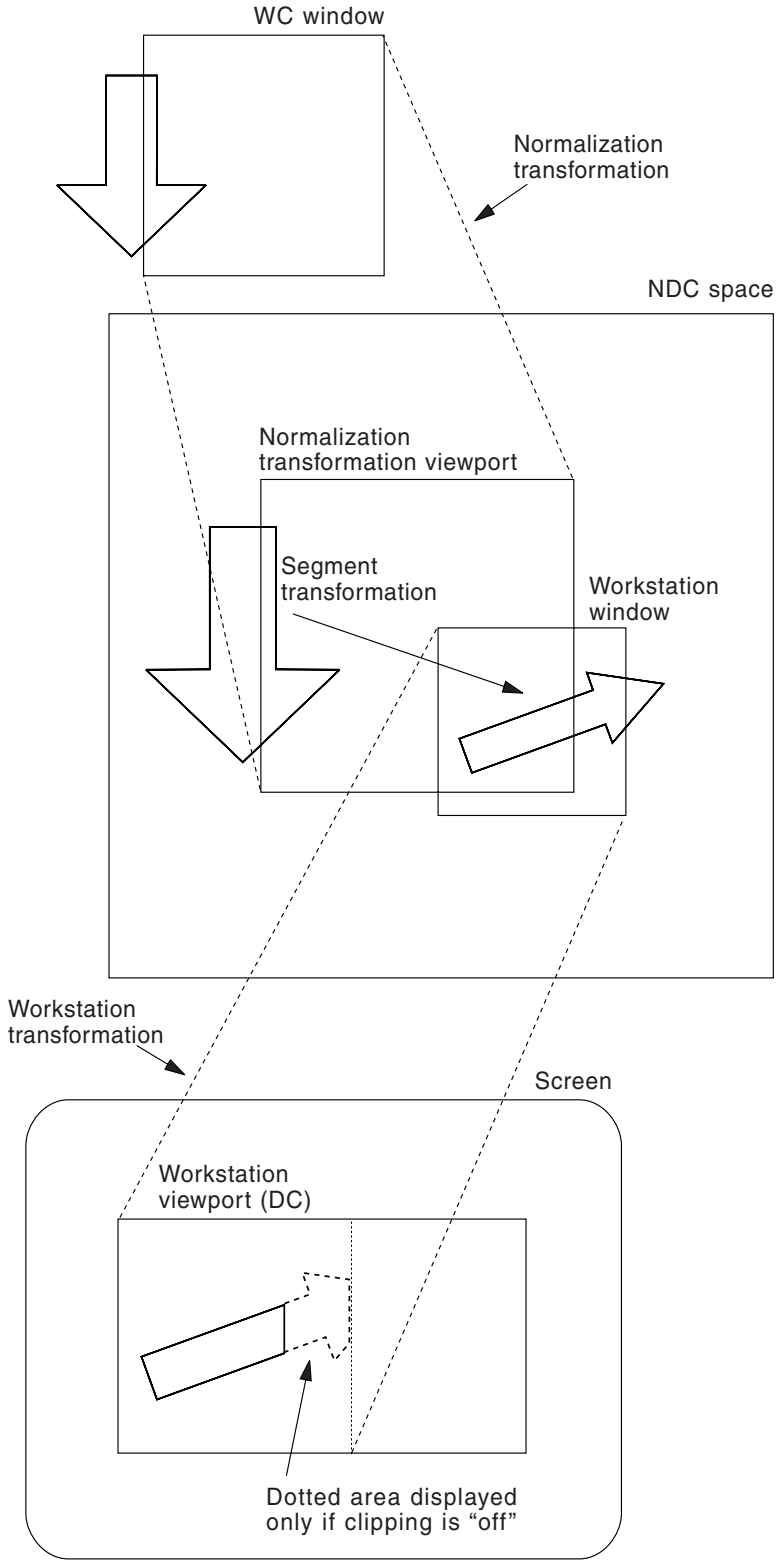
To rotate the scaled and shifted segment 45 degrees, first convert the angle to radians (0.7853982). Then call Accumulate transformation matrix (GACTM) again and enter the rotation value.

```
ACCUMULATE TRANSFORMATION MATRIX (Input Matrix,
     20.0, 20.0, 0.0, 0.0, 0.7853982, 1.0, 1.0, 1,
     Output Matrix)
```

This sequence produces the following composite display:



The diagram on the next page shows the way in which the three types of transformation (normalization, segment, and workstation) are used to build the image on the display surface.


WC window
Normalization
transformation
NDC space
Normalization
transformation viewport
Segment
transformation
Workstation
window
Workstation
transformation
Screen
Workstation
viewport (DC)
Dotted area displayed
only if clipping is “off”

Input functions

GDDM-GKS supports six input classes for interactive graphics programs: *locator*, *valuator*, *choice*, *pick*, *stroke*, and *string*. A logical input device is an abstract device that delivers input data to your application program, based on one of the six input classes.

You **initialize** a logical input device in GKS by establishing an **initial value**, a **prompt and echo type**, an **echo area**, and an initial data record for the device. There is a function for initializing each input class. There are default values for all these parameters, and no initialization is necessary unless the defaults must be changed.

Input classes

The **input class** determines the type of logical input value, or "measure," that the device delivers. The six input classes are

- **Locator**. Selects a position on the display surface, usually by moving a graphics input cursor or a cross-hair cursor to the desired position. The logical input value returned by a locator is a point in world coordinates, together with a normalization transformation number. (See Note below.)
- **Valuator**. Returns a real number corresponding to the condition of a physical input device, such as a position on a dial, or a number entered by the operator.
- **Choice**. Returns a non-negative integer value that represents a selection from a number of choices. Usually the operator chooses one of the fixed alternatives by pressing a button or function key.
- **Pick**. Returns a a segment name and a pick identifier. Typically a pick input operation consists of positioning a graphics input cursor or cross-hair cursor over a primitive and, for example, pressing Enter.
- **Stroke**. Returns a series of points in world coordinates together with a normalization transformation number. (See Note below.) Typical stroke devices include graphics tablets, mice, and graphics cursors.
- **String**. Returns a character string, typically from a keyboard.

Note: If multiple normalization transformations containing the input point are defined, the results of locator and stroke input depend on the **viewport input priority**; this is set by using the Set viewport input priority (GSVPIP) function.

Input modes

In GDDM-GKS, input functions operate in **request** mode.

You obtain input by calling the Request *** function for the appropriate device. The program operation is held until the operator responds with a request mode trigger. For example, when you call Request locator (GRQLC), program processing is halted until a user moves the cursor to the desired position on the screen and presses Enter at the keyboard as a trigger to terminate request mode. The value returned by the logical input device is the current measure of the physical workstation (for instance, cursor position or the value of a pressed function key).

Prompt and echo types

Logical input devices are equipped with **prompt** and **echo** facilities that inform the user of the state of input processing. Echoing provides feedback about input. There is an echo switch that you can set on or off by calling the Set *** mode function for the appropriate device. The default echo area for valuator, string, and choice devices is the upper right quadrant of the display surface. The default echo area for locator, stroke, and pick devices is the maximum display surface size. You can change the default echo area by calling the Initialize *** function for the appropriate input class.

Metafile functions

A GKS metafile (GKSM) is a sequential file that you can use for long-term storage of graphical and non-graphical data. You could use GKSMs for transporting data between GKS applications or between different physical locations.

GKSMs are GKS workstations; there are two types:

- MO - metafile output
- MI - metafile input.

Several different MO or MI GKSMs can be used concurrently. A metafile can have data written into it as an MO workstation; then the workstation can be closed, and the metafile can be opened as an MI workstation to retrieve the data.

Graphical output data is sent to an active MO workstation in the same way as to a physical-device workstation, but the data is stored, and no graphics image is created. In addition, the Write item to GKSM (GWITM) function is provided, to allow you to add user data to the metafile.

To retrieve data, three functions are used:

- Get item type from GKSM (GGTITM) - yields information about the current item;
- Read item from GKSM (GRDITM) - reads an item and makes the next item current
- Interpret item (GIITM) - uses information in an item (that has been read from the GKSM) to make changes in the GKS state variables, and generate a graphics image on all active workstations.

Utility functions

GKS provides four **utility** functions to help you to compute transformation matrixes and to process packed data records:

- Evaluate transformation matrix (GEVTM) computes segment transformation matrixes from input values for shift, rotation, and scale.
- Accumulate transformation matrix (GACTM) composes a new matrix from an input transformation matrix plus new input values for shift, rotation, and scale.
- Pack data record (GPRECS) accepts individual unpacked elements of a data record, and returns them in packed format.
- Unpack data record (GURECS) performs the inverse of the Pack data record (GPRECS) function.

Packed data records are required by the input functions that initialize logical input devices.

Transformation matrixes are required as input parameters for Set segment transformation (GSSGT) and Insert segment (GINSG).

Inquiry functions

GKS provides many **inquiry** functions that return information to your application programs about the current state of the system. Inquiry functions return information about:

- Primitives and segments
- Normalization and workstation transformations
- GKS operating levels
- Workstation capabilities
- Workstation types
- Pick identifiers
- Input device states
- Default data for input devices.

The Inquiry functions return values derived from the state lists and description tables established by GKS (see Appendix A, “GKS data structures” on page 359 for more information).

A normal return is indicated by a zero in the error indicator output parameter; if the data requested is unavailable, the parameter contains a GKS error number.

The inquiry functions are useful if you need specific information about the system and its workstations. In this example, an inquiry function returns data needed to set the workstation viewport to be the lower left quadrant of the display surface.

```
INQUIRE DISPLAY SPACE SIZE (wktype, error,
     dcunits, xdevice, ydevice, xraster, yraster)
SET WORKSTATION VIEWPORT (wkid, 0.0, xdevice/2.0,
     0.0, ydevice/2.0)
```

Error handling

The following functions are provided by GKS for handling errors detected during calls to GKS:

- Error handling (GERHND) is called by GKS when an error is detected during a call to a GKS function. The function can be replaced by an application-supplied error-handling procedure, allowing customized processing when errors are detected. The application-supplied error-handling procedure can invoke any of the GKS inquiry functions; it can also invoke either the Error logging (GERLOG) function or the Emergency close GKS (GECLKS) function (or both) but it may not alter the GKS state by invoking any other GKS functions. The Error handling (GERHND) function merely calls the Error logging (GERLOG) function described below.
- Error logging (GERLOG) can be called by the error-handling procedure when an error has been detected by GKS. It causes a message containing the GKS error number, the function in which it was detected, and explanatory text to be

written to the error file that is specified when the Open GKS (GOPKS) function is called.

- Emergency close GKS (GECLKS) can be called by an application or by an application-supplied error-handling procedure when errors are found. Its function is to close GKS, but first to try to save as much of the graphical information produced as possible.

Errors detected by the GKS inquiry functions do not cause the error-handling procedure to be invoked, because the error number is returned in an error indicator output parameter.

Error handling and application-supplied error-handling procedures are discussed in more detail in Chapter 4, “Using GDDM-GKS” on page 43.

Designing an interactive graphics program

This section contains tutorial information that builds upon the information presented earlier in this chapter. It explains some details about argument conventions, and outlines an interactive graphics program using GDDM-GKS.

Argument conventions

Values in arguments passed to GKS must be valid. That is, they must be meaningful and in the correct range. However, some aspects of GKS are workstation-dependent. For example, the availability of a particular color depends on the capabilities of the workstation. This means that an input argument can be **valid** (allowed as an input) but **not supported** by the particular device. In this case, GKS chooses the closest color available on the device.

It is important to remember that all arguments must be allocated for any call made to GKS. This is true even if a particular argument has a “don’t care” value. For example, in the function Initialize pick (GINPK), which sets initial values for a pick input device, the arguments for **data record** and **length of data record** are not used. However, you must initialize variable storage for both arguments and include them in the call. This procedure ensures not only the success of that particular call in the application program, but also the compatibility of your application program with other implementations of GKS that may use those arguments. If the space allocated for an output argument is insufficient, or if the arguments are not of the correct type, unpredictable results can occur.

The ROOM program

The ROOM program is an interactive GKS graphics program. Source code for this program, suitable for compilation using either VS FORTRAN or FORTRAN IV compilers, is included on your GDDM-GKS distribution tape.

First the objectives of the program are described. Then the program’s structure is outlined. Finally, some parts of the program are explained in detail, to show you how various functions are used together. The examples are written for the VS FORTRAN compiler.

Program objectives

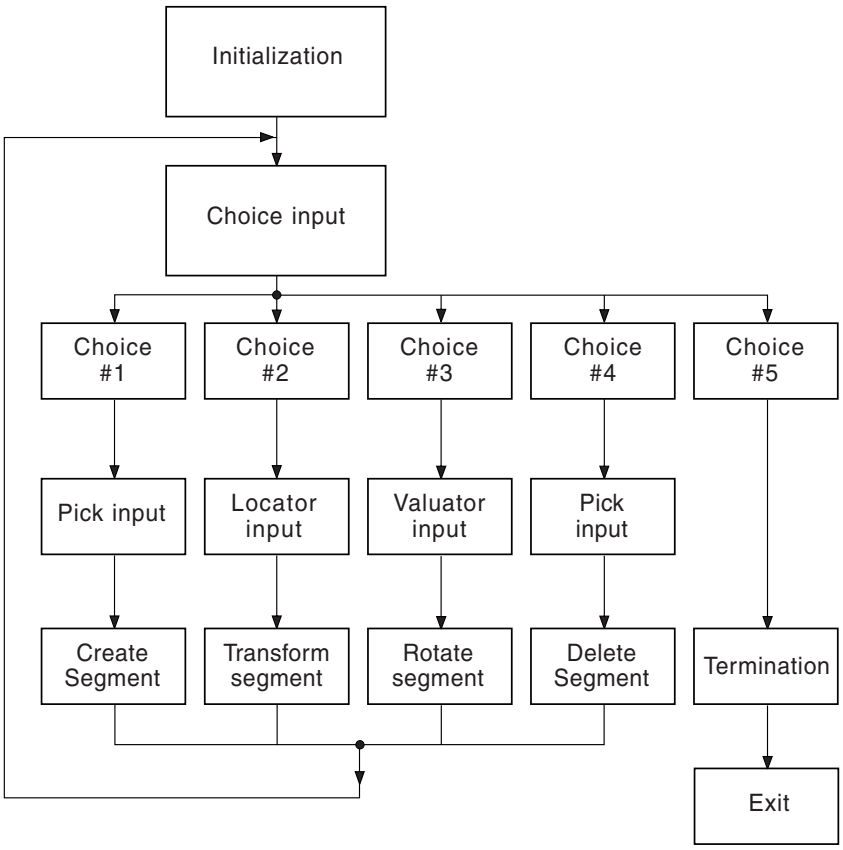
The ROOM program meets these objectives:

- Displays the outline of a room, message prompts, and menu for input
- Lets a user pick, locate, rotate, or delete pieces of furniture from the room.

ROOM program outline

1. Open GKS.
2. Open all workstations to be used, and activate all *output* or *outin* workstations.
3. Set up normalization transformation and workstation transformations.
4. Create pictures on the screen, organized into segments.
5. Display the menu of function key options for choice input.
6. Accept choice input and branch according to the input choice number.
7. Accept pick input for a piece of furniture to be placed in the room.
8. Create a new segment in response to pick input.
9. Accept locator input for furniture placement.
10. Transform the segment to shift it to the chosen location.
11. Accept valuator input for a rotation value.
12. Transform the segment according to the rotation value entered.
13. Delete a segment when the user chooses that option.
14. Deactivate all workstations that are active, and close all workstations.
15. Close GKS.

Diagram of the program



Initializing GKS

This procedure initializes the system by opening GKS.

```
OPEN GKS (errorfile, buffer size)
```

GKS is opened, and the file that is defined in the *errorfile* parameter is opened by GKS to receive GKS error messages.

Opening and activating workstations

```
wkid=1
wktype=1
OPEN WORKSTATION (wkid, 0, wktype)
ACTIVATE WORKSTATION (wkid)
```

Here the user console workstation is opened, and given a workstation identifier of 1. The workstation is then activated.

All workstations in a GKS program must be **opened** before they can be used for input or output. When you **activate** a workstation, it receives subsequent GKS output or stored segments. Only *output*, *outin*, *WISS*, and *MO* workstations are activated.

Normalization and workstation transformations

The normalization and workstation transformations together represent a scaling of world coordinates to device coordinates. Each transformation consists of a window and a viewport.

With the transformations set in the following way, the example program uses the entire screen as an output surface. This setting causes the aspect ratio of the display to be different from the aspect ratio of the picture as created in WC space. That is, a square will be output as a rectangle.

```
     SET WINDOW (1, 0.0, 100.0, 0.0, 100.0)
     INQUIRE DISPLAY SPACE SIZE (worktype,
          error, dcunits, xdevice, ydevice, xraster,
          yraster)
Find the larger of the two dimensions
     scale = xdevice
     If xdevice < ydevice
          then scale = ydevice
Calculate the aspect ratio
     xndc = xdevice / scale
     yndc = ydevice / scale
     SET VIEWPORT (1, 0.0, xndc, 0.0, yndc)
     SET WORKSTATION WINDOW (wkid, 0.0, xndc,
          0.0, yndc)
     SELECT NORMALIZATION TRANSFORMATION (1)
```

The range for the world window is set from 0.0 to 100.0 along each axis. Then the maximum display surface size is inquired, and the largest dimension in device coordinates is determined. The NDC viewport is scaled to have the same aspect ratio as the device surface. The workstation window is set equal to the world viewport (both in NDC). The workstation window is then mapped to the entire display surface. Finally the normalization transformation just defined is selected.

Primitives and segments

The primitives for the program must be created within segments to make them pickable during pick input. The initial picture consists of three segments.

The first segment consists of a panel containing two items of furniture. The primitives for the desk and chair are output after pick identifiers are set for each item. The Set pick identifier (GSPKID) function designates the current pick identifier. In GKS, primitives are associated with the pick identifier that was current when they were created. When a primitive is picked during pick input, the unique pick identifier is returned along with the associated segment name. In this case, the segment name is the same for the desk and the chair, although the pick identifiers are different.

```
Create the first segment
     CREATE SEGMENT (100)
Set pick identifier for desk
     SET PICK IDENTIFIER (101)
Set fill color to background color
     SET FILL AREA COLOR INDEX (0)
Set fill style to solid
     SET FILL AREA INTERIOR STYLE (1)
Output invisible fill area
     FILL AREA (5, xdray, ydray)
Set line color to foreground
     SET POLYLINE COLOR INDEX (1)
Output polyline with 5 points
     POLYLINE (5, xdray, ydray)
Set text color
     SET TEXT COLOR INDEX (3)
Output text identifier
     TEXT (78.5, 92.0, 'desk')
Set pick identifier for chair
     SET PICK IDENTIFIER (102)
Output invisible fill area
     FILL AREA (5, xchray3, ychray3)
Output polyline with 9 points
     POLYLINE (9, xchray1, ychray1)
Output polyline with 2 points
     POLYLINE (2, xchray2, ychray2)
Output text identifier
     TEXT (86.0, 73.5, 'chair')
Close first segment
     CLOSE SEGMENT
```

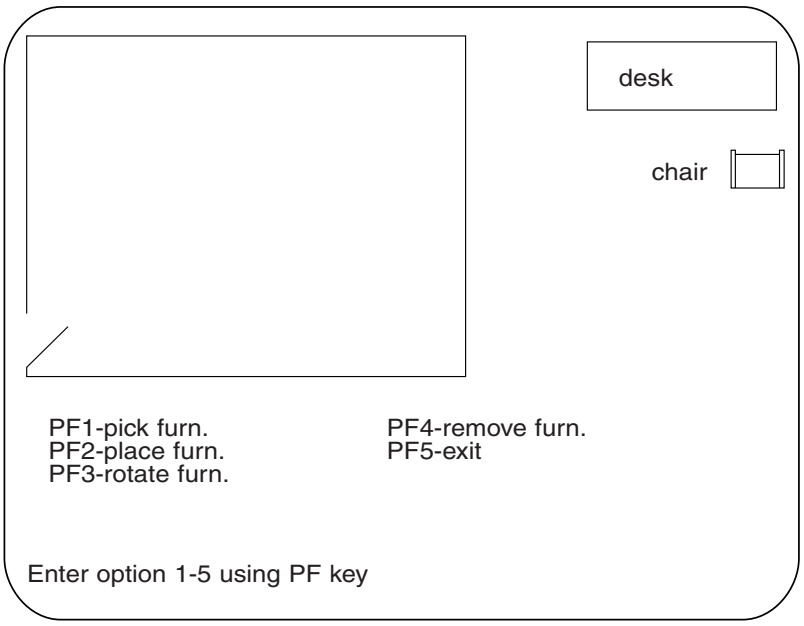
During pick input, the user moves a graphics cursor or cross-hair cursor over a part of the primitive to be picked and presses Enter. The cross-hair cursor must be over one of the points that defines the primitive for a valid pick to be registered. This program places congruent invisible fill areas (drawn in background color) behind the visible primitives. This makes it easier to pick a primitive, because the cross-hair cursor can be anywhere inside the perimeter of the fill area for a valid pick.

The second segment consists of a polyline connecting seven points.

```
create second segment
     CREATE SEGMENT (200)
set line color
     SET POLYLINE COLOR INDEX (2)
output lines for room outline
     POLYLINE (7, xrray1, yrray1)
close second segment
     CLOSE SEGMENT
```

The third segment consists of the menu of options for choice input.

This is what the screen looks like after the initial segments are generated:



Choice input

ROOM is a simple menu-driven program. The menu is controlled by a GKS choice input device. The user selects program options by pressing one of the function keys. The program loops at this point until a valid choice has been made. Upon receiving a valid choice, the program branches to make the appropriate response.

No call to Initialize choice (GINCH) is necessary, because the program uses the default initial values for the choice device.

```
Set choice device to request mode, no echo, for PF keys
     SET CHOICE MODE (wkid, 2, 0, 0)
     LOOP:
          UNTIL STATUS = 1
          AND CHOICE NUMBER >=1
          AND CHOICE NUMBER <= 5
          REQUEST CHOICE (wkid, 2, status,
               choice number)
     END LOOP
```

The status parameter indicates whether a valid choice was made at the device, which in this instance means whether a function key was pressed. The application then determines whether the choice was within the valid range of 1 to 5. These options represent branches to the 5 major operations left in the program:

- **Pick input** to select a piece of furniture
- **Locator input** to place the furniture
- **Valuator input** to enter a rotation angle for a **segment transformation**
- **Pick input** to select a furniture segment to be deleted
- **Exiting the program**.

Pick input

When the operator chooses PF1 (“pick furniture”) the program branches here. The pick device is set to **request mode**, and the echo facility is enabled. The program loops until a valid pick is made.

```
set pick device to request mode, echo on
     SET PICK MODE (wkid, 1, 0, 1)
     LOOP:
          UNTIL STATUS = 1
          AND SEGMENT NAME = 100
          REQUEST PICK (wkid, 1, status, segment name,
               pick identifier)
     END LOOP
```

The status parameter indicates whether a valid pick was made at the device. A pick is valid when a primitive within a segment is picked. The application then determines whether the segment containing the picked primitive is segment 100, which contains the desk and chair primitives. Each primitive has a unique pick identifier returned by the Request pick (GRQPK) function.

Creating a new segment

The program creates a new segment according to the returned pick identifier. The new segment contains the piece of furniture that the operator wants to move into the room. The first new segment is given the number 1, and subsequent new segments are named in increments of 1.

```
     CREATE SEGMENT (segment name)
output fill area in background color
     FILL AREA (5, xdsk, ydsk)
     POLYLINE (5, xdsk, ydsk)
make new segment invisible
     SET VISIBILITY (segment name, 0)
     CLOSE SEGMENT
```

The new segment is created at the origin (0,0) and made **invisible** so that it cannot appear on the display until it is placed. A pick identifier is assigned to it so that it can be identified later if the operator chooses to remove it from the room.

After the new segment is created, the program branches back to the menu of options.

Locator input

When the operator chooses PF2 (“place furniture”) the program branches here. The locator device is initialized so that the **initial cursor position** is the center of the room.

```
     INITIALIZE LOCATOR (wkid, 1, 1, 29.93,
          69.51, 1, 0.0, rx, 0.0, ry, 0, datarcd)
```

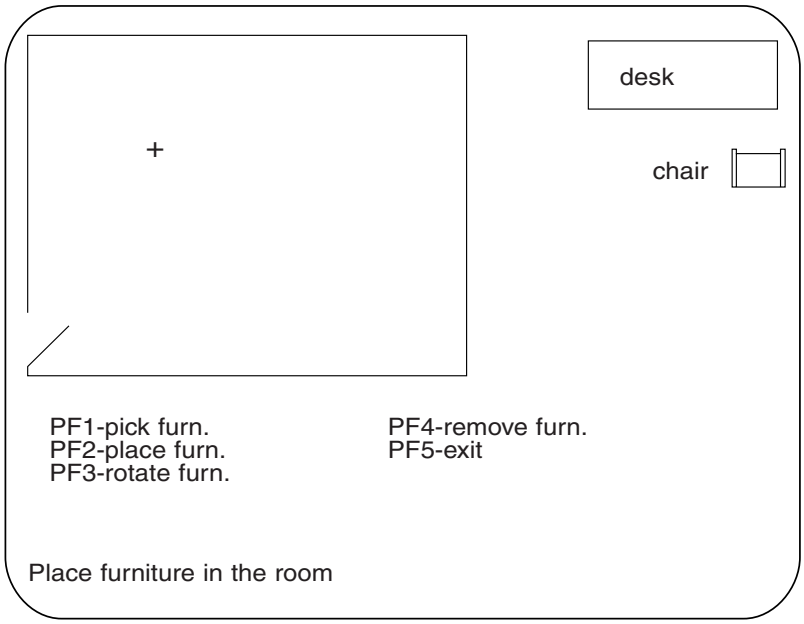
The device is set to **request** mode, and the echo facility is enabled. The program loops until a valid locator point is chosen by the operator.

```
SET LOCATOR MODE (wkid, 1, 0, 1)
LOOP:
     UNTIL STATUS = 1
     AND XCOR < 60.0
     AND YCOR > 40.0
     REQUEST LOCATOR (wkid, 1, status,
          transformation number, xcor, ycor)
END LOOP
```

The status parameter indicates whether a valid location was chosen at the device. The program then checks if the chosen point is valid, which in this instance means whether the point chosen was within the perimeter of the room outline.

The echo type (1) chosen in the Initialize locator (GINLC) function call echoes the operator input by indicating the current cursor position.

This is what the screen looks like during locator input:



Segment transformations

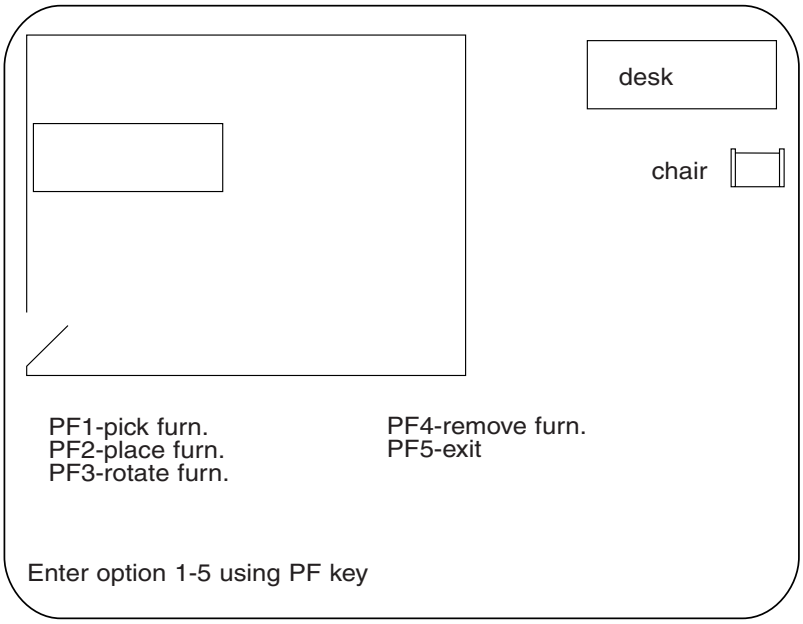
To place the piece of furniture inside the room, a segment transformation must be applied to the new segment. The program uses the point returned by the locator input to compute a **shift vector** that does this. You enter a shift vector for the transformation as a relative displacement value from a fixed point. In this case, the new segment is located at the origin (0,0). The shift vector should then be the locator point less the dimensions of the furniture itself.

```
EVALUATE TRANSFORMATION MATRIX (0.0, 0.0,
     xcor-furnwidth, ycor-furnheight, 0.0, 1.0,
     1.0, 0, matrix1)
SET SEGMENT TRANSFORMATION (segment name, matrix1)
SET VISIBILITY (segment name, 1)
```

After the matrix is evaluated, the transformation is set and the piece of furniture is moved inside the room. Visibility is then set to 1 so that the primitive appears on the display.

The program branches back to the menu of options.

This is what the screen looks like after the desk has been picked and located within the room:



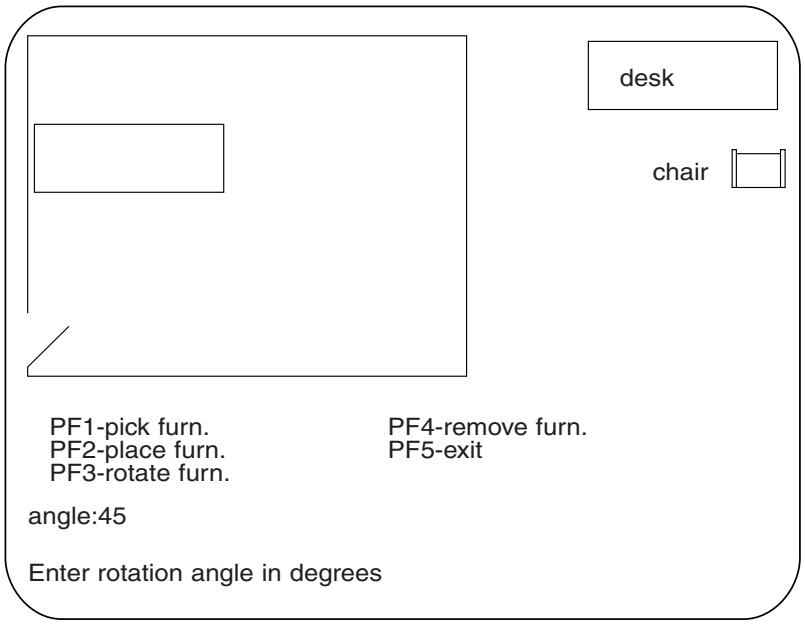
Valuator input

When the operator chooses PF3 (“rotate furniture”) the program branches here. The Text (GTXS) function is used to prompt the operator to enter a rotation angle. Then the Request valuator (GRQVL) function calls for input from the valuator device on the workstation:

```
LOOP:
     UNTIL STATUS = 1
     REQUEST VALUATOR (wkid, 1, status, value)
END LOOP
```

The status parameter indicates whether a value was entered at the device. The program converts the value to radians and uses it for the rotation value in the next segment transformation.

Here is what the screen looks like during valuator input:



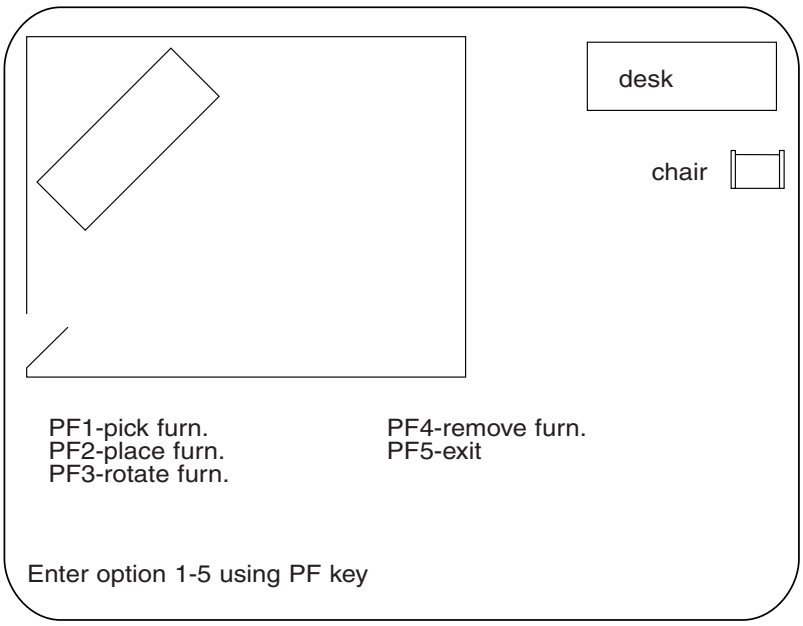
Accumulating transformation matrixes

The effects of segment transformations are not cumulative. That is, the matrixes are applied to segments as they were originally created. The first transformation matrix moved the piece of furniture from the origin (0,0) to a designated position in the room. The next transformation must rotate the item without letting it assume its original position at the origin. The Accumulate transformation matrix (GACTM) function accepts the previous matrix, **matrix1**, as input, and composes it with the new value for rotation.

```
ACCUMULATE TRANSFORMATION MATRIX (matrix1, xcor,
     ycor, 0.0, 0.0, angle, 1.0, 1.0, 0, matrix2)
SET SEGMENT TRANSFORMATION (segment name, matrix2)
```

The program branches back to the menu of options.

Here is what the screen looks like after the desk has been rotated 45 degrees:



Deleting a segment

When the operator chooses PF4 (“remove furniture”) the program branches here. If only the original three segments exist, the program issues the error message “No furniture is in the room.” If there are more than three segments, the program calls Request pick (GRQPK) to let the operator pick the segment to be deleted.

```
LOOP:
     UNTIL STATUS = 1
     AND SEGMENT NAME < 100
     REQUEST PICK (wkid, 1, status, segment name,
          pick identifier)
END LOOP
DELETE SEGMENT (segment name)
```

The deleted segment disappears immediately from the display. The program branches back to the menu of options.

Deactivating and closing workstations

When the operator chooses PF5 (“exit”) the program branches here.

To exit, the program follows the reverse of the sequence used to open the system and open and activate its workstation. First, it must deactivate any workstations that were active and close any that were open.

```
deactivate workstation 1
     DEACTIVATE WORKSTATION (wkid)
close workstation 1
     CLOSE WORKSTATION (wkid)
```

Closing GKS

After the workstation is deactivated and closed, the program closes GKS and the error file.

```
     CLOSE GKS
```

Chapter 3. Introduction to GDDM-GKS

This chapter introduces you to GDDM-GKS, which is an implementation of GKS. It describes the following:

- The relationship between GDDM-GKS and GDDM Base
- Differences between GDDM-GKS and the GKS standards
- The GDDM-GKS FORTRAN binding
- Programming languages other than FORTRAN
- Non-GKS external names.

Relation to GDDM Base

GDDM-GKS is an IBM licensed program; GDDM/VM or GDDM/MVS (referred to generically in this book as GDDM Base) is a prerequisite. Within GDDM's overall aim of providing workstation-independent image, graphics, and alphanumerics support, GKS provides a set of graphics functions through an application programming interface (API), which can be used as an alternative to GDDM Base graphics.

Relation to the GKS standard

This implementation is at level 2b of the ISO GKS standard (with some minor differences detailed in the next section). The following facilities are available:

- Full output, including full bundle concept
- Multiple normalization transformations
- Multiple workstations
- Full workstation control
- Metafile workstations
- Full segmentation, including *workstation independent segment storage*
- Request input mode
- Viewport input priorities
- Pick input.

Differences between GDDM-GKS and the ISO and ANS GKS standards

1. Facilities that are not defined (or that are implementation-dependent) in ISO GKS, but are defined by ANS GKS, are implemented to conform to ANS GKS. These facilities include:

   Data records for graphical input devices

   Initial setting of aspect source flags to *individual*.
2. The *pattern reference point* attribute is ignored on graphics devices.
3. The *pattern size* attribute is ignored on graphics devices.

© Copyright IBM Corp. 1984, 1987

4. Patterns are not transformed; user-defined pattern representations are truncated or padded, if necessary, to fit the cell size of graphics devices.
5. The display area of a workstation is considered to be the (GDDM) graphics field defined (implicitly or explicitly using non-GKS facilities) for the workstation.
6. It is not possible to invoke any GKS functions before GKS is opened, unless GDDM has been initialized by an FSINIT call.
7. The text fonts provided do not support *cap*, *half*, and *base* alignment; the cap line is considered to be the same as the top line and the base line is treated as being the same as the bottom line. *Half* alignment is implemented as the midpoint between the top and bottom lines. For precise positioning, top and bottom alignment should be used.
8. The functions Inquire pixel (GQPX), Inquire pixel array (GQPXA), and Inquire pixel array dimensions (GQPXAD) are not supported; the workstations supported by GDDM-GKS have no pixel store read-back capability.
9. The characteristics for workstation type 1 are determined dynamically, rather than from a static table.

FORTRAN binding

The GDDM-GKS application programming interface (API) is based on the ANSI GKS language binding for FORTRAN.

The binding defines the call syntax and parameters to be used when GKS functions are invoked from FORTRAN applications. Both the FORTRAN 77 and FORTRAN 77 Subset function calls are provided.

To aid program portability, entry points are provided for all GKS functions defined in the standard, including level 2c functions, which are not required for a level 2b implementation of GKS.

Programming languages other than FORTRAN

No bindings are provided for programming languages other than FORTRAN; however, entry points are provided so that all of the GKS functions can be invoked from the following languages:

- PL/I
- COBOL
- System/370 Assembler Language
- APL2 and VS APL
- IBM BASIC.

Details are given in “Considerations for languages other than FORTRAN” on page 46.

Non-GKS external names

This section lists identifiers of non-GKS procedures, functions, data areas, and files, that may be visible to applications or to the subsystem, when GDDM-GKS is used.

The list is provided to avoid potential name clashes if applications are transported from other GKS implementations to GDDM-GKS.

The list is:

1. All identifiers beginning with the prefix ADM.
2. The names of GDDM Base calls; these are described in the *GDDM Base Application Programming Reference* book. All GDDM Base call names have one of the following formats:

   - ASxxxx
   - DSxxxx
   - ESxxxx
   - FSxxxx
   - GSxxxx
   - IMxxxx
   - ISxxxx
   - MSxxxx
   - PSxxxx
   - PTxxxx
   - SPxxxx
   - SSxxxx
   - WSxxxx.

Chapter 4. Using GDDM-GKS

This chapter describes:

- How to use the three types of programming interface to GDDM-GKS (the nonreentrant, reentrant, and system programmer interfaces)
- Programming-language-dependent considerations
- Mixing GDDM-GKS functions and other GDDM functions.
- Using GDDM-GKS in a windowing environment
- Using GDDM-GKS workstations
- How workstations are opened
- How to replace the default error-handling procedure
- Using GDDM-GKS under various subsystems.

Invocation of GDDM-GKS functions

Access to the GDDM-GKS functions by the application program is by GDDM interface modules that are link-edited or loaded with the program. The interface modules convert call statements in the program to a standard internal interface to invoke the GDDM-GKS functions. This makes GDDM-GKS itself independent of the subsystem being used, and allows the use of three different application interfaces:

Nonreentrant interfaces

This is the standard interface for most application programs that use GDDM-GKS and do not require any special processing. All applications written to conform to the FORTRAN binding use this interface.

Reentrant interface

This allows the programs using GDDM-GKS to be made reentrant with the advantages that reentrancy provides; that is, the ability of the program to be used by more than one user at the same time.

Applications written to use the reentrant interface do not conform to the GKS standards, and will need changing before they can be transported to other GKS implementations.

System programmer interface (SPI)

This is provided for programmers who intend to use GDDM-GKS as the basis for a graphics system of their own. It enables GDDM-GKS function calls to be written in a coded form and gives greater control over the subsystem environment.

An application program using the nonreentrant interface cannot use either of the other interfaces. An application program can use the reentrant and system programmer interfaces interchangeably.

© Copyright IBM Corp. 1984, 1987

The nonreentrant interface

The nonreentrant interface applies to application programs that need not be reentrant; for example, a program written in FORTRAN.

Each call statement takes the form:

```
CALL ffffff (parameter, ... )
```

where `ffffff` and the parameter(s) are the appropriate GKS function name and parameters as described in Chapter 5, “GDDM-GKS functions.”

The reentrant interface

Application programs requiring reentrancy can use another form:

```
CALL ffffff (AAB, parameter, ... )
```

where `ffffff` and the parameters are as described for the nonreentrant interface, and `AAB` (application anchor block) is an application-provided, word-aligned control block of the following format:

Offset	Length	Name	Description
0	8	AAB	Application anchor block
0	4	AABFC	GDDM feedback code
0	2	AABSC	GDDM severity code
2	2	AABEC	GDDM error code
4	4	AABAP	GDDM anchor pointer

When using this interface, the application program must provide the storage for the AAB (at least 8 bytes for GDDM’s use). The program is free to extend the AAB for other uses (typically, to provide for passing information to an application-defined error-handling procedure).

The GDDM anchor pointer (AABAP) is set by the GDDM application interface component at initialization, and identifies the GDDM instance being addressed. It is reset to zero on termination. This pointer is used by GDDM to retain storage across activations.

The severity code (AABSC) is set to the error severity code on return to the application:

0 Normal
4 Warning
8 Error
12 Severe error
16 Unrecoverable errors.

The error code (AABEC) is the related GDDM-GKS error message number; this is not the same as the GKS error number.

Reentrancy of the GDDM invocation is determined by the reentrant properties of the AAB. If the AAB is in reentrant storage, GDDM is reentrant.

If the application program is modular, and reentrant use of GDDM from several modules is required, each such module must have access to the AAB. For example, the program can pass the AAB as a parameter across its module calls. If

an application-defined error-handling procedure is used, this procedure must accept the AAB as the first parameter.

The system programmer interface

The system programmer interface (SPI) is a special interface available to "system programming" types of applications. It is available only in reentrant form, and shares many features with the reentrant interface.

Each call takes the form:

```
CALL ADMASP (AAB, RCP, component parameters, ... )
```

where `ADMASP` is the defined system programmer interface entry point, `AAB` is as defined for the reentrant interface, `RCP` is the request control parameter (defined below), and `component parameters` are the parameters for the function specified in the RCP.

The RCP is a 4-byte, fullword-aligned function code defining the GDDM Base or GDDM-GKS function to be called. The RCP code for each GDDM-GKS function call is given, in both hexadecimal and decimal format, with the description of the call, later in this chapter. Also, all the RCP codes are listed in Appendix F, "GDDM-GKS RCP codes" on page 419. For the RCP codes for all GDDM Base and GDDM-PGF functions, refer to the *GDDM Diagnosis* book.

ADMASP is a single entry point resolved by the GDDM interface modules that are link-edited or loaded with the application. The use of the AAB is as described for the reentrant interface. Calls to the system programmer and reentrant interfaces may be mixed, if the same AAB is passed on each call.

In the simplest case, the system programmer interface merely provides a means of accessing a GDDM-GKS function by a function code (the RCP) rather than by selecting an entry point.

This interface provides an alternative initialization function (SPINIT) that allows control of environmental aspects. The SPINIT function is described in the *GDDM-PGF Programming Reference* book.

FORTRAN language considerations

GDDM-GKS FORTRAN application programs can be written for the following compilers:

- VS FORTRAN
- FORTRAN IV G
- FORTRAN IV H

VS FORTRAN

The GDDM-GKS functions can be invoked from GKS applications written in VS FORTRAN using the FORTRAN 77 function calls.

Because the FORTRAN 77 binding uses variable-length strings when passing character arguments, some calls are provided as VS FORTRAN subroutines that are linked with the application program. These calls cannot be invoked using the reentrant interface, the system programmer interface, or programs that use the

system programmer interface. The entry points are contained in the VS FORTRAN binding routine, ADMJB77. The calls are:

GGTST	Get string
GGINST	Initialize string
GMSG	Message
GQSTS	Inquire string device state
GQTXX	Inquire text extent
GRQST	Request string
GSMST	Sample string
GTX	Text
GPREC	Pack data record
GUREC	Unpack data record

FORTRAN IV

GDDM-GKS functions can be invoked from GKS applications written in FORTRAN IV using the FORTRAN 77 Subset function calls.

Because the FORTRAN 77 Subset binding assumes that, for some calls, character strings are always 80 bytes long, these calls are provided as FORTRAN IV subroutines that are linked with the application program. These calls cannot be invoked using the reentrant interface, the system programmer interface, or programs that use the system programmer interface. The entry points are contained in the FORTRAN IV binding routine, ADMJBIV. The calls are:

GGTST	Get string
GGINST	Initialize string
GQSTS	Inquire string device state
GRQST	Request string
GSMST	Sample string
GPREC	Pack data record
GUREC	Unpack data record

For messages, text primitives, and when inquiring the text extent, use the following function calls; the names are formed by appending an “S” to the VS FORTRAN binding name:

GMSGS	Message
GQTXXS	Inquire text extent
GTXS	Text

Considerations for languages other than FORTRAN

GDDM-GKS functions can also be called from applications implemented in the other GDDM programming languages.

Because other languages cannot access the entry points contained in the FORTRAN binding routines, ADMJB77 and ADMJBIV, the following calls are provided; the names are formed by appending an “S” to the VS FORTRAN binding name:

GGTSTS	Get string
GGINSTS	Initialize string
GMSGS	Message
GQSTSS	Inquire string device state

GQTXXS	Inquire text extent
GRQSTS	Request string
GSMSTS	Sample string
GTXS	Text
GPRECS	Pack data record
GURECS	Unpack data record

Each of the following sections contains information about one of the other programming languages (PL/I, COBOL, assembler language, APL, and IBM BASIC). For more information, refer to the *GDDM Base Application Programming Reference* book.

PL/I

In PL/I, it is necessary to declare each GDDM-GKS function used as an external entry; for example, when using the nonreentrant interface:

```
DECLARE GOPKS ENTRY (FIXED BINARY (31), FIXED BINARY (31))
              EXTERNAL OPTIONS (ASM,INTER);
```

or, when using the reentrant interface:

```
DECLARE GOPKS ENTRY (*,FIXED BINARY (31), FIXED BINARY (31))
              EXTERNAL OPTIONS (ASM,INTER);
```

or, when using the system programmer interface:

```
DECLARE ADMASP EXTERNAL ENTRY OPTIONS (ASM,INTER);
```

Note: No DECLARE statements are supplied for use with GDDM-GKS.

COBOL

The call format for COBOL is:

```
CALL callname [USING parameter-1 [parameter-2] ... ]
```

For example,

```
MOVE 1 TO ERRFILE.
MOVE 0 TO BUFFER.
CALL GOPKS USING ERRFILE BUFFER.
```

Assembler language

In assembler language, linkage is performed according to normal OS conventions:

1. Register 1 points to the address list containing the parameter addresses. The high-order bit must be “on” in the last word. If no parameters are to be passed, Register 1 must contain the value 0 or must point to a fullword of value 0 with the high-order bit “on.”
2. Register 13 points to a register save area of at least 72 bytes (18 fullwords).
3. Register 14 points to the return point in the application program.
4. Immediately before a call to GDDM-GKS, Register 15 points to the entry point with the name “callname.”
5. A branch to “callname” is performed.

In assembler language, linkage can be performed using the OS CALL macro, coded with the VL option. For example:

```
         CALL GOPKS,(ERRFIL,BUFFER),VL
         .......
ERRFIL   DC   F'1'
BUFFER   DC   F'0'
```

Note that calls with no parameters must be coded in the following form to ensure that Register 1 is set correctly:

```
CALL GCLKS,(0),VL
```

APL

GDDM supports VS APL and APL2. The appropriate *Terminal User's Guide* for the subsystem being used under VS APL and the *APL2 Programming: System Services Reference* manual (see "Books from related libraries" on page xiii) describe AP 126 (the GDDM auxiliary processor).

The APL code that corresponds to the name of a GDDM-GKS function call is given with the description of that function, in Chapter 5, "GDDM-GKS functions" on page 63. Also, all the APL codes are listed in Appendix G, "GDDM-GKS APL codes" on page 427. The APL codes for GDDM Base and GDDM-PGF calls are listed in the *GDDM Base Application Programming Reference* and *GDDM-PGF Programming Reference* books, respectively.

The GDDM auxiliary processor, AP 126, manages requests from an APL program. To use it you must follow this procedure:

1. Offer to share a variable with AP 126 to be used as the control variable; the control variable name must begin with CTL.
2. Check the degree of coupling returned and set access control.
3. Offer to share another variable with AP 126 to be used as the data variable; the data variable name must be the same as the control variable name except that the data variable name must begin with DAT.
4. Check the degree of coupling returned and set access control.
5. Use a specification statement to assign to the data variable any character data needed for the next request.
6. Assign to the control variable the request code for the next request together with any required parameters.
7. Ensure that the request completed successfully by referencing the control variable; 0 0 is returned if the request was successful.
8. Retrieve any character data returned by the request from the data variable.

An example of using AP 126 for GDDM calls in general is given in the *GDDM Base Application Programming Guide*.

IBM BASIC

A call interface to GDDM is provided by IBM BASIC. The first argument to the CALL GDDM command is the RCP code; for example:

```
100  GOPKS=939524096
110  CALL GDDM (GOPKS,1,0)
```

More information on using IBM BASIC with GDDM is given in the publications for IBM BASIC.

Mixing GKS functions and other GDDM functions

The following sections give information you need if you want to mix GDDM-GKS and other GDDM functions.

Mixing with graphical GDDM functions

GKS API calls should not be mixed with GDDM Base graphics, or GDDM-PGF calls to any page in use by GKS (see the section "GDDM hierarchy"). Although calls to these functions are not inhibited, the results of the calls are undefined and no guarantee is made to support applications that use them when mixed with GKS calls. This restriction applies to:

- All GDDM Base GSxxx calls
- All GDDM-PGF calls (CHxxx and CSxxx)

Note: GKS defines a pattern set for each device in use. If patterns are drawn, using non-GKS functions, on pages not in use by GKS, the expected patterns may not be produced.

Mixing with non-graphical GDDM functions

Image and alphanumerics calls can be mixed with GKS calls, but note that GKS construes that output is to a single page at each device. GKS input operations at a workstation cause the return of alphanumeric input data as well as graphics input.

Implicit FSINIT and FSTERM

An application accessing only the GDDM-GKS facilities and using the nonreentrant API, need not issue calls to the FSINIT and FSTERM functions, because these are performed implicitly when the Open GKS (GOPKS) and Close GKS (GCLKS) functions are invoked.

Applications that need to invoke the GKS functions allowed before GKS is opened must call FSINIT before invoking any GKS functions; they must then call FSTERM after GKS has been closed.

Applications using non-GKS facilities before GKS is opened, or after GKS is closed, or using the reentrant or SPI interfaces, must invoke FSINIT to initialize GDDM, and FSTERM after GKS has been closed.

GDDM hierarchy

When a device is opened as a GKS workstation, GKS construes the usable display surface of the device to be that defined by the current GDDM graphics hierarchy at the device (partition set, partition, page, and graphics field). All levels of the hierarchy not explicitly or implicitly set are defaulted in the normal way. GKS construes that all output to the display will be to the current page.

Applications may not use GKS functions to display drawings on more than one page or partition at a device. Pages and partitions can be selected using non-GKS functions, but should be restored, by the application, to those in use by GKS before GKS functions are invoked.

Devices may be opened by the GDDM Base DSOPEN call (either explicitly or implicitly according to GDDM rules) and be opened later as GKS workstations.

GKS ensures that the primary and alternate devices current before the GKS call are also current after the GKS function has completed.

Error handling

When mixing GDDM-GKS function calls and other GDDM calls, any errors occurring are dealt with by the error-handling facilities provided for each type of call. The GDDM external default ERRTHRS must not be set to any value other than the IBM-provided default; if this default is changed, results are unpredictable.

Using GDDM-GKS in a windowing environment

If you are using GDDM-GKS applications in a windowing environment such as ISPF, you must take precautions to ensure that applications running concurrently do not use the same identifiers for GKS error log and metafile output files. If two applications use the same identifiers, the results are unpredictable.

Using GDDM-GKS workstations

The following types of workstations can be used with GDDM-GKS:

- Physical devices
- Workstation independent segment storage (WISS)
- Metafiles
- GDF files.

In each case, connection to a particular workstation is established by the function Open workstation (GOPWK), which associates the workstation identifier with a workstation type and a connection identifier. You can obtain information about the class and characteristics of each workstation type by using the workstation description table Inquiry functions.

Physical devices

You can open and use a physical device such as a terminal, printer, or plotter by issuing a call to Open workstation (GOPWK) with the workstation type parameter set to one of the following values:

1, 6, 7, 8, 9, 10, 11, 12, or 13.

GDDM-GKS supports all of the IBM family-1 and family-2 devices supported by GDDM Base for graphics.

Workstation type 1

If workstation type 1 is specified, a default workstation, normally the user console, is opened. Any unused connection identifier can be given.

The workstation opened is the GDDM default primary device unless a current GDDM primary device already exists; in this case, processing is described in the section “How workstations are opened” on page 55.

Workstation types 6 through 13

To use workstation types 6 through 13, you must define the kinds of devices with which these workstation types are associated, and the connection to an appropriate physical device. This is done by setting the GDDM external default values and nicknames described below. GDDM provides an integrated method by which these user default specifications (UDS) can be set.

If the application is to run under the VM/CMS or MVS subsystems you can provide the definitions and nicknames in a GDDM external defaults file, which is interpreted when GKS is opened. Alternatively, you can use a GDDM external defaults module or the GDDM calls ESSUDS or ESEUDS, before GKS is opened. Finally, users of the system programmer interface can pass the required values by invoking the GDDM call SPINIT to initialize GDDM.

The GDDM UDS facilities and the use of external defaults files, modules, nicknames, and related calls are described in detail in the *GDDM Base Application Programming Reference* book.

Defining workstation types: Workstation types 6 through 13 are defined by supplying a UDS for the default value GKSWS in either source or encoded format.

The source format UDS for the GKSWS value is:

```
[label] ADMMDFT GKSWS=(devtok-list) [optional comments]
```

where

`label` Optional (ignored - it is not part of the UDS)

`GKSWS=(devtok-list)`
A list of up to 8 GDDM device tokens. The list takes the form

`dt1,dt2,...,dt8`

where `dt1` through `dt8` are device tokens indicating the devices corresponding to workstation types 6 through 13 respectively. No defaults are set initially.

A GDDM device token is a code, up to 8 characters long, giving entry to a table of pre-established device hardware characteristics. The table entry is used by GDDM-GKS to define the workstation description table used for the workstation type.

You must specify only the device tokens of family-1 and family-2 devices when defining workstation types; that is, the device tokens specified must have been generated as part of the GDDM table ADMLSYS1.

More information on device tokens and a list of GDDM-supplied device tokens for family-1 and family-2 devices can be found in the *GDDM Base Application Programming Reference* book.

Connecting workstations: When a workstation type of 6 through 13 is specified in the Open workstation (GOPWK) call, the connection identifier parameter is used to access a particular physical device. You must identify the device by providing a

GDDM nickname that has `FAM` (family) `1` and `NAME GKWSnnnn`, where `nnnn` is the value specified for the connection identifier parameter.

Nicknames are described in detail in the *GDDM Base Application Programming Reference* book. You can specify the device family (TOFAM), address, and other information for device use. If the nickname contains a device token entry then the device token specified must be appropriate for the device and must be the same as the one given to define the workstation type.

The following sections give examples for some devices.

3270-PC/G and /GX ranges, 3179-G, 3278, and 3279

GDDM supplies a number of device tokens for locally-attached 3270-PC/G and /GX, and 3270-PC AT/G and /GX workstations; also for locally-attached 3179-G displays; and for locally or remotely-attached 3278 and 3279 displays.

To use any of these devices as a workstation other than the user console, you must first define the workstation type using a suitable device token. For example, to define 3270-PC/G devices with 32 rows by 80 columns, and a mouse, as workstation type 6, include the following statement in your external defaults file:

```
ADMMDFT GKSWS=(L5279A1M)
```

Note: As described earlier, the first device token in the statement defines workstation type 6, the next defines workstation type 7, and so on.

The device information provided by the device token must match the characteristics of the physical device that you are going to use. For example, if you use device token L5279A1T, the tablet should be used with a stylus rather than with a tablet puck, because this device token only defines a tablet with a stylus.

Further examples are:

- 3270-PC/GX color devices (with tablet); include the following statement:

  ```
  ADMMDFT GKSWS=(L5379CST)
  ```

- Locally-attached 3279 displays; include the following statement:

  ```
  ADMMDFT GKSWS=(L79A3)
  ```

- 3179-G displays (without mouse); include the following statement:

  ```
  ADMMDFT GKSWS=(L3179G)
  ```

You also need a statement to specify the connection identifier associated with the physical address of the device. For example, if the device is dialed in to address 062, and you will be using connection identifier 7 when your application issues an Open workstation (GOPWK) call for the device, then the following statement must be included:

```
ADMMNICK FAM=1,NAME=GKWS0007,TONAME=(062)
```

5080

GDDM does not supply any device tokens giving the characteristics of IBM 5080 workstations. If you are using a 5080 device (and associated 3270 display), you are recommended to use it as your user console, where it can be opened as workstation type 1. If you cannot use it as your user console, you must provide a device token giving the device information and configuration. For information on

creating device tokens, refer to the *GDDM System Customization and Administration* manual.

If you plan to use a 5080, refer to the *GDDM Base Application Programming Guide* manual. and the *GDDM Base Application Programming Reference* manual for details of the processing options you will need to specify, and of the use of the 5080 and 3270 keyboards.

Plotters

GDDM supplies device tokens for plotters attached to 3179-G displays and to the 3270-PC/G and /GX ranges of workstations. To use a plotter attached to these devices you must first define the workstation type using a suitable device token. For example, to define 7374 plotters attached to any 3270-PC/G or /GX work stations as workstation type 6, include the following statement in your external defaults file:

```
ADMMDFT GKSWS=(L7374)
```

You also need a statement to specify the connection identifier associated with the physical address of the device. For example, if a 7374 plotter is attached to a 3270-PC/G being used as the user console, and you will be using connection identifier 7 when your application issues an Open workstation (GOPWK) call for the plotter, the following statement must be included:

```
ADMMNICK FAM=1,NAME=GKWS0007,TONAME=(*,ADMPLOT)
```

If you have more than one plotter attached to a member of the 3270-PC/G or /GX ranges of workstations, you can specify a plotter's name instead of ADMPLOT; the plotters are given names when GCP is configured. If ADMPLOT is specified, the first plotter is used.

If the plotter is attached to a dialed-in display or workstation, the device address can be specified. For example, to use a 7372 plotter attached to a 3179-G display dialed in to address 062, you could use the following statements in your external defaults file:

```
ADMMDFT GKSWS=(L3179G72)
ADMMNICK FAM=1,NAME=GKWS0007,TONAME=(062,ADMPLOT)
```

The plotter can then be used by invoking the Open workstation (GOPWK) function specifying the connection identifier parameter as 7 and workstation type 6.

Printers

GDDM supplies device tokens for family-1 and family-2 printers. To use a printer you must first define the workstation type using a suitable device token. For example, to define 3287 printers as workstation type 6, include the following statement in your external defaults file:

```
ADMMDFT GKSWS=(L87)
```

You also need a statement to specify the connection identifier associated with the physical address of the device. For example, if you will be using connection identifier 7 when your application issues an Open workstation (GOPWK) call for the device, the following statement must be included:

```
ADMMNICK FAM=1,NAME=GKWS0007,DEVTOK=L87,TOFAM=2,TONAME=(name)
```

where the value given for *name* depends on your subsystem:

- For VM/CMS, *name* is PUNCH or the filename of the print file to be generated. More information is given in the *GDDM Base Application Programming Reference* book.
- For TSO, *name* is the device identifier of your printer and must be one of the names in the master print queue dataset of the GDDM print utility. More information is given in the *GDDM Base Application Programming Reference* book.

Workstation independent segment storage

Workstation independent segment storage (WISS) is opened by specifying workstation type 2 in the Open workstation (GOPWK) call. Only a single WISS is permitted. The connection identifier parameter is ignored.

GKS metafiles

Metafile output (MO) workstations are opened by specifying workstation type 3 in the Open workstation (GOPWK) call.

Metafile input (MI) workstations are opened by specifying workstation type 4 in the Open workstation (GOPWK) call.

The connection identifier parameter must be an integer that identifies the file to be used. The mapping by GKS of the integer to the file to be used is subsystem-dependent:

- For VM/CMS systems the file accessed is:

  ADMJnnnn ADMGKSM A

  where nnnn represents the connection identifier specified.
- For other subsystems appropriate disk storage facilities and naming conventions are used; see “Using GDDM-GKS under various subsystems” on page 60 for details.

GDDM-GKS metafile output files are created as GDDM objects, and are intended to be read and written only by GDDM-GKS and GDDM utilities. The items stored in GDDM-GKS metafiles can be retrieved by an application using the functions Get item type from GKSM (GGTITM) and Read item from GKSM (GRDITM). The structure of the metafile items returned is described in Appendix C, “Metafile structure” on page 383.

When GKS is used in a windowing environment (for example, under ISPF) the usual precautions should be taken to ensure that file identification clashes do not occur.

GDF files

GDDM-GKS allows graphics data to be output for storage as GDDM floating-point GDF orders in GDDM ADMGDF objects. GDF-file workstations are GKS output workstations. Segments associated with the workstation are saved in an ADMGDF object when the workstation is closed.

Notes:

1. Only the final picture at the workstation is saved; intermediate states are not displayed or saved.
2. Pictures saved in this way cannot be retrieved by GDDM-GKS.

GDF-file workstations are opened by specifying workstation type 5 in the Open workstation (GOPWK) call. The connection identifier parameter must be an integer that identifies the file to be used. The mapping by GKS of the integer to the file to be used is subsystem-dependent:

- For VM/CMS systems the file accessed is:

  ADMJnnnn ADMGDF A

  where nnnn represents the connection identifier specified.
- For other subsystems, appropriate disk storage facilities and naming conventions are used; see “Using GDDM-GKS under various subsystems” on page 60 for details.

When GKS is used in a windowing environment (for example, under ISPF) the usual precautions should be taken to ensure that file identification clashes do not occur.

How workstations are opened

When an Open workstation (GOPWK) function is invoked to open a graphics device, GDDM-GKS performs the following actions:

- If the workstation type parameter is 1:
  - If no current primary device exists:
    - If the connection identifier is equal to the device-id of a device opened by a non-GKS GDDM call, that device is used.
    - If no device has been opened by a DSOPEN call for a device with a device-id equal to the connection identifier, the user console is opened using a GDDM Base DSOPEN call as follows:
      1. Device-id=connection-id
      2. Family=1
      3. Device-token=‘⋆’
      4. Procopt-count=0
      5. Procopt-list=null
      6. Name-count=0
      7. Name-list=null
  - If a current primary device already exists (opened by a non-GKS GDDM call), that device is used (the connection identifier is ignored).
- If the workstation type parameter ≥ 6:
  - If a device has been opened with a device-id equal to the connection identifier given in the GKS call, GDDM-GKS uses that device as the new workstation.

  - If no device has been opened by a DSOPEN call for a device with a device-id equal to the connection identifier, GDDM-GKS opens a device using a DSOPEN call having the following parameters:

    1. Device-id=connection-id.
    2. Family=1
    3. Device-token= '*'
    4. Procopt-count=0
    5. Procopt-list=null
    6. Name-count=1
    7. Name-list='GKWSnnnn'

    Further device selection information is then obtained from the nickname definition GKWSnnnn, according to the usual GDDM rules. Refer to the *GDDM Base Application Programming Reference* book for details.

- When a GDF-file workstation is opened, GDDM-GKS opens a dummy device using the highest unused device-id available.
- If the device indicated in the Open workstation (GOPWK) call has already been opened by a previous Open workstation (GOPWK) or DSOPEN call, but with another connection identifier or device-id, an error is returned.

If a device for which an Open workstation (GOPWK) call has been performed is closed by a DSCLS call before a Close workstation (GCLWK) call for the device is made, output to or input from the device returns an error.

When a workstation is closed using a Close workstation (GCLWK) call, and the device was first opened by use of an explicit or implicit DSOPEN call, the device remains available for access by GDDM Base functions but not by GKS functions. If the device was first opened by an Open workstation (GOPWK) call, GDDM-GKS issues a DSCLS call for the device.

If you use both GDDM and GDDM-GKS calls to access devices, you must ensure that you do not issue a DSCLS call for a workstation, or change the graphics hierarchy at the device, while it is open as a GDDM-GKS workstation.

In all cases, the GDDM hierarchy above the graphics field (device, partition set, partition, page), as set by GDDM Base calls, remains unchanged on return from calls to GKS functions.

Error handling in GDDM-GKS

This section discusses error handling in GDDM-GKS and describes how to replace the default GDDM-GKS error-handling procedure with an application-defined procedure.

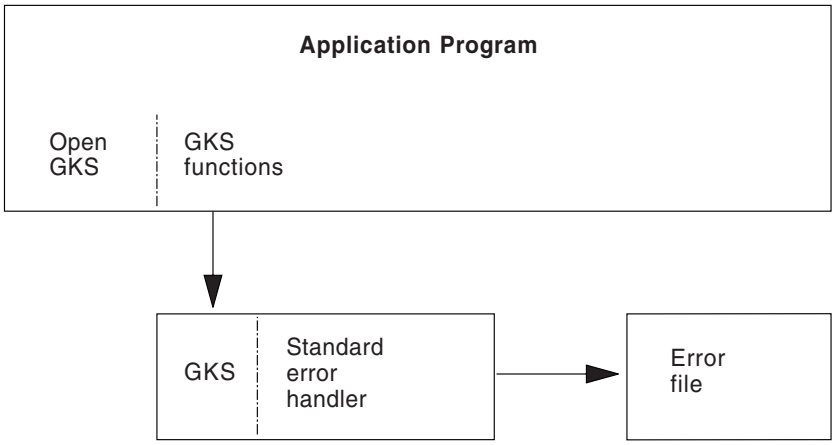
Standard GDDM-GKS error handling

When errors are detected during the execution of a GKS function the internal error-reporting functions invoke the standard error-handling function, Error handling (GERHND), which accepts the error definition as its parameters, and then invokes a standard error logger, Error logging (GERLOG), which formats an appropriate message and writes it to the error file.

The user (or an application programmer performing debugging) may display or print this error file using system facilities available on the user's subsystem.

When control is returned to GDDM-GKS from the error handler, internal error reaction routines perform any error recovery required. Normally, if a function causes an error, the function is not performed and has no effect; in some cases, clean-up operations are needed to get this result. However, some errors, detected in driver or operating system procedures called by GDDM-GKS, may be so severe that they cause GKS to try to save the results of previous operations by performing Emergency close GKS (GECLKS) processing.

The standard error handler is a procedure shipped as part of GDDM-GKS to capture and report errors, and is available for all languages and interfaces supported by GDDM-GKS.



To take advantage of the standard GKS error handler, you need only identify an error file as an argument of the Open GKS (GOPKS) function call. When an error is detected by GKS, this file will contain the messages described in the section "Error file information" on page 59.

Only a single error is reported during any call to a GDDM-GKS function. If errors are detected on a call to an inquiry function, the error handler is never called, because the error will be reported to the application in the error indicator output parameter required by each inquiry function.

It is not possible for an application to call the standard error-handling function directly.

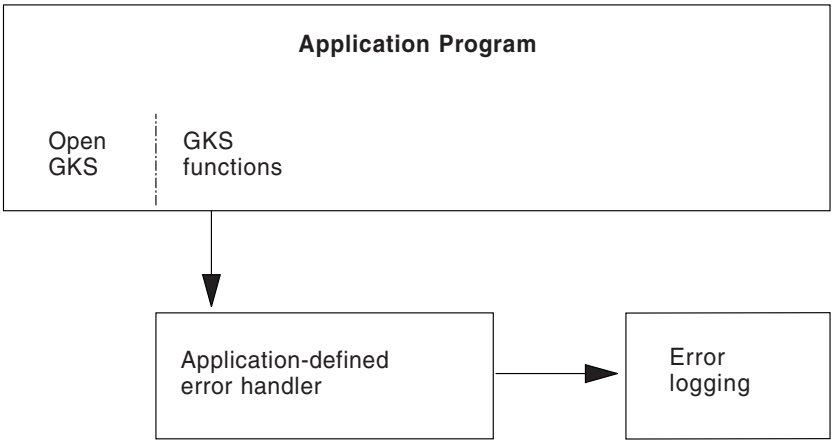
Application-defined error handling

If you are using the nonreentrant interface or the reentrant interface, you can replace the standard error handler with another that is customized to:

- Handle error conditions in a way more appropriate to the particular application
- Return error and GKS state information to the main body of the application in a data area shared with it
- Log error data, using a different format or displaying data on a medium such as an interactive display terminal
- Provide both a customized error handler and error logger.

If the application replaces standard error handling by providing its own error-handling procedure, and errors are detected during the execution of a GKS function, the internal error-reporting functions:

1. Set the GKS error state to “on.”
2. Call the application-supplied error-handling procedure.
3. Set the GKS error state to “off.”
4. Perform any built-in error action.



The application-defined error-handling procedure must:

- Be written in the same language as the application. If the application is a mixed-language program, the application-defined procedure must be in assembler language.
- Provide an entry point GERHND, which will be called by GKS functions when errors are detected.
- Accept the following parameters when it is invoked:

  Nonreentrant interface

  1. An integer giving the GKS error number for the error detected.
  2. An integer giving a number identifying the function in which the error was detected. These numbers are given in the lists of functions, in Chapter 5, “GDDM-GKS functions” on page 63.
  3. An integer giving the error file identification as specified in the Open GKS (GOPKS) function call.

For example, if your application is written in PL/I, you could use the following PROCEDURE statement:

```
GERHND: PROC (ERRNR,FCTID,ERRFIL) OPTIONS FORTRAN;
```

where `ERRNR`, `FCTID`, and `ERRFIL` are the three integer parameters.

Reentrant interface

1. The AAB (application anchor block) as defined in the main body of the application.
2. An integer giving the GKS error number for the error detected.
3. An integer giving a number identifying the function in which the error was detected. These numbers are given in the lists of functions, in Chapter 5, “GDDM-GKS functions” on page 63.
4. An integer giving the error file identification as specified in the Open GKS (GOPKS) function call.

- Return to GKS. Failure to return leaves GKS in the error state, thus preventing further execution.

If the application-supplied error-handling procedure has been invoked by GDDM-GKS, the GKS error state is “on” until the error state is set to “off” by returning to GDDM-GKS. While GKS is in error state the following limited set of GDDM-GKS functions can be invoked:

- All GKS inquiry functions

  This allows the error-handling procedure access to the set of GKS state variables.

- The standard Error logging (GERLOG) function

  This allows the GKS error information to be written to the error file specified in Open GKS (GOPKS).

- The Emergency close GKS (GECLKS) function

  This allows the application to close GKS when it cannot recover or continue executing because of the error detected. The Emergency close GKS (GECLKS) function attempts to save as much of the graphical information produced as possible.

Error file information

The Error logging (GERLOG) function causes error information to be written to the error file specified when GKS is opened. It is called by the standard error-handling procedure, and can be called by an application-supplied error-handling procedure if defined.

For each error logged by the Error logging (GERLOG) function, two messages are written to the error file. The first message is an indicator message (ADM3500) giving the GKS-defined error number and the name of the function in which it was detected. The indicator message is followed by a message giving a text description of the error and any supplementary information available. These messages have identifying numbers in the range ADM3501 through ADM3999 and are described in the *GDDM Messages* book.

The error message numbers do not correspond to the GKS error numbers, which are given in the indicator message, although in most cases the error message number can be obtained by adding 3500 to the error number. The exceptions are errors 2000 through 2003, for which the corresponding messages are 3900 through 3903, respectively.

The error file definition and characteristics are subsystem-dependent and are described in the following section.

Using GDDM-GKS under various subsystems

GDDM-GKS requires the installation of GDDM Base, which in turn may be used under several different subsystems. Some of the GKS facilities provided vary with the subsystem in use, and GDDM also imposes some restrictions or limitations when used on the different subsystems. This chapter contains information needed to use GDDM-GKS under the following subsystems:

- MVS, including use under MVS/XA
- VM/CMS.

Not all the information you need is presented here; only the differences from the information needed for using GDDM Base are given. Where the phrase “As for GDDM Base” is used, refer to the *GDDM Base Application Programming Reference* book for the required information.

MVS, including use under MVS/XA

GDDM-GKS will run under TSO, TSO/Batch, and MVS/Batch.

Programming languages

As for GDDM Base.

Data sets and file processing

As for GDDM Base, with the following additions:

- Metafile input/output

  GDDM-GKS metafiles are maintained as members of partitioned data sets. Basic partitioned access method (BPAM) is used to read and write metafiles on the partitioned data sets, and processing is as for GDDM Base. The data-set characteristics are:

  – GDDM file name (ddname) - ADMGKSM

  – Data set type - Partitioned

  – DCB characteristics:

  RECFM F, LRECL 400, BLKSIZE 400

  or

  RECFM FB, LRECL 400, BLKSIZE 400*n

  The member-name used when a metafile is created is ADMJnnnn, where nnnn is the connection identifier given in the Open workstation (GOPWK) function call.

- GKS error file

The Error logging (GERLOG) function always refers to the error file using a ddname. The error file parameter on the Open GKS (GOPKS) function call is ignored. The data-set characteristics are:

- GDDM file name (ddname) - ADMERLOG
- Data set type - Sequential or SYSOUT class
- DCB characteristics:

  RECFM VAR, LRECL 80 (or greater), BLKSIZE as LRECL

  or

  RECFM VAR, LRECL 80 (or greater), BLKSIZE = LRECL+4 or greater

Compiling programs
As for GDDM Base.

Link-editing programs
As for GDDM Base with the following changes:

One of the GDDM Base interface modules ADMASNT, ADMASRT, or ADMASPT must be explicitly included in the link-edit process. Inclusion by linkage editor automatic call facilities is not available.

Executing programs
As for GDDM Base.

VM/CMS

Programming languages
As for GDDM Base.

Data sets and file processing
As for GDDM Base, with the following additions:

- Metafile input/output

  GDDM-GKS metafiles are maintained as conventional CMS disk files. The data-set characteristics are:

  - GDDM filetype - ADMGKSM
  - RECFM F
  - LRECL 400

  The filename used when a metafile is created is ADMJnnnn, where nnnn is the connection identifier given in the Open workstation (GOPWK) function call.

- GKS error file

  The Error logging (GERLOG) function routes messages to a conventional CMS disk file. The data-set characteristics are:

  - GDDM filetype - ADMERLOG
  - RECFM V
  - LRECL 80

The filename used when an error file is created is ADMJnnnn, where nnnn is the error file identifier in the Open GKS (GOPKS) function call.

Compiling programs

As for GDDM Base.

Loading programs

As for GDDM Base.

Executing programs

As for GDDM Base.

Chapter 5. GDDM-GKS functions

This chapter consists of two listings of the GKS functions (ordered alphabetically by function and by function type), and the functions themselves.

For each function, the following information (if relevant) is given:

- The FORTRAN binding name and the textual name
- The syntax (keyword and parameters)
- The APL code
- The RCP code
- The function type, and a summary of the function provided
- Descriptions of the parameters
- A list of the operating states in which the function is valid
- A list of related functions
- A fuller description of the function, with any special information needed to use it
- Any GDDM-GKS restrictions
- A list of the GKS errors generated specifically by incorrect use of the function.

The following sections expand on this information.

Syntax conventions

The conventions used in presenting the syntax are:

- Uppercase words, parentheses, and commas must be coded exactly as shown.
- Lowercase words should be replaced by arguments appropriate to the programming language being used.

APL and RCP codes

Tables containing the APL codes for all functions are given in Appendix G, "GDDM-GKS APL codes" on page 427.

The GDDM request control parameter (RCP) code is given for each function in both hexadecimal and decimal format. This is to assist programmers who are using the system programmer interface to GDDM-GKS. Tables containing the RCP codes for all functions are given in Appendix F, "GDDM-GKS RCP codes" on page 419.

Function type and summary of function

First, the function type is given (control, input, and so on). This is followed by a one-sentence description of the function. If the function is subject to GDDM-GKS restrictions, a reference to the restriction is included here. Also, for those functions that have different versions for FORTRAN and for other languages (or for VS FORTRAN and FORTRAN IV), a note about the other version is added.

Parameter types

Some parameters must be assigned a value by you when a function is invoked; such parameters are indicated by appending *(specified by user)* to the parameter name. Parameters that receive information from GDDM-GKS are indicated by appending *(returned by GDDM-GKS)* to the parameter name.

© Copyright IBM Corp. 1984, 1987

Valid parameter values

Some parameter descriptions contain an indication of the values that you are permitted to supply, or that GDDM-GKS may return. When a range of values is given, the **ellipsis** symbol (...) is used to indicate that any of the omitted values are valid. If the range of values depends on factors that cannot be predetermined, the value n is used to represent the extreme value in the range. For example, the entry “The number of predefined color indexes (2 ... n)” means that the valid values are from 2 through an undetermined number, which in this case is the actual number of predefined color indexes.

Data types required for parameters

The parameters used in the functions have the data types described below:

Numerical parameters

All numerical parameters must be fullword items. Unless specifically stated otherwise in the parameter descriptions, the following conventions apply:

Integer parameters

Attribute codes	Lengths and dimensions
Control codes	Option codes
Identification numbers	

Fullword integer parameters must be declared as follows:

FORTRAN	INTEGER*4
PL/I	FIXED BINARY(31)
COBOL	PICTURE S9(8) COMPUTATIONAL or equivalent
Assembler	F-constant

Single-precision floating-point parameters

Geometric character attributes	Marker positions
Line end-points	Viewport coordinates
Color table values	Window coordinates

Floating-point parameters must be declared as follows:

FORTRAN	REAL*4
PL/I	FLOAT DECIMAL(6)
COBOL	COMPUTATIONAL-1
Assembler	E-constant

The absolute values of nonzero floating-point parameters must be in the range 1.0E–18 through 1.0E18.

Character parameters

These must be supplied as the appropriate EBCDIC codes. They must be declared as follows:

FORTRAN	Use string literals or numeric data array initialized with string literals
PL/I	CHARACTER(n) /*Note: Must **not** be VARYING*/
COBOL	PICTURE X (n)
Assembler	C-constant

Arrays

These must be declared as follows:

FORTRAN	Use a one-dimensional array. (You can use a multidimensional array if it causes the correct storage mapping. Note that FORTRAN arrays are stored in column-major order).
PL/I	Use a one-dimensional array. (You can use a multidimensional array if it causes the correct storage mapping. Note that PL/I arrays are stored in row-major order).
COBOL	Use the OCCURS clause.
Assembler	Use contiguous fullwords.

Enumeration type parameters

These are parameters representing one value from a set of values. All enumeration types in GDDM-GKS are mapped to integers. The integer corresponding to each value in the set is given in the parameter description.

In your program, you can either code these integers directly, or declare variables initialized to the appropriate integer values. The variables can then be used when passing input parameters or to test values returned in output parameters. The parameter description includes suggested mnemonic names for variables corresponding to the values in the set.

Appendix B, “GDDM-GKS enumeration types” on page 379 contains a list of all GKS enumeration types and associated integer values.

Related functions

If any functions are listed under this heading, refer to their descriptions in this chapter; they may contain information that is useful to you in coding the function.

GDDM-GKS restrictions

All of the GKS standard functions are described in this chapter. However, not all GKS functions are implemented in GDDM-GKS (because it is not at the highest GKS level), and some functions have slight differences from the GKS standard; such restrictions are documented after the full description for each affected function.

Errors and error messages

For each GKS function, a list is given of the principal GKS errors that can be reported as a result of incorrect use of the function. Each error has the number, and the actual text, as it appears in the GKS standard.

In addition to the errors listed for each function, the following errors can occur:

```
300  Storage overflow has occurred in GKS
301  Storage overflow has occurred in segment storage
302  Input/Output error has occurred while reading
303  Input/Output error has occurred while writing
304  Input/Output error has occurred while sending data to a
     workstation
305  Input/Output error has occurred while receiving data from a
     workstation
306  Input/Output error has occurred during program library
     management
307  Input/Output error has occurred while reading workstation
     description table
308  Arithmetic error has occurred
```

If the standard GDDM-GKS Error logging (GERLOG) function is used, each of the GKS errors causes two messages to be written to the error file specified when GKS is opened. The first message is always ADM3500, and it contains the GKS error number, together with the name of the function in which it was detected. The second message gives a text description of the error; the numbers of these messages are in the range ADM3501 through ADM3999, and bear the following relationship to the GKS error number:

- For GKS error numbers in the range 1 through 308, the GDDM-GKS message number is calculated by adding 3500 to the GKS error number.
- For GKS error numbers in the range 2000 through 2003, the GDDM-GKS message number is calculated by adding 1900 to the GKS error number.

The GDDM-GKS messages are listed and further explained in the *GDDM Messages* book.

GKS function numbers

In the following lists, the “Number” column contains the GKS function number. This number is used by the default Error handling (GERHND) function. You may want to use these numbers in a user-defined error-handling procedure. Refer to “Error handling in GDDM-GKS” on page 56 for details on providing such a procedure.

Functions that do not report errors, or that report errors in an error indicator parameter, do not have function numbers.

Alphabetical list of functions

Function	Name	Number	Page
Accumulate transformation matrix	GACTM	106	78
Activate workstation	GACWK	4	80
Associate segment with workstation	GASGWK	61	81
Await event	GWAIT	93	356
Cell array	GCA	16	82
Clear workstation	GCLRWK	6	85
Close GKS	GCLKS	1	85
Close segment	GCLSG	57	87
Close workstation	GCLWK	3	87
Copy segment to workstation	GCSGWK	62	90
Create segment	GCRSG	56	89
Deactivate workstation	GDAWK	5	91
Delete segment	GDSG	59	92
Delete segment from workstation	GDSGWK	60	93
Emergency close GKS	GECLKS		94
Error handling	GERHND		95
Error logging	GERLOG		96
Escape	GESC	11	97
Evaluate transformation matrix	GEVTM	105	98
Fill area	GFA	15	100
Flush device events	GFLUSH	94	102
Generalized drawing primitive	GGDP	17	103
Get choice	GGTCH	98	105
Get item type from GKSM	GGTITM	102	106
Get locator	GGTLC	95	107
Get pick	GGTPK	99	108
Get string (FORTRAN only)	GGTST	100	110
Get string	GGTSTS	100	111
Get stroke	GGTSK	96	109
Get valuator	GGTVL	97	112
Initialize choice	GINCH	72	114
Initialize locator	GINLC	69	116
Initialize pick	GINPK	73	120
Initialize string (FORTRAN only)	GINST	74	127
Initialize string	GINSTS	74	129
Initialize stroke	GINSK	70	123
Initialize valuator	GINVL	71	132
Inquire aspect source flags	GQASF		144
Inquire character base vector	GQCHB		147
Inquire character expansion factor	GQCHXP		153
Inquire character height	GQCHH		148
Inquire character spacing	GQCHSP		150
Inquire character up vector	GQCHUP		151
Inquire character width	GQCHW		152

Function	Name	Number	Page
Inquire pixel array dimensions	GQPXAD		228
Inquire polyline color index	GQPLCI		212
Inquire polyline facilities	GQPLF		213
Inquire polyline index	GQPLI		214
Inquire polyline representation	GQPLR		215
Inquire polymarker color index	GQPMCI		216
Inquire polymarker facilities	GQPMF		217
Inquire polymarker index	GQPMI		219
Inquire polymarker representation	GQPMR		219
Inquire predefined color representation	GQPCR		207
Inquire predefined fill area representation	GQPFAR		208
Inquire predefined pattern representation	GQPPAR		221
Inquire predefined polyline representation	GQPPLR		222
Inquire predefined polymarker representation	GQPPMR		223
Inquire predefined text representation	GQPTXR		224
Inquire segment attributes	GQSGA		229
Inquire set member of active workstations	GQACWK		143
Inquire set member of associated workstations	GQASWK		145
Inquire set member of open workstations	GQOPWK		202
Inquire set member of segment names in use	GQSGUS		232
Inquire set member of segment names on workstation	GQSGWK		233
Inquire string device state (FORTRAN only)	GQSTS		236
Inquire string device state	GQSTSS		238
Inquire stroke device state	GQSKS		235
Inquire text alignment	GQTXAL		240
Inquire text color index	GQTXCI		241
Inquire text extent (VS FORTRAN only)	GQTXX		247
Inquire text extent	GQTXXS		250
Inquire text facilities	GQTXF		241
Inquire text font and precision	GQTXFP		243
Inquire text index	GQTXI		244
Inquire text path	GQTXP		245
Inquire text representation	GQTXR		246
Inquire valuator device state	GQVLS		252
Inquire workstation category	GQWKCA		255
Inquire workstation classification	GQWKCL		256
Inquire workstation connection and type	GQWKC		254
Inquire workstation deferral and update states	GQWKDU		257
Inquire workstation maximum numbers	GQWKM		258
Inquire workstation state	GQWKS		259
Inquire workstation transformation	GQWKT		260
Insert segment	GINSG	63	122
Interpret item	GIITM	104	112
Message (VS FORTRAN only)	GMSG	10	133
Message	GMSGS	10	134
Open GKS	GOPKS	0	135
Open workstation	GOPWK	2	136
Pack data record (FORTRAN only)	GPREC	107	140
Pack data record	GPRECS	107	141
Polyline	GPL	12	138
Polymarker	GPM	13	139
Read item from GKSM	GRDITM	103	261
Redraw all segments on workstation	GRSGWK	7	274
Rename segment	GRENSG	58	262
Request choice	GRQCH	84	263
Request locator	GRQLC	81	265
Request pick	GRQPK	85	267

Function	Name	Number	Page
Request string (FORTRAN only)	GRQST	86	270
Request string	GRQSTS	86	271
Request stroke	GRQSK	82	268
Request valuator	GRQVL	83	272
Sample choice	GSMCH	90	299
Sample locator	GSMLC	87	304
Sample pick	GSMPK	91	305
Sample string (FORTRAN only)	GSMST	92	307
Sample string	GSMSTS	92	308
Sample stroke	GSMSK	88	306
Sample valuator	GSMVL	89	310
Select normalization transformation	GSELNT	52	287
Set aspect source flags	GSASF	41	274
Set character expansion factor	GSCHXP	28	280
Set character height	GSCHH	31	276
Set character spacing	GSCHSP	29	278
Set character up vector	GSCHUP	32	279
Set choice mode	GSCHM	78	277
Set clipping indicator	GSCLIP	53	281
Set color representation	GSCR	48	282
Set deferral state	GSDS	9	284
Set detectability	GSDTEC	68	286
Set fill area color index	GSFACI	38	288
Set fill area index	GSFAI	35	289
Set fill area interior style	GSFAIS	36	290
Set fill area representation	GSFAR	46	291
Set fill area style index	GSFASI	37	293
Set highlighting	GSHLIT	66	295
Set linetype	GSLN	19	297
Set linewidth scale factor	GSLWSC	20	298
Set locator mode	GSLCM	75	296
Set marker size scale factor	GSMKSC	24	302
Set marker type	GSMK	23	301
Set pattern reference point	GSPARF	40	314
Set pattern representation	GSPAR	47	312
Set pattern size	GSPA	39	311
Set pick identifier	GSPKID	42	315
Set pick mode	GSPKM	79	316
Set polyline color index	GSPLCI	21	317
Set polyline index	GSPLI	18	318
Set polyline representation	GSPLR	43	319
Set polymarker color index	GSPMCI	25	321
Set polymarker index	GSPMI	22	322
Set polymarker representation	GSPMR	44	323
Set segment priority	GSSGP	67	325
Set segment transformation	GSSGT	64	327
Set string mode	GSSTM	80	329
Set stroke mode	GSSKM	76	328
Set text alignment	GSTXAL	34	331
Set text color index	GSTXCI	30	333
Set text font and precision	GSTXFP	27	334
Set text index	GSTXI	26	337
Set text path	GSTXP	33	338
Set text representation	GSTXR	45	339
Set valuator mode	GSVLM	77	342
Set viewport	GSVP	50	343
Set viewport input priority	GSVPIP	51	344

Function	Name	Number	Page
Set visibility	GSVIS	65	341
Set window	GSWN	49	348
Set workstation viewport	GSWKVP	55	345
Set workstation window	GSWKWN	54	347
Text (VS FORTRAN only)	GTX	14	349
Text	GTXS	14	351
Unpack data record (FORTRAN only)	GUREC	108	352
Unpack data record	GURECS	108	353
Update workstation	GUWK	8	355
Write item to GKSM	GWITM	101	358

List of functions by function type

Control functions

Name	Function	Number	Page
GOPKS	Open GKS	0	135
GCLKS	Close GKS	1	85
GOPWK	Open workstation	2	136
GCLWK	Close workstation	3	87
GACWK	Activate workstation	4	80
GDAWK	Deactivate workstation	5	91
GCLRWK	Clear workstation	6	85
GRSGWK	Redraw all segments on workstation	7	274
GUWK	Update workstation	8	355
GSDS	Set deferral state	9	284
GMSG	Message (VS FORTRAN only)	10	133
GMSGS	Message	10	134
GESC	Escape	11	97

Output functions

Name	Function	Number	Page
GPL	Polyline	12	138
GPM	Polymarker	13	139
GTX	Text (VS FORTRAN only)	14	349
GTXS	Text	14	351
GFA	Fill area	15	100
GCA	Cell array	16	82
GGDP	Generalized drawing primitive	17	103

Output attributes

Workstation-independent primitive attributes

Name	Function	Number	Page
GSPLI	Set polyline index	18	318
GSLN	Set linetype	19	297
GSLWSC	Set linewidth scale factor	20	298
GSPLCI	Set polyline color index	21	317
GSPMI	Set polymarker index	22	322

Name	Function	Number	Page
GSMK	Set marker type	23	301
GSMKSC	Set marker size scale factor	24	302
GSPMCI	Set polymarker color index	25	321
GSTXI	Set text index	26	337
GSTXFP	Set text font and precision	27	334
GSCHXP	Set character expansion factor	28	280
GSCHSP	Set character spacing	29	278
GSTXCI	Set text color index	30	333
GSCHH	Set character height	31	276
GSCHUP	Set character up vector	32	279
GSTXP	Set text path	33	338
GSTXAL	Set text alignment	34	331
GSFAI	Set fill area index	35	289
GSFAIS	Set fill area interior style	36	290
GSFASI	Set fill area style index	37	293
GSFACI	Set fill area color index	38	288
GSPA	Set pattern size	39	311
GSPARF	Set pattern reference point	40	314
GSASF	Set aspect source flags	41	274
GSPKID	Set pick identifier	42	315

Workstation attributes (representations)

Name	Function	Number	Page
GSPLR	Set polyline representation	43	319
GSPMR	Set polymarker representation	44	323
GSTXR	Set text representation	45	339
GSFAR	Set fill area representation	46	291
GSPAR	Set pattern representation	47	312
GSCR	Set color representation	48	282

Transformation functions

Normalization transformation

Name	Function	Number	Page
GSWN	Set window	49	348
GSVP	Set viewport	50	343
GSVPIP	Set viewport input priority	51	344
GSELNT	Select normalization transformation	52	287
GSCLIP	Set clipping indicator	53	281

Workstation transformation

Name	Function	Number	Page
GSWKWN	Set workstation window	54	347
GSWKVP	Set workstation viewport	55	345

Segment functions

Segment manipulation functions

Name	Function	Number	Page
GCRSG	Create segment	56	89
GCLSG	Close segment	57	87
GRENSG	Rename segment	58	262
GDSG	Delete segment	59	92
GDSGWK	Delete segment from workstation	60	93
GASGWK	Associate segment with workstation	61	81
GCSGWK	Copy segment to workstation	62	90
GINSG	Insert segment	63	122

Segment attributes

Name	Function	Number	Page
GSSGT	Set segment transformation	64	327
GSVIS	Set visibility	65	341
GSHLIT	Set highlighting	66	295
GSSGP	Set segment priority	67	325
GSDTEC	Set detectability	68	286

Input functions

Initialization of input devices

Name	Function	Number	Page
GINLC	Initialize locator	69	116
GINSK	Initialize stroke	70	123
GINVL	Initialize valuator	71	132
GINCH	Initialize choice	72	114
GINPK	Initialize pick	73	120
GINST	Initialize string (FORTRAN only)	74	127
GINSTS	Initialize string	74	129

Setting the mode of input devices

Name	Function	Number	Page
GSLCM	Set locator mode	75	296
GSSKM	Set stroke mode	76	328
GSVLM	Set valuator mode	77	342
GSCHM	Set choice mode	78	277
GSPKM	Set pick mode	79	316
GSSTM	Set string mode	80	329

Request input functions

Name	Function	Number	Page
GRQLC	Request locator	81	265
GRQSK	Request stroke	82	268
GRQVL	Request valuator	83	272
GRQCH	Request choice	84	263
GRQPK	Request pick	85	267
GRQST	Request string (FORTRAN only)	86	270
GRQSTS	Request string	86	271

Sample input functions

Name	Function	Number	Page
GSMLC	Sample locator	87	304
GSMSK	Sample stroke	88	306
GSMVL	Sample valuator	89	310
GSMCH	Sample choice	90	299
GSMPK	Sample pick	91	305
GSMST	Sample string (FORTRAN only)	92	307
GSMSTS	Sample string	92	308

Event input functions

Name	Function	Number	Page
GWAIT	Await event	93	356
GFLUSH	Flush device events	94	102
GGTLC	Get locator	95	107
GGTSK	Get stroke	96	109
GGTVL	Get valuator	97	112
GGTCH	Get choice	98	105
GGTPK	Get pick	99	108
GGTST	Get string (FORTRAN only)	100	110
GGTSTS	Get string	100	111

Metafile functions

Name	Function	Number	Page
GWITM	Write item to GKSM	101	358
GGTITM	Get item type from GKSM	102	106
GRDITM	Read item from GKSM	103	261
GIITM	Interpret item	104	112

Utility functions

Name	Function	Number	Page
GEVTM	Evaluate transformation matrix	105	98
GACTM	Accumulate transformation matrix	106	78
GPREC	Pack data record (FORTRAN only)	107	140
GPRECS	Pack data record	107	141
GUREC	Unpack data record (FORTRAN only)	108	352
GURECS	Unpack data record	108	353

Inquiry functions

Inquiry function for operating state value

Name	Function	Page
GQOPS	Inquire operating state value	201

Inquiry functions for GKS description table

Name	Function	Page
GQLVKS	Inquire level of GKS	194
GQEWK	Inquire list element of available workstation types	181
GQWKM	Inquire workstation maximum numbers	258
GQMNTN	Inquire maximum normalization transformation number	199

Inquiry functions for GKS state list

Name	Function	Page
GQOPWK	Inquire set member of open workstations	202
GQACWK	Inquire set member of active workstations	143
GQPLI	Inquire polyline index	214
GQPMI	Inquire polymarker index	219
GQTXI	Inquire text index	244
GQCHH	Inquire character height	148
GQCHUP	Inquire character up vector	151
GQCHW	Inquire character width	152
GQCHB	Inquire character base vector	147
GQTXP	Inquire text path	245
GQTXAL	Inquire text alignment	240
GQFAI	Inquire fill area index	184
GQPA	Inquire pattern size	203
GQPARF	Inquire pattern reference point	206
GQPKID	Inquire pick identifier	209
GQLN	Inquire linetype	193
GQLWSC	Inquire linewidth scale factor	196
GQPLCI	Inquire polyline color index	212
GQMK	Inquire marker type	197
GQMKSC	Inquire marker size scale factor	198
GQPMCI	Inquire polymarker color index	216
GQTXFP	Inquire text font and precision	243
GQCHXP	Inquire character expansion factor	153
GQCHSP	Inquire character spacing	150
GQTXCI	Inquire text color index	241
GQFAIS	Inquire fill area interior style	185
GQFASI	Inquire fill area style index	187
GQFACI	Inquire fill area color index	182
GQASF	Inquire aspect source flags	144
GQCNTN	Inquire current normalization transformation number	154
GQENTN	Inquire list element of normalization transformation numbers	175
GQNT	Inquire normalization transformation	200
GQCLIP	Inquire clipping indicator	154
GQOPSG	Inquire name of open segment	201
GQSGUS	Inquire set member of segment names in use	232
GQSIM	Inquire more simultaneous events	234

Inquiry functions for workstation state list

Name	Function	Page
GQWKC	Inquire workstation connection and type	254
GQWKS	Inquire workstation state	259
GQWKDU	Inquire workstation deferral and update states	257
GQEPLI	Inquire list element of polyline indexes	177
GQPLR	Inquire polyline representation	215
GQEPMI	Inquire list element of polymarker indexes	178
GQPMR	Inquire polymarker representation	219
GQETXI	Inquire list element of text indexes	179
GQTXR	Inquire text representation	246
GQTXX	Inquire text extent (VS FORTRAN only)	247
GQTXXS	Inquire text extent	250
GQEFAI	Inquire list element of fill area indexes	173
GQFAR	Inquire fill area representation	186
GQEPAI	Inquire list element of pattern indexes	176
GQPAR	Inquire pattern representation	205

Name	Function	Page
GQECI	Inquire list element of color indexes	172
GQCR	Inquire color representation	155
GQWKT	Inquire workstation transformation	260
GQSGWK	Inquire set member of segment names on workstation	233
GQLCS	Inquire locator device state	190
GQSKS	Inquire stroke device state	235
GQVLS	Inquire valuator device state	252
GQCHS	Inquire choice device state	149
GQPKS	Inquire pick device state	210
GQSTS	Inquire string device state (FORTRAN only)	236
GQSTSS	Inquire string device state	238

Inquiry functions for workstation description table

Name	Function	Page
GQWKCA	Inquire workstation category	255
GQWKCL	Inquire workstation classification	256
GQDSP	Inquire display space size	166
GQDWKA	Inquire dynamic modification of workstation attributes	170
GQDDS	Inquire default deferral state values	158
GQPLF	Inquire polyline facilities	213
GQPPLR	Inquire predefined polyline representation	222
GQPMF	Inquire polymarker facilities	217
GQPPMR	Inquire predefined polymarker representation	223
GQTXF	Inquire text facilities	241
GQPTXR	Inquire predefined text representation	224
GQFAF	Inquire fill area facilities	182
GQPFAR	Inquire predefined fill area representation	208
GQPAF	Inquire pattern facilities	204
GQPPAR	Inquire predefined pattern representation	221
GQCF	Inquire color facilities	146
GQPCR	Inquire predefined color representation	207
GQEGDP	Inquire list element of available generalized drawing primitives	174
GQGDP	Inquire generalized drawing primitive	188
GQLWK	Inquire maximum length of workstation state tables	195
GQSGP	Inquire number of segment priorities supported	230
GQDSGA	Inquire dynamic modification of segment attributes	163
GQLI	Inquire number of available logical input devices	192
GQDLC	Inquire default locator device data	160
GQDSK	Inquire default stroke device data	164
GQDVL	Inquire default valuator device data	168
GQDCH	Inquire default choice device data	157
GQDPK	Inquire default pick device data	161
GQDST	Inquire default string device data	167

Inquiry functions for segment state list

Name	Function	Page
GQASWK	Inquire set member of associated workstations	145
GQSGA	Inquire segment attributes	229

Pixel inquiries

Name	Function	Page
GQPXAD	Inquire pixel array dimensions	228
GQPXA	Inquire pixel array	227
GQPX	Inquire pixel	225

Inquiry function for GKS error state list

Name	Function	Page
GQIQOV	Inquire input queue overflow	189

Error-handling functions

Name	Function	Page
GECLKS	Emergency close GKS	94
GERHND	Error handling	95
GERLOG	Error logging	96

The GKS functions

The remainder of this chapter consists of the GKS function descriptions. They are presented in the order of the FORTRAN binding name.

GACTM

Purpose

GACTM	(minp, xo, yo, dx, dy, phi, fx, fy, sw, mout)
APL code	1418
GKS RCP code	X'38006A00' (939551232)

Function: To accumulate transformation matrix.

Utility function. Composes a new transformation matrix from a pre-existing matrix and new input values for rotation, shift, and scale.

Parameters

minp *(specified by user) (array of short floating-point numbers)*
An existing transformation matrix. You can use the Inquire segment attributes (GQSGA) function to obtain the values for this parameter.
xo *(specified by user) (short floating point)*
yo *(specified by user) (short floating point)*
A given point, in world coordinates or normalized device coordinates.

dx *(specified by user) (short floating point)*
dy *(specified by user) (short floating point)*
A shift vector in world or normalized device coordinates. The shift vector moves the segment from one place to another on the display screen.
phi *(specified by user) (short floating point)*
The rotation angle in radians; a positive value indicates a counterclockwise direction.
fx *(specified by user) (short floating point)*
fy *(specified by user) (short floating point)*
The scale factors. These factors control the size of the segment on the display surface.
sw *(specified by user) (fullword integer)*
The coordinate switch. The possible values are:
0 (GWC) World coordinates.
1 (GNDC) Normalized device coordinates.
mout *(returned by GDDM) (array of short floating-point numbers)*
A segment transformation matrix in world or normalized device coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Evaluate transformation matrix (GEVTM), Set segment transformation (GSSGT)

Description

This function composes a new segment transformation matrix from the input matrix and the transformation defined by the input values for shift, rotation, and scale. The input segment transformation matrix *minp* can be a matrix previously defined either by this function or by Evaluate transformation matrix (GEVTM). The resulting new matrix is used, for example, in the functions Set segment transformation (GSSGT) and Insert segment (GINSG).

The identity matrix is stored with each segment in the segment state list when it is created. When you call GACTM or GEVTM, and then Set segment transformation (GSSGT), the newly created matrix replaces the matrix in the segment state list, and it controls the appearance of the segment until a new matrix is set. The effects of repeated applications of a matrix are not cumulative. Therefore you "accumulate" a matrix in order to retain the effects of a previous matrix while making further changes in the appearance of a display.

For example, you can display a segment at half its original size by scaling it in a transformation matrix calculated by calling GEVTM. If you then want to shift the scaled segment, GACTM takes the first transformation and adds the shift vector to it in the new matrix. You can shift the segment again, or scale it, or rotate it, by calls to GACTM and Set segment transformation (GSSGT).

The coordinate switch (*sw*) governs whether the shift vector and fixed point are given world coordinate units or normalized device coordinate units. If you use WC units, the shift vector and fixed point are transformed by the current normalization transformation.

The shift vector is entered as the relative displacement between two points. For example, if the input fixed point is (10.0, 10.0), and the segment transformation shifts the display to (25.0, 25.0), the shift vector is entered as (15.0, 15.0).

You enter the rotation angle in radians. To convert an angle in degrees to radians, you use the following equation:

```
radians = pi/180 * degrees
```

For example, to rotate a display by 45 degrees, you enter 0.7853982 as the value for the rotation angle.

You enter identity values for those aspects that you do not want to change in any specific accumulation of matrixes. These are 0.0 (zero) for shift and rotation and 1.0 (one) for scale.

The segment transformation is stored as a matrix array with six elements. Point (*x*, *y*) is transformed into point (*x*', *y*') as follows:

The order of transformation is: scale, rotate, and shift. Scaling and rotation are performed relative to the input fixed point. Output values defining the shift components are in NDC units. The other elements have no units.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
2000	Enumeration type out of range

GACWK

Purpose

GACWK	**(wkid)**
APL code	1307
GKS RCP code	X'38000400' (939525120)

Function: To activate workstation.

Control function. Makes the workstation ready for GKS output.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.

Operating states

WSOP, WSAC

Related functions

Close workstation (GCLWK), Open workstation (GOPWK), Deactivate workstation (GDAWK)

Description

This function makes a workstation ready to receive output. The specified workstation is marked active in the workstation state list and is added to the list of active workstations in the GKS state list. If the GKS operating state is WSOP, the state is set to WSAC.

Output primitives are sent to, and segments are stored on, all active workstations and no others. *Input* and *MI* workstations cannot be activated.

Principal errors

```
6         GKS not in proper state: GKS shall be either in the state WSOP or
          in the state WSAC
20        Specified workstation identifier is invalid
25        Specified workstation is not open
29        Specified workstation is active
33        Specified workstation is of category MI
35        Specified workstation is of category INPUT
43        Maximum number of simultaneously active workstations would be
          exceeded
```

GASGWK

Purpose

GASGWK	(wkid, sgna)
APL code	1362
GKS RCP code	X'38003D00' (939539712)

Function: To associate segment with workstation.

Segment function. Sends a stored segment to a workstation in the same way as if the workstation were active when the segment was created.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.
sgna *(specified by user) (fullword integer)*
The identifier selected when the segment was created.

Operating states

WSOP, WSAC

Related functions

Create segment (GCRSG), Copy segment to workstation (GCSGWK), Delete segment from workstation (GDSGWK), Open workstation (GOPWK), Set viewport (GSVP)

Description

This function sends a segment from WISS (workstation independent segment storage) to the workstation *wkid*. The segment name is added to the list of segments at the workstation in the workstation state list. The workstation identifier is added to the list of associated workstations in the segment state list.

If the segment is not present in WISS, error 124 is reported.

If the segment is already associated with the workstation *wkid*, the function has no effect.

The clipping rectangles stored with the primitives in the segment are copied unchanged.

Principal errors

6	GKS not in proper state: GKS shall be either in the state WSOP or in the state WSAC
20	Specified workstation identifier is invalid
25	Specified workstation is not open
27	Workstation Independent Segment Storage is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
120	Specified segment name is invalid
124	Specified segment does not exist on Workstation Independent Segment Storage

---

GCA

Purpose

GCA	(px, py, qx, qy, dimx, dimy, isc, isr, dx, dy, colia)
APL code	1316
GKS RCP code	X'38001000' (939528192)

Function: To cell array.

Output function. Generates a *cell array* primitive.

Parameters

px *(specified by user) (short floating point)*
py *(specified by user) (short floating point)*
qx *(specified by user) (short floating point)*
qy *(specified by user) (short floating point)*
The corners of the cell rectangle in world coordinates.
dimx *(specified by user) (fullword integer)*
dimy *(specified by user) (fullword integer)*
The dimensions of the color index array, *colia*.
isc *(specified by user) (fullword integer)*
isr *(specified by user) (fullword integer)*
The indexes of the start column and the start row in *colia* of the cell array.
dx *(specified by user) (fullword integer)*
dy *(specified by user) (fullword integer)*
The number of columns and the number of rows of the cell array.
colia *(specified by user) (an array of fullword integers)*
The color index array (*dimx* groups of *dimy* elements).

Operating states

WSAC, SGOP

Related functions

None

Description

A cell array is a primitive generated by associating a color index array with a rectangle, called the **cell rectangle**, which is aligned with the world coordinate axes.

The cell rectangle is defined by two corners, P and Q, with world coordinates (*px*, *py*) and (*qx*, *qy*).

The rectangle is divided into a grid, *dx*×*dy*, of equal-sized cells. The width of each cell is ∨*px*–*qx*∨÷*dx*, and the height is ∨*py*–*qy*∨÷*dy*. (The symbol "∨" signifies that the absolute value is used; this is the value irrespective of the sign.)

To generate the cell array output primitive, the color of each cell is specified by the index of the associated element of the color index array *colia*. The color index array elements correspond to the cells in the cell rectangle as follows:

- Element (*isc*, *isr*) corresponds to the cell having P at one corner.
- Element (*isc*+*dx*–1, *isr*+*dy*–1) corresponds to the cell having Q at one corner.
- Element (*isc*, *isr*+*dy*–1) corresponds to the cell having point (*px*, *qy*) at one corner.
- Element (*isc*+*dx*–1, *isr*) corresponds to the cell having point (*qx*, *py*) at one corner.

The color index array can be mapped to the cell rectangle in any combination of directions along the x and y axes. Assuming that P is the top-left corner and Q is the bottom-right corner of the cell rectangle, a one-for-one mapping is performed. If P and Q are reversed, the color index array is mapped to a reflection of the cell rectangle, in both axes.

An entire color index array can be used by setting *isc*=1, *isr*=1, *dx*=*dimx*, and *dy*=*dimy*. Alternatively, a portion of the color index array can be used by setting *isc* and *isr* to indicate the start column and row in *colia*, and *dx* and *dy* to the number of columns and number of rows to be used.

If a color index is not present in the color table at a workstation, GDDM-GKS will substitute a workstation-dependent index at that workstation.

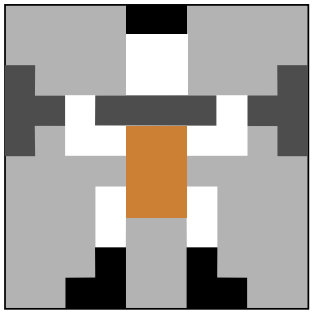
The cell array is subject to all transformations, potentially transforming the rectangular cells into rhomboids (diamond-shaped parallelograms).

The following example shows a function call to create a simple 10-by-10 cell array:

```
 INTEGER COLIA(10,10)/4,4,4,4,5,5,4,4,4,4,
$                     4,4,4,4,1,1,4,4,4,4,
$                     2,4,4,4,1,1,4,4,4,2,
$                     2,2,1,2,2,2,2,1,2,2,
$                     2,4,1,1,3,3,1,1,4,2,
$                     4,4,4,4,3,3,4,4,4,4,
$                     4,4,4,1,3,3,1,4,4,4,
$                     4,4,4,1,4,4,1,4,4,4,
$                     4,4,4,5,4,4,5,4,4,4,
$                     4,4,5,5,4,4,5,5,4,4/

 CALL GCA (25.0,75.0,75.0,25.0,10,10,1,1,10,10,COLIA)
```

This is the cell array that is created:



The cell array function is not fully supported on plotter workstations. On these workstations, cell arrays are simulated by drawing the outline of the transformed cells in the color associated with color index 1, and using solid line style and normal line width.

Principal errors

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

91 Dimensions of color array are invalid

GCLKS

Purpose

GCLKS	
APL code	1301
GKS RCP code	X'38000100' (939524352)

Function: To close GKS.

Control function. Releases all GKS buffers and closes all files.

Parameters

None

Operating states

GKOP

Related functions

Open GKS (GOPKS)

Description

This function releases all GKS buffers and closes all files. The GKS description table, GKS state list and workstation description tables become unavailable. GKS is set into the operating state GKCL (GKS closed). You can reopen GKS by calling Open GKS (GOPKS).

Principal errors

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

GCLRWK

Purpose

GCLRWK	(wkid, cofl)
APL code	1309
GKS RCP code	X'38000600' (939525632)

Function: To clear workstation.

Control function. Clears the display surface of a workstation according to the value of a control flag and releases all segments at the workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.
cofl *(specified by user) (fullword integer)*
The control flag. This is an integer that specifies the clearing of the display surface. The possible values are:

0	(GCONDI)	Conditionally (the screen is cleared only if something is drawn upon it).
1	(GALWAY)	Always (the screen is cleared even if it is empty).

Operating states

WSOP, WSAC

Related functions

Associate segment with workstation (GASGWK), Update workstation (GUWK), Delete segment from workstation (GDSGWK), Set workstation window (GSWKWN), Set workstation viewport (GSWKVP)

Description

This function executes any deferred actions for the specified workstation and then clears the display surface according to the value of *cofl*.

If *cofl* is 0, the display surface clears only if an object appears on it.

If *cofl* is 1, the display surface always clears.

If the workstation state list (WSL) for the workstation indicates that a workstation transformation update is pending, the current workstation window and viewport are set to those requested by previous calls to the functions Set workstation window (GSWKWN) or Set workstation viewport (GSWKVP). The “workstation transformation update state” entry in the WSL is set to *notpending*.

GCLRWK also releases all segments at the workstation. For each segment stored on the workstation, the workstation is deleted from the list of associated workstations kept in the segment state list. Any segments associated only with workstation *wkid* are deleted. The list of segments stored at the workstation is set to *empty*.

Finally, the WSL entry “new frame action necessary at update” for the workstation is set to *no* and the entry “display surface empty” is set to *empty*.

Principal errors

6	GKS not in proper state: GKS shall be either in the state WSOP or in the state WSAC
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
2000	Enumeration type out of range

GCLSG

Purpose

GCLSG

APL code	1358
GKS RCP code	X'38003900' (939538688)

Function: To close segment.

Segment function. Defines the end of a segment.

Parameters

None

Operating states

SGOP

Related functions

Create segment (GCRSG)

Description

This function defines the end of a segment. You cannot add primitives to a closed segment. This function sets the GKS operating state to WSAC.

Principal errors

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

GCLWK

Purpose

GCLWK	(wkid)
APL code	1303
GKS RCP code	X'38000300' (939524864)

Function: To close workstation.

Control function. Disengages a workstation from GKS.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.

Operating states

WSOP, WSAC, SGOP

Related functions

Open workstation (GOPWK), Close GKS (GCLKS), Deactivate workstation (GDAWK), Update workstation (GUWK)

Description

This function performs an implicit Update workstation (GUWK) (with the update regeneration flag parameter set to *perform*) and then closes the workstation, by taking the following actions:

1. The workstation state list is released.
2. The workstation identifier is deleted from the list of open workstations in the GKS state list.
3. The workstation identifier is deleted from the list of associated workstations in the segment state list of each segment at the workstation. If this is the last workstation associated with the segment, the segment is deleted.
4. The connection to the workstation is released. The display surface of the workstation is not cleared explicitly, but if it is the user console, it can be cleared when functions causing output to the console (for example the Message (GMSGS) function) are called, or when GKS is closed.

If no other workstation is open, this function sets the GKS operating state to GKOP.

Note that, if one of the errors listed below is reported, the workstation is not closed. However, other errors, outside the control of GDDM-GKS, may be detected, while allowing the function to complete; for example, error 304 can be reported if GDDM-GKS cannot access the device because your program has closed it using a GDDM DSCLS call.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
29	Specified workstation is active

GCRSG

Purpose

GCRSG	(sgna)
APL code	1357
GKS RCP code	X'38003800' (939538432)

Function: To create segment.

Segment function. Names a new segment and opens it to receive new output primitives.

Parameters

sgna *(specified by user) (fullword integer)*
The name (identifier) of the new segment.

Operating states

WSAC

Related functions

Close segment (GCLSG), Delete segment (GDSG), Rename segment (GRENSG)

Description

This function creates a new segment and sets GKS into the operating state SGOP.

The following actions are taken:

- A segment state list is established for the new segment.
- The segment name is recorded as the name of the open segment in the GKS state list. Subsequent output primitives are collected in the new segment until the Close segment (GCLSG) function is next called.
- All currently active workstations are included in the list of associated workstations in the segment state list.
- The segment name is added to the list of segments in the state list of each active workstation.
- The segment name is added to the list of segment names in use in the GKS state list.

Only one segment can be open at a time. Primitive attributes are not affected.

Once a segment is closed it cannot be reopened. If you want to make changes to a closed segment, it must be deleted and recreated.

Attributes are associated with a segment, so that the appearance of the segment can be modified when it is used again. Functions are available to change the settings of these attributes:

Detectability
Highlighting
Segment priority
Visibility

Segments can be redrawn on a workstation, renamed, ordered, deleted, copied to different workstations, and inserted into the open segment or into the stream of primitives outside segments.

Segments can also be rotated, shifted, and scaled, using a transformation matrix. (See the functions Evaluate transformation matrix (GEVTM) and Accumulate transformation matrix (GACTM).)

Principal errors

3	GKS not in proper state: GKS shall be in the state WSAC
120	Specified segment name is invalid
121	Specified segment name is already in use

GCSGWK

Purpose

GCSGWK	**(wkid, sgna)**
APL code	1363
GKS RCP code	X'38003E00' (939539968)

Function: To copy segment to workstation.

Segment function. Sends segment primitives to the workstation for display after transformation and clipping.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.
sgna *(specified by user) (fullword integer)*
The segment name selected when the segment was created.

Operating states

WSOP, WSAC

Related functions

Create segment (GCRSG), Associate segment with workstation (GASGWK), Open workstation (GOPWK)

Description

This function sends segment primitives from *workstation independent segment storage (WISS)* to the display surface of workstation *wkid*. Primitives in the segment *sgna* are sent after being transformed and clipped with the stored clipping rectangle (the normalization transformation viewport current when each primitive was created).

All primitives keep the values of the primitive attributes assigned to them when they were created.

If the segment *sgna* is not present in WISS, error 124 is reported. The workstation *wkid* cannot be the WISS.

This function cannot be invoked when a segment is open. The primitives copied to the workstation are not stored in a segment; they remain only until the workstation is updated or cleared. By contrast, the Associate segment with workstation (GASGWK) function makes the segment a permanent part of the workstation output.

Principal errors

6	GKS not in proper state: GKS shall be either in the state WSOP or in the state WSAC
20	Specified workstation identifier is invalid
25	Specified workstation is not open
27	Workstation Independent Segment Storage is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
120	Specified segment name is invalid
124	Specified segment does not exist on Workstation Independent Segment Storage

GDAWK

Purpose

GDAWK	(wkid)
APL code	1308
GKS RCP code	X'38000500' (939525376)

Function: To deactivate workstation.

Control function. Ends GKS output to the workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.

Operating states

WSAC

Related functions

Activate workstation (GACWK), Close workstation (GCLWK), Close GKS (GCLKS)

Description

This function ends GKS graphics output to the workstation. Primitives are no longer sent to this workstation and new segments are no longer stored. Segments already stored are retained.

The specified workstation is marked inactive in the workstation state list, and is deleted from the list of active workstations in the GKS state list. If no other workstation is active, the GKS operating state is set to WSOP.

Principal errors

3	GKS not in proper state: GKS shall be in the state WSAC
20	Specified workstation identifier is invalid
30	Specified workstation is not active
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT

GDSG

Purpose

GDSG	**(sgna)**
APL code	1360
GKS RCP code	X'38003B00' (939539200)

Function: To delete segment.

Segment function. Deletes a segment from all workstations.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name selected when the segment was created.

Operating states

WSOP, WSAC, SGOP

Related functions

Create segment (GCRSG), Delete segment from workstation (GDSGWK), Set deferral state (GSDS)

Description

This function deletes the segment from all workstations. The segment name may then be reused by the application program.

The following actions are taken:

- The segment name is removed from the list of segments at each workstation that contains it.
- The segment name is deleted from the list of segment names in use in the GKS state list.
- The segment state list for the segment is deleted.

Whether the segment is immediately deleted from the display of a workstation, or deletion is deferred, depends on the implicit regeneration mode for the workstation; this can be set using the function Set deferral state (GSDS).

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
120	Specified segment name is invalid
122	Specified segment does not exist
125	Specified segment is open

GDSGWK

Purpose

GDSGWK	**(wkid, sgna)**
APL code	1361
GKS RCP code	X'38003C00' (939539456)

Function: To delete segment from workstation.

Segment function. Deletes a segment from the specified workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.
sgna *(specified by user) (fullword integer)*
The segment name selected when the segment was opened.

Operating states

WSOP, WSAC, SGOP

Related functions

Create segment (GCRSG), Delete segment (GDSG), Set deferral state (GSDS), Activate workstation (GACWK)

Description

This function deletes the segment from the workstation *wkid*.

The following actions are taken:

- The segment name is removed from the list of segments in the workstation state list.
- The workstation identifier is deleted from the list of associated workstations in the segment state list. If the segment is no longer associated with any workstation, it is deleted, as in the function Delete segment (GDSG).

Whether the segment is immediately deleted from the display of the workstation, or deletion is deferred, depends on the implicit regeneration mode of the workstation; this can be set using the function Set deferral state (GSDS).

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
120	Specified segment name is invalid
123	Specified segment does not exist on specified workstation
125	Specified segment is open

GECLKS

Purpose

GECLKS

APL code	1304
GKS RCP code	X'3800F000' (939585536)

Function: To emergency close GKS.

Error-handling function. Closes GKS, saving as much graphical data as possible at the workstations.

Parameters

None

Operating states

GKCL, GKOP, WSOP, WSAC, SGOP

Related functions

Error handling (GERHND)

Description

The following actions are performed (if possible):

1. Close segment (if open)
2. Update workstation for all open workstations
3. Deactivate all active workstations
4. Close all open workstations
5. Close GKS.

This function may be called even if the error state is *on*. If GKS is already closed (operating state GKCL), no action is taken.

Principal errors

None

GERHND

Purpose

GERHND **(errnr,fctid,errfil)**

Function: To handle errors. The default or user-supplied function invoked by GDDM-GKS when errors are detected.

Parameters

errnr *(specified by GDDM-GKS) (fullword integer)*
The error number.

fctid *(specified by GDDM-GKS) (fullword integer)*
The identification of the GKS function called by the application program in which the error was detected. See the list of functions, at the beginning of this chapter; this list gives the number identifying each function.

errfil *(specified by GDDM-GKS) (fullword integer)*
The error file. This parameter has been defined in Open GKS (GOPKS).

Related functions

Error logging (GERLOG), Emergency close GKS (GECLKS)

Description

This function is invoked by GDDM-GKS when an error has been detected during a call to any GKS function that does not have an error indicator parameter. (The parameters are supplied by GDDM-GKS; you do not code these.)

The GDDM-GKS default function, Error handling (GERHND), just calls the Error logging (GERLOG) function with the same parameters.

It is not possible for an application to call the default error-handling function directly.

You can replace the default error-handling function with a procedure designed to your own requirements. Refer to “Error handling in GDDM-GKS” on page 56 for details on how to do this.

Principal errors

None

GERLOG

Purpose

GERLOG	(errnr, fctid, errfil)
APL code	1306
GKS RCP code	X'3800F100' (939585792)

Function: To error logging.

Error-handling function. Writes error messages to the error file.

Parameters

errnr *(specified by user) (fullword integer)*
The error number.

fctid *(specified by user) (fullword integer)*
The identification of the GKS function called by the application program that caused the error to be detected.

errfil *(specified by user) (fullword integer)*
The error file. This parameter has been defined in the Open GKS (GOPKS) function call.

Related functions

Error handling (GERHND)

Description

This function can be called from a user-supplied error-handling procedure to log GDDM-GKS errors in the error file. Two messages are written to the error file. The first message is always ADM3500, which gives the GKS error number and the name of the function in which the error occurred; the actual error message then follows, giving a brief description of the error. For example:

```
ADM3500 I GKS ERROR 60 IN FUNCTION GSPLI
ADM3560 E POLYLINE INDEX IS INVALID
```

If you call this function from your main program (when no GKS error has been detected), or with parameters different from those passed to your error-handling procedure, this function has no effect: no error is reported and no message is logged.

Principal errors

None

GESC

Purpose

GESC	(fctid, lidr, idr, mlodr, lodr, odr)
APL code	1491
GKS RCP code	X'38000B00' (939526912)

Function: To escape.

Control function. Allows the use of non-standard workstation capabilities. (See the section “GDDM-GKS restrictions” below.)

Parameters

fctid *(specified by user) (fullword integer)*
The function identifier.
lidr *(specified by user) (fullword integer)*
The length of the data record *idr*.
idr *(specified by user) (array of 80-byte character tokens)*
The data record.
mlodr *(specified by user) (fullword integer)*
The maximum length of the output data record.
lodr *(returned by GDDM) (fullword integer)*
The number of array elements occupied by *odr*.
odr *(returned by GDDM) (array of 80-byte character tokens)*
The output data record.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

None.

Description

(See the section “GDDM-GKS restrictions” below.)

This function provides for implementation-dependent, non-standard activity at a workstation. For example, a workstation that has received *cell array* primitive output could be instructed to process that output in some way.

GDDM-GKS Restrictions

GDDM-GKS does not provide any escape functions. If this function is called, one of the errors 8, 180, or 181 is reported.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
180	Specified escape function is not supported
181	Specified escape function identification is invalid

GEVTM

Purpose

GEVTM	(xo, yo, dx, dy, phi, fx, fy, sw, mout)
APL code	1416
GKS RCP code	X'38006900' (939550976)

Function: To evaluate transformation matrix.

Utility function. Computes a segment transformation matrix for use in scaling, shifting, and rotating segments.

Parameters

xo *(specified by user) (short floating point)*
yo *(specified by user) (short floating point)*
The given point in either world or normalized device coordinates.
dx *(specified by user) (short floating point)*
dy *(specified by user) (short floating point)*
The shift vector in either world or normalized device coordinates.
phi *(specified by user) (short floating point)*
The rotation angle in radians; a positive value indicates a counterclockwise direction.

fx *(specified by user) (short floating point)*
fy *(specified by user) (short floating point)*
The scale factors.
sw *(specified by user) (fullword integer)*
The coordinate switch. The possible values are:
0 (GWC) World coordinates.
1 (GNDC) Normalized device coordinates.
mout *(returned by GDDM) (array of short floating-point numbers)*
The segment transformation matrix.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Accumulate transformation matrix (GACTM)

Description

This function computes values for the matrix defined by the fixed point, shift vector, rotation angle, and scale factors. The resulting new matrix is used, for example, in the functions Insert segment (GINSG) and Set segment transformation (GSSGT).

The identity matrix is stored with each segment in the segment state list when it is created. When you call GACTM or GEVTM, and then Set segment transformation (GSSGT), the newly created matrix replaces the matrix in the segment state list, and it controls the appearance of the segment until a new matrix is set. The effects of repeated applications of a matrix are not cumulative. Therefore you "accumulate" a matrix in order to retain the effects of a previous matrix while making further changes in the appearance of a display.

The coordinate switch (*sw*) governs whether the shift vector and fixed point are given world coordinate units or normalized device coordinate units. If you use WC units, the shift vector and the fixed point are transformed by the current normalization transformation.

The shift vector is entered as the relative displacement between two points. For example, if the input fixed point is (10.0, 10.0), and the segment transformation shifts the display to (25.0, 25.0), the shift vector is entered as (15.0, 15.0).

The rotation angle is entered in radians. To convert an angle in degrees to radians, you use the following equation:

```
radians = pi/180 * degrees
```

For example, to rotate a display 45 degrees, you enter 0.7853982 as the value for the rotation angle.

You enter identity values for those aspects that you do not want to change in any specific accumulation of matrixes. These are 0.0 (zero) for shift and rotation and 1.0 (one) for scale. The segment transformation is stored as an array with six elements. Point (*x*, *y*) is transformed into point (*x'*, *y'*) as follows:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} M(1) & M(3) & M(5) \\ M(2) & M(4) & M(6) \end{bmatrix} X \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

The order of transformation is: scale, rotate (both relative to the input fixed point), and shift. Output values defining the shift components are in NDC units. The other elements do not have units.

Principal errors

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

2000 Enumeration type out of range

GFA

Purpose

GFA	(n, px, py)
APL code	1315
GKS RCP code	X'38000F00' (939527936)

Function: To fill area.

Output function. Defines a polygon and fills it using current *fill area* attributes.

Parameters

n *(specified by user) (fullword integer)*
The number of points in the polygon.

px *(specified by user) (array of short floating-point numbers)*

py *(specified by user) (array of short floating-point numbers)*
The coordinates of the points, in world coordinates.

Operating states

WSAC, SGOP

Related functions

Set fill area index (GSFAI), Set fill area representation (GSFAR), Set fill area interior style (GSFAIS), Set fill area style index (GSFASI), Set fill area color index (GSFACI), Set window (GSWN).

Description

This function fills the polygon described by the input points, using the current fill area attributes.

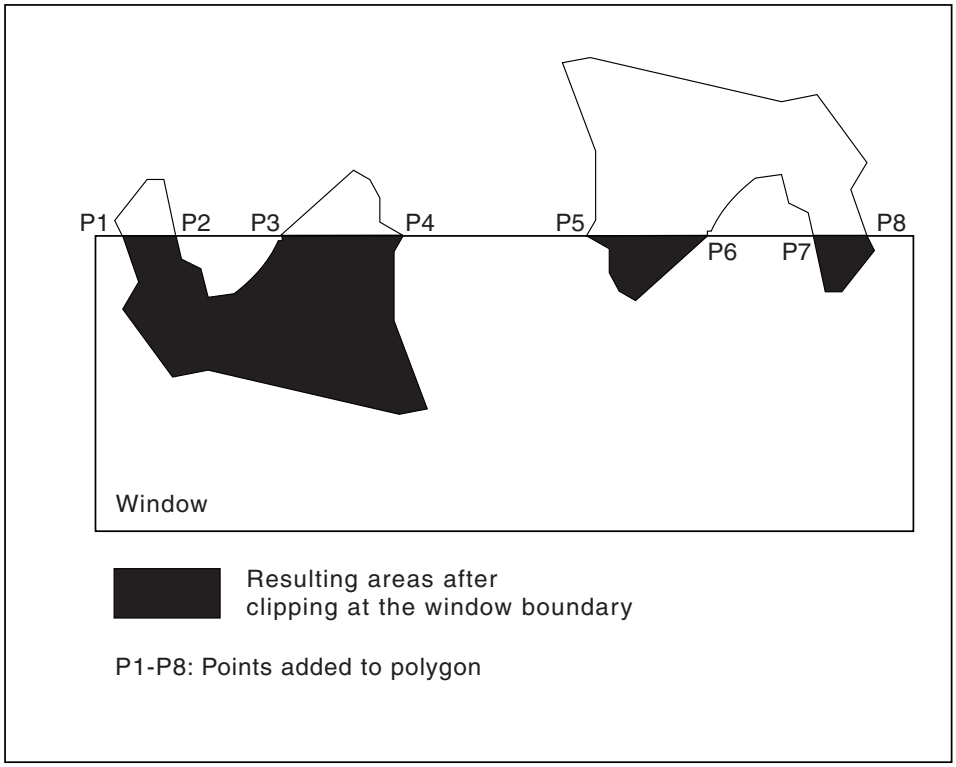
If the interior style is *hollow*, the boundary of the polygon is drawn instead using the fill area color index. The boundary is not drawn for other styles.

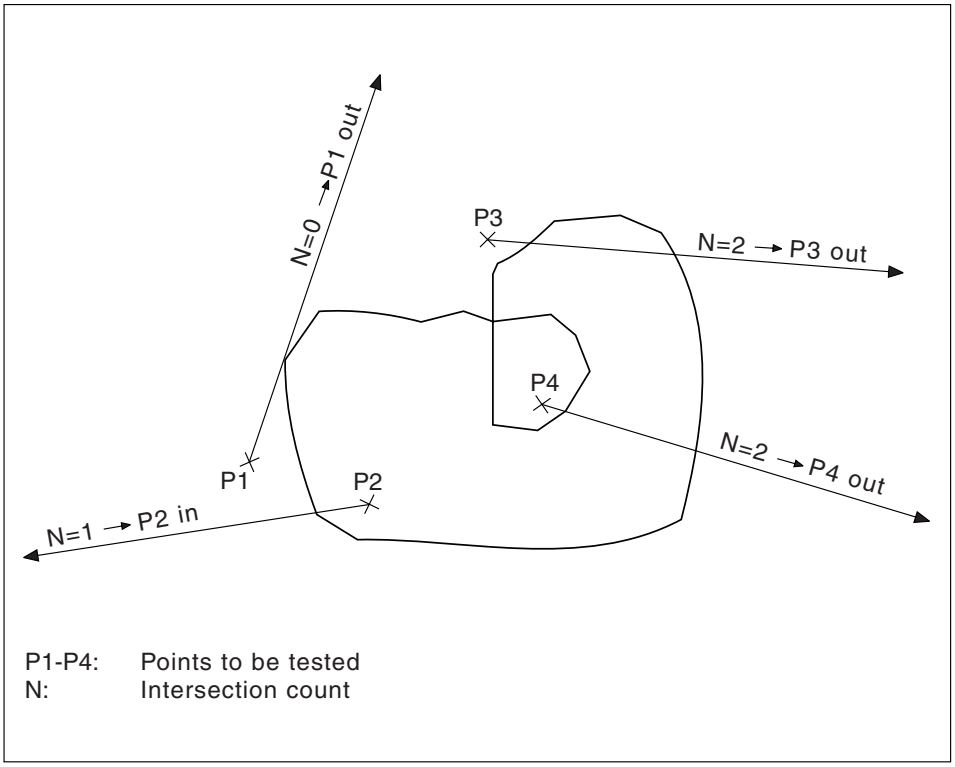
The fill area is subject to all transformations; however, interior styles *pattern* and *hatch* are unaffected by transformations.

If clipping is on, the fill area is clipped to the window. If parts of the fill area are clipped, the resulting new boundaries become part of the area boundaries. Multiple subareas may be generated.

The first illustration on the next page shows the effect of clipping on fill area boundaries.

The second illustration shows how GKS defines the interior of a polygon. For a given point, create a straight line starting at that point and going to infinity. If the number of intersections between the straight line and the polygon is odd, the point is within the polygon. If the number of intersections is even, the point is outside. If the straight line passes a polygon vertex tangentially, the intersection count is not affected.





Principal errors

5	GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
100	Number of points is invalid

GFLUSH

Purpose

GFLUSH	**(wkid, icl, idnr)**
APL code	1392
GKS RCP code	X'38005E00' (939548160)

Function: To flush device events.

Input function. Deletes all specified input class events from the input queue. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.
icl *(specified by user) (fullword integer)*
The input class. The possible values are:
1 (GLOCAT) Locator

2 (GSTROK) Stroke
3 (GVALUA) Valuator
4 (GCHOIC) Choice
5 (GPICK) Pick
6 (GSTRIN) String

idnr *(specified by user) (fullword integer)*
The logical input device number.

Operating states

WSOP, WSAC, SGOP

Related functions

Await event (GWAIT)

Description

(See the section “GDDM-GKS restrictions” below.)

GKS maintains an input queue with input events of any logical input devices that are operating in *event* mode. This function removes from the input queue all events that came from the specified input device. This ensures that no unwanted or old interactions remain on the input queue that could interfere with newly started interaction processes.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS will report error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
143	EVENT and SAMPLE input mode are not available at this level of GKS
147	Input queue has overflowed
2000	Enumeration type out of range

GGDP

Purpose

GGDP	**(n, px, py, primid, ldr, datrec)**
APL code	1493
GKS RCP code	X'38001100' (939528448)

Function: To generalized drawing primitive.

Output function. Generates geometric output that is both implementation-dependent and workstation-dependent. (See the section "GDDM-GKS restrictions" below.)

Parameters

n *(specified by user) (fullword integer)*
The number of points present in the *pxa* and *pya* vectors.
px *(specified by user) (array of short floating-point numbers)*
py *(specified by user) (array of short floating-point numbers)*
The coordinates of the points in world coordinates.
primid *(specified by user) (fullword integer)*
The generalized drawing primitive identifier.
ldr *(specified by user) (fullword integer)*
The length of the data record *datrec*.
datrec *(specified by user) (array of 80-byte character tokens)*
The data record.

Operating states

WSAC, SGOP

Related functions

Inquire generalized drawing primitive (GQGDP)

Description

This function generates a set of special geometric output primitives called generalized drawing primitives (GDPs), which supplement and extend the standard *polyline*, *polymarker*, *text*, *fill area*, and *cell array* output primitives of GKS.

GDDM-GKS Restrictions No GDPs are provided by GDDM-GKS. If this function is invoked, GDDM-GKS reports error 104.

Principal errors

5	GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
100	Number of points is invalid
102	Generalized drawing primitive identifier is invalid
103	Content of generalized drawing primitive data record is invalid
104	At least one active workstation is not able to generate the specified generalized drawing primitive
105	At least one active workstation is not able to generate the specified generalized drawing primitive under the current transformations and clipping rectangle

GGTCH

Purpose

GGTCH	(stat, chnr)
APL code	1398
GKS RCP code	X'38006200' (939549184)

Function: To get choice.

Input function. Returns a *choice* number. (See the section "GDDM-GKS restrictions" below.)

Parameters

stat *(returned by GDDM) (fullword integer)*
The status of the choice device. The possible choices are:
0 (GOK) OK
0 (GNCHOI) No choice

The status indicates whether or not the choice number obtained from the operator is valid.

chnr *(returned by GDDM) (fullword integer)*
The choice number. The choice number is an integer that represents one of a number of choices available on a choice device.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize choice (GINCH), Request choice (GRQCH), Set choice mode (GSCHM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function returns the choice logical input value in the current event report.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS
150	No input value of the correct class is in the current event report

GGTITM

Purpose

GGTITM	(wkid, type, idrl)
APL code	1410
GKS RCP code	X'38006600' (939550208)

Function: To get item type from GKSM.

Metafile function. Returns the next item type and data record length.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.
type *(returned by GDDM) (fullword integer)*
The item type. This number identifies the stored item.
idrl *(returned by GDDM) (fullword integer)*
The item data record length.

Operating states

WSOP, WSAC, SGOP

Related functions

Write item to GKSM (GWITM), Read item from GKSM (GRDITM), Interpret item (GIITM)

Description

A GKS metafile (GKSM) is a sequential file that behaves like a workstation. To retrieve information from a metafile, you must open a *metafile input (MI)* workstation. This function returns the type and the data record length of the current metafile item. You can now use the function Read item from GKSM (GRDITM) to obtain the item data record and to make the next item in the metafile the current item.

Refer to Appendix C, “Metafile structure” on page 383 for a list of the metafile item types.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
34	Specified workstation is not of category MI
162	No item is left in GKS Metafile input
163	Metafile item is invalid

GGTLC

Purpose

GGTLC	(tnr, lpx, lpy)
APL code	1393
GKS RCP code	X'38005F00' (939548416)

Function: To get locator.

Input function. Returns the *locator* position in world coordinates. (See the section "GDDM-GKS restrictions" below.)

Parameters

tnr *(returned by GDDM) (fullword integer)*
The normalization transformation number.
lpx *(returned by GDDM) (short floating point)*
lpy *(returned by GDDM) (short floating point)*
The locator position in world coordinates.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize locator (GINLC), Request locator (GRQLC), Set locator mode (GSLCM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function returns the locator logical input value in the current event report.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS
150	No input value of the correct class is in the current event report

GGTPK

Purpose

GGTPK	(stat, sgna, pkid)
APL code	1399
GKS RCP code	X'38006300' (939549440)

Function: To get pick.

Input function. Returns the segment name, status and *pick* identifier. (See the section "GDDM-GKS restrictions" below.)

Parameters

stat *(returned by GDDM) (fullword integer)*
The status. The possible values are:
0 (GOK) OK
1 (GNPICK) No pick

If the status equals *OK*, the pick input contains a segment name and pick identifier. The *OK* status also indicates that the pick device was pointing to an output primitive stored in a segment when the event was created. A *no pick* status means that the device was not pointing to a primitive inside a segment.

sgna *(returned by GDDM) (fullword integer)*
The segment name. This identifies the picked segment.

pkid *(returned by GDDM) (fullword integer)*
The pick identifier. This identifies the picked object.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize pick (GINPK), Request pick (GRQPK), Set pick mode (GSPKM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function returns the pick logical value in the current event report.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS
150	No input value of the correct class is in the current event report
2000	Enumeration type out of range

GGTSK

Purpose

GGTSK	(n, tnr, np, px, py)
APL code	1395
GKS RCP code	X'38006000' (939548672)

Function: To get stroke.

Input function. Returns the points of the *stroke* input from the current event queue. (See the section "GDDM-GKS restrictions" below.)

Parameters

n *(specified by user) (fullword integer)*
The maximum number of points to be returned.
tnr *(returned by GDDM) (fullword integer)*
The normalization transformation number.
np *(returned by GDDM) (fullword integer)*
The number of points actually returned.
px *(returned by GDDM) (array of short floating-point numbers)*
py *(returned by GDDM) (array of short floating-point numbers)*
The points in world coordinates.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize stroke (GINSK), Request stroke (GRQSK), Set stroke mode (GSSKM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function returns the stroke logical input value in the current event report.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS
150	No input value of the correct class is in the current event report

GGTST

Purpose

GGTST **(lostr, str)**

Function: Input function. Returns a character string. Use this call only if your program is written in FORTRAN IV or VS FORTRAN. Otherwise, use the function Get string (GGTSTS) instead. (See the section “GDDM-GKS restrictions” below.)

Parameters

lostr *(returned by GDDM) (fullword integer)*
The number of characters returned.

str *(returned by GDDM) (character)*
The string. In a VS FORTRAN program, the string can be of variable length. For FORTRAN IV, you must define the string as CHARACTER*80.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize string (FORTRAN only) (GINST), Request string (FORTRAN only) (GRQST), Set string mode (GSSTM)

Description

(See the section “GDDM-GKS restrictions” below.)

This function returns the *string* logical input value in the current event report.

GDDM-GKS restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

```
  7  GKS not in proper state: GKS shall be in one of the states
     WSOP, WSAC, or SGOP
143  EVENT and SAMPLE input mode are not available at this level
     of GKS
150  No input value of the correct class is in the current event
     report
```

GGTSTS

Purpose

GGTSTS	(mstr, lostr, str)
APL code	1401
GKS RCP code	X'38006400' (939549696)

Function: To get string.

Input function. Returns a character string. If your program is written in FORTRAN IV or VS FORTRAN, use the function Get string (FORTRAN only) (GGTST) instead. (See the section "GDDM-GKS restrictions" below.)

Parameters

mstr *(specified by user) (fullword integer)*
The maximum number of characters in *str*.
lostr *(returned by GDDM) (fullword integer)*
The number of characters returned.
str *(returned by GDDM) (character)*
The string.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize string (GINSTS), Request string (GRQSTS), Set string mode (GSSTM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function returns the *string* logical input value in the current event report.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS
150	No input value of the correct class is in the current event report

GGTVL

Purpose

GGTVL	(val)
APL code	1396
GKS RCP code	X'38006100' (939548928)

Function: To get valuator.

Input function. Returns the *valuator* input. (See the section "GDDM-GKS restrictions" below.)

Parameters

val *(returned by GDDM) (short floating point)*
The value returned from the valuator device. The value of a valuator device is a real number in the range given by the lower and upper limit of that device.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize valuator (GINVL), Request valuator (GRQVL), Set valuator mode (GSVLM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function returns the valuator logical input value in the current event report.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS
150	No input value of the correct class is in the current event report

GIITM

Purpose

GIITM	(type, idrl, ldr, datrec)
APL code	1412
GKS RCP code	X'38006800' (939550720)

Function: To interpret item.

Metafile function. Interprets the contents of a metafile item.

Parameters

type *(specified by user) (fullword integer)*
The item type. The item type is a number generated internally by GDDM-GKS when a call to a GKS function causes metafile items to be stored on a metafile.
idrl *(specified by user) (fullword integer)*
The length of the item data record (the number of significant characters in the data record array).
ldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.
datrec *(specified by user) (array of 80-byte character tokens)*
An array containing the item data record.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Write item to GKSM (GWITM), Get item type from GKSM (GGTITM), Read item from GKSM (GRDITM)

Description

This function interprets the contents of the metafile item provided by your application, causing changes to the state lists and generating graphical output. When reading a metafile you can get the type and the data record length of the current metafile item by using the function Get item type from GKSM (GGTITM). The function Read item from GKSM (GRDITM) returns the data record of the current metafile item. The item type, the data record length, and the data record can then be passed to this function (GIITM).

Note that user items, written to output metafiles by the function Write item to GKSM (GWITM), cannot be interpreted. Appendix C, “Metafile structure” on page 383 lists all metafile items that can be generated by GDDM-GKS.

When an item is interpreted, the effects on the state lists are the same as if the function corresponding to the item were called directly.

To display a picture stored as a metafile:

1. Open and activate the workstation on which the picture is to be displayed, by using the Open workstation (GOPWK) and Activate workstation (GACWK) functions.
2. Open the metafile containing the picture to be displayed as a metafile input (MI) workstation.

   Then for each item in the metafile (until the function Get item type from GKSM (GGTITM) returns the End item, type 0):
3. Use the Get item type from GKSM (GGTITM) function to return the item type and the length of the data record to your program.
4. Use the Read item from GKSM (GRDITM) function to return the item data record to your program and to make the next item in the metafile the current item.
5. Call this function using the returned item type, data record length, and data record.

Items corresponding to GKS functions that normally apply only to a single workstation are interpreted at all active workstations.

If *datrec* is too small to contain the length specified by *idrl*, error 2003 is reported.

In addition to the errors given below, other errors can be reported, depending on the type of the item being interpreted.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
161	Item length is invalid
163	Metafile item is invalid
164	Item type is not a valid GKS item
165	Content of item data record is invalid for the specified item type
167	User item cannot be interpreted
168	Specified function is not supported in this level of GKS
2003	Invalid data record

GINCH

Purpose

GINCH	**(wkid, chdnr, istat, ichnr, pet, xmin, xmax, ymin, ymax, ldr, datrec)**
APL code	1496
GKS RCP code	X'38004800' (939542528)

Function: To initialize choice.

Input function. Initializes a *choice* input device at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
chdnr *(specified by user) (fullword integer)*
The choice device number. The possible values are:
1 The Enter key
2 The PF keys
3 The alphanumeric light pen
4 The data keys
5 The mouse or tablet buttons
istat *(specified by user) (fullword integer)*
The initial status. The possible values are:
1 (GOK) OK
2 (GNCHOI) No choice
ichnr *(specified by user) (fullword integer)*
The initial choice number. If *istat* is *GOK*, this parameter must be between 1 and the maximum number of choice alternatives (held in the workstation description table) for the device.
pet *(specified by user) (fullword integer)*
The prompt and echo type.
xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The echo area in device coordinates.
ldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.
datrec *(specified by user) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Request choice (GRQCH), Set choice mode (GSCHM), Inquire number of available logical input devices (GQLI), Inquire default choice device data (GQDCH)

Description

This function initializes the choice device by defining the initial status, initial choice number, prompt and echo type, echo area, and choice data record. These are stored in the workstation state list entry for the specified choice device.

GDDM-GKS provides one prompt and echo type:

pet=1 If *wkid* is an IBM 5080 Graphics System equipped with lighted PF keys, the keys are lit. For PF keys on other workstations, and for mouse and tablet devices, no prompt is displayed.

No choice echo is displayed; the echo area and choice data record are not used.

Not all choice devices are available at all workstations. You can use the function Inquire number of available logical input devices (GQLI) to determine which devices are available at a particular workstation. You can use the function Inquire default choice device data (GQDCH) to obtain the maximum number of choice alternatives for a particular choice device.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
51	Rectangle definition is invalid
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode
144	Specified prompt and echo type is not supported on this workstation
145	Echo area is outside display space
146	Contents of input data record are invalid
152	Initial value is invalid
2000	Enumeration type out of range

GINLC

Purpose

GINLC	**(wkid, lcdnr, tnr, ipx, ipy, pet, xmin, xmax, ymin, ymax, ldr, datrec)**
APL code	1370
GKS RCP code	X'38004500' (939541760)

Function: To initialize locator.

Input function. Initializes a *locator* input device at a workstation. (See the section "GDDM-GKS restrictions" below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
lcdnr *(specified by user) (fullword integer)*
The locator device number. GDDM-GKS provides only one locator device at a workstation. The locator device number is always 1.
tnr *(specified by user) (fullword integer)*
The initial normalization transformation number.
ipx *(specified by user) (short floating point)*
ipy *(specified by user) (short floating point)*
The initial locator position in world coordinates.

pet *(specified by user) (fullword integer)*
The prompt and echo type. GDDM-GKS provides the standard prompt and echo types 1, 2, 3, 4, 5, and one implementation-dependent type, -1.
xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The echo area in device coordinates. The echo area is not used for any of the prompt and echo types provided.
ldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*. If the value specified is 0, a workstation-dependent default data record is used.
datrec *(specified by user) (array of 80-byte character tokens)*
The data record, containing additional information depending on the prompt and echo type.

Data records are defined for prompt and echo types 4, 5 and -1. The Pack data record (GPRECS) function can be used to create the input data records. (See the section "GDDM-GKS restrictions" below.)

Operating states

WSOP, WSAC, SGOP

Related functions

Request locator (GRQLC), Set locator mode (GSLCM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function initializes the locator device by defining the initial locator position, initial normalization transformation number, prompt and echo type, echo area, and locator data record. These are stored in the workstation state list entry for the locator device.

The initial locator position is where the graphic cursor first appears. It must lie within the window of the initial normalization transformation specified by *tnr*. When you call the function Request locator (GRQLC), the transformation *tnr* is used to transform the initial locator position to NDC. The transformed point must lie within the workstation window and outside the viewport of any normalization transformation with higher priority than *tnr*; otherwise, the initial locator position is workstation-dependent.

The prompt and echo types provided by GDDM-GKS are:

pet=1 The current position of the locator is designated using the device standard graphic cursor.

pet=2 The current position of the locator is designated using a cross-hair cursor.

Available for 3179-G, 3270-PC/G and /GX ranges, 5550-family, and 5080 workstations only. Note that, even though this prompt and echo type has been requested, at 3179-G and the 3270-PC/G and /GX ranges of workstations, the cursor type used can be preselected or changed by the operator by using the keyboard functions.

pet=3 The current position of the locator is designated using a tracking cross.

Available for 3270-PC/G and /GX ranges, and 5080 workstations only.

pet=4 The current position of the locator is designated using a rubber-band line. A rubber-band line constantly connects the initial locator position to the current position as it moves.

Available for 3270-PC/G and /GX ranges, and 5080 workstations only. The data record is not used but is defined; it allows either the current *polyline* attributes, or attributes specified in the data record, to be used to draw the rubber line.

If the current attributes are to be used, the parameters of the Pack data record (GPRECS) function are:

```
il   = 1
ia   = (attribute control flag=GCURNT)
rl   = 0
ra   = ()
sl   = 0
lstr = ()
str  = ()
```

If attributes are to be specified in the data record, the parameters of the Pack data record (GPRECS) function are:

```
il   = 7
ia   = (attribute control flag=GSPEC),
        line type aspect source flag (ASF),
        line width scale factor ASF,
        polyline color index ASF,
        polyline index,
        line type,
        polyline color index)
rl   = 1
ra   = (line width scale factor)
sl   = 0
lstr = ()
str  = ()
```

pet=5 The current position of the locator is designated using a “rubber-box” rectangle. The diagonal of the rectangle is the line connecting the initial locator position and the current position.

Available for 3270-PC/G and /GX ranges, and 5080 workstations only. The data record is not used but is defined; it allows either the current *polyline* or *fill area* attributes, or attributes specified in the data record, to be used to draw the rectangle.

If the current attributes are to be used, the parameters of the Pack data record (GPRECS) function are:

```
il   = 2
ia   = (polyline/fill area control flag=GPLINE or
       GFILLA, attribute control flag=GCURNT)
rl   = 0
ra   = ()
sl   = 0
lstr = ()
str  = ()
```

If polyline attributes are to be specified in the data record, the parameters of the Pack data record (GPRECS) function are:

```
il   = 8
ia   = (polyline/fill area control flag=GPLINE,
        attribute control flag=GSPEC,
        line type ASF,
        line width scale factor ASF,
        polyline color index ASF,
        polyline index,
        line type,
        polyline color index)
rl   = 1
ra   = (line width scale factor)
sl   = 0
lstr = ()
str  = ()
```

If fill area attributes are to be specified in the data record, the parameters of the Pack data record (GPRECS) function are:

```
il   = 9
ia   = (polyline/fill area control flag=GFILLA,
        attribute control flag=GSPEC,
        fill area interior style ASF,
        fill area style index ASF,
        fill area color index ASF,
        fill area index,
        fill area interior style,
        fill area style index,
        fill area color index)
rl   = 0
ra   = ()
sl   = 0
lstr = ()
str  = ()
```

pet=-1 The current position of the cursor is designated using a segment locator. A copy of the segment specified in the data record *datrec* is moved as the current locator position changes.

Available for 3270-PC/G and /GX ranges, and 5080 workstations only.

The data record must contain the name of the segment to be attached to the locator. The segment need not exist when this function is invoked but must exist at the workstation when you call the Request locator (GRQLC) function. The parameters of the Pack data record (GPRECS) function are:

```
il   = 1
ia   = (segment name)
rl   = 0
ra   = ()
sl   = 0
lstr = ()
str  = ()
```

GDDM-GKS Restrictions

The data records for prompt and echo types 4 and 5 are not honored by the workstations supported by GDDM-GKS.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
51	Rectangle definition is invalid
60	Polyline index is invalid
63	Linetype is equal to zero
65	Linewidth scale factor is less than zero
80	Fill area index is invalid
84	Style (pattern or hatch) index is equal to zero
92	Color index is less than zero
120	Specified segment name is invalid
122	Specified segment does not exist
123	Specified segment does not exist on specified workstation
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode
144	Specified prompt and echo type is not supported on this workstation
145	Echo area is outside display space
146	Contents of input data record are invalid
152	Initial value is invalid
2000	Enumeration type out of range

GINPK

Purpose

GINPK	**(wkid, pkdnr, istat, isgna, ipkid, pet, xmin, xmax, ymin, ymax, ldr, datrec)**
APL code	1497
GKS RCP code	X'38004900' (939542784)

Function: To initialize pick.

Input function. Initializes a *pick* input device at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pkdnr *(specified by user) (fullword integer)*
The pick device number. GDDM-GKS provides only one pick device at a workstation. The pick device number is always 1.
istat *(specified by user) (fullword integer)*
The initial status. The possible values are:
0 (GOK) OK
1 (GNPICK) No pick

isgna *(specified by user) (fullword integer)*
The initial segment name. If *istat* is *GOK*, this parameter must identify a segment at the workstation that is visible and detectable.
ipkid *(specified by user) (fullword integer)*
The initial pick identifier. If *istat* is *GOK*, this parameter must be the pick identifier of at least one primitive in the segment *isgna*.
pet *(specified by user) (fullword integer)*
The prompt and echo type.
xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The echo area in device coordinates.
ldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.
datrec *(specified by user) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Request pick (GRQPK), Set pick mode (GSPKM)

Description

This function initializes the pick input device by defining the initial segment, initial pick identifier, prompt and echo type, echo area, and status.

GDDM-GKS provides one prompt and echo type:

pet=1 The prompt and echo are provided by a cursor which is moved to the primitive to be picked. On the 3270-PC/G and /GX ranges of workstations, a square showing the pick aperture is superimposed on a cross-hair cursor. If *istat* is *GOK*, the initial cursor position is at a primitive that satisfies the initial combination of segment name and pick identifier. The echo area and data record are not used.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
37	Specified workstation is not of category OUTIN
51	Rectangle definition is invalid
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode
144	Specified prompt and echo type is not supported on this workstation
145	Echo area is outside display space
146	Contents of input data record are invalid
152	Initial value is invalid
2000	Enumeration type out of range

GINSG

Purpose

GINSG	(sgna, m)
APL code	1364
GKS RCP code	X'38003F00' (939540224)

Function: To insert segment.

Segment function. Copies the primitives of an existing segment into the current open segment or stream of primitives outside segments.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name selected when the segment was created.
m *(specified by user) (array of short floating-point numbers)*
The transformation matrix to be applied after the segment transformation. You can use either Accumulate transformation matrix (GACTM) or Evaluate transformation matrix (GEVTM) to obtain the values for this parameter.

Operating states

WSAC, SGOP

Related functions

Create segment (GCRSG), Evaluate transformation matrix (GEVTM), Accumulate transformation matrix (GACTM)

Description

This function inserts the primitives contained in segment *sgna* into the current open segment. If no segment is open, the primitives are copied into the stream of primitives outside segments. In each case, the transformed primitives are sent from WISS (workstation independent segment storage) to all active workstations.

The transformation matrix *m* is generated by calling the Evaluate transformation matrix (GEVTM) or Accumulate transformation matrix (GACTM) utility functions. The coordinates of the primitives in segment *sgna* are first transformed by the segment transformation matrix stored with them in the segment state list; then they are transformed by the matrix specified in the Insert segment (GINSG) call. Together, these matrixes form the insert transformation.

The insert transformation (conceptually) takes place in NDC space. Other than the segment transformation, segment attributes of the inserted segment are ignored. All clipping rectangles in the inserted segment are ignored. Each primitive processed is assigned a new clipping rectangle. If the clipping indicator is on, the new clipping rectangle is the viewport of the currently selected normalization transformation. If the clipping indicator is off, the new clipping rectangle is (0.0,

1.0) x (0.0, 1.0) NDC. All primitives processed by a single call to Insert segment (GINSG) receive the same clipping rectangle.

If the specified segment is not in WISS, or if it is the open segment, error 124 or 125 is reported as appropriate. When segments are inserted, the values of the primitive attributes within the inserted segments are unchanged. The values of primitive attributes used in the creation of subsequent primitives within the current open segment are unaffected.

Principal errors

5	GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
27	Workstation Independent Segment Storage is not open
120	Specified segment name is invalid
124	Specified segment does not exist on Workstation Independent Segment Storage
125	Specified segment is open

GINSK

Purpose

GINSK	**(wkid, skdnr, tnr, n, ipx, ipy, pet, xmin, xmax, ymin, ymax, buflen, ldr, datrec)**
APL code	1494
GKS RCP code	X'38004600' (939542016)

Function: To initialize stroke.

Input function. Initializes a *stroke* input device at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
skdnr *(specified by user) (fullword integer)*
The stroke device number. GDDM-GKS provides only one stroke device at a workstation. The stroke device number is always 1.
tnr *(specified by user) (fullword integer)*
The initial normalization transformation number.
n *(specified by user) (fullword integer)*
The number of points in the initial stroke. This must be less than or equal to the length of the stroke input buffer *buflen*.
ipx *(specified by user) (array of short floating-point numbers)*
ipy *(specified by user) (array of short floating-point numbers)*
The initial stroke in world coordinates.
pet *(specified by user) (fullword integer)*
The prompt and echo type.

GDDM-GKS provides prompt and echo types 1, 3, and 4.

xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The echo area in device coordinates. The echo area is not used for any of the available prompt and echo types.

buflen *(specified by user) (fullword integer)*
The length of the stroke input buffer. If this is greater than the maximum input buffer size for the device (held in the workstation description table), the maximum input buffer size is used.

ldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*. If the value specified is 0, a workstation-dependent default data record is used.

datrec *(specified by user) (array of 80-byte character tokens)*
The data record, containing additional information according to the prompt and echo type.

Data records are defined for all the stroke prompt and echo types. The Pack data record (GPRECS) function can be used to create the input data records.

Operating states

WSOP, WSAC, SGOP

Related functions

Request stroke (GRQSK), Set stroke mode (GSSKM), Inquire default stroke device data (GQDSK), Inquire stroke device state (GQSKS)

Description

This function initializes the stroke device by defining the initial stroke, normalization transformation number, prompt and echo type, echo area, and stroke data record.

The initial stroke must lie within the window of the normalization transformation specified by *tnr*. The transformation *tnr* is used to transform the initial stroke to NDC. The transformed stroke must lie within the workstation window and outside the viewport of any normalization transformation with higher priority than *tnr*. The stroke is considered to lie within a window or viewport only if all of the points in the stroke are inside. Error 152 is reported if these conditions are not met.

The initial stroke device position is at the first point of the initial stroke.

The prompt and echo types provided by GDDM-GKS are:

pet=1 Display a line joining successive points of the current stroke. The echo area is not used. The data record is used only at 3270-PC/G and /GX ranges, and IBM 5080 workstations; it contains elements for the x,y interval and time interval. If these elements are all zero or the data record is not used, points in the stroke are generated when the operator moves the cursor and activates the device by pressing a mouse or tablet button, or the Enter key. If the data record is used and any of x,y interval or time interval are not zero, points are generated automatically as the cursor is moved, tracing an outline of the cursor path.

The data record can be created by calling Pack data record (GPRECS) with the following parameters:

```
il   = 0
ia   = ()
rl   = 3
ra   = (x-interval,
        y-interval,
        time interval in seconds)
sl   = 0
lstr = ()
str  = ()
```

pet=3 Display a marker at each point of the current stroke.

The echo area is not used. The data record contents indicate whether the current *polymarker* attributes, or attributes specified in the data record, should be used to draw the markers.

If the current attributes are to be used, the parameters of the Pack data record (GPRECS) function are:

```
il   = 1
ia   = (attribute control flag=GCURNT)
rl   = 3
ra   = (x-interval,
        y-interval,
        time interval in seconds)
sl   = 0
lstr = ()
str  = ()
```

If attributes are to be specified in the data record, the parameters of the Pack data record (GPRECS) function are:

```
il   = 7
ia   = (attribute control flag=GSPEC,
        marker type aspect source flag (ASF),
        marker size scale factor ASF,
        polymarker color index ASF,
        polymarker index,
        marker type,
        polymarker color index)
rl   = 4
ra   = (x-interval,
        y-interval,
        time interval in seconds,
        marker size scale factor)
sl   = 0
lstr = ()
str  = ()
```

The data record is not honored at 3270-PC/G and /GX ranges, and IBM 5080 workstations. The x,y interval and time interval elements are not used at any workstation.

pet=4 Display a line joining successive points of the current stroke.

The echo area is not used. The data record contents indicate whether the current *polyline* attributes, or attributes specified in the data record, should be used to draw the connecting lines.

If the current attributes are to be used, the parameters of the Pack data record (GPRECS) function are:

```
il   = 1
ia   = (attribute control flag=GCURNT)
rl   = 3
ra   = (x-interval,
        y-interval,
        time interval in seconds)
sl   = 0
lstr = ()
str  = ()
```

If attributes are to be specified in the data record, the parameters of the Pack data record (GPRECS) function are:

```
il   = 7
ia   = (attribute control flag=GSPEC),
         linetype ASF,
         linewidth scale factor ASF,
         polyline color index ASF,
         polyline index,
         linetype,
         polyline color index)
rl   = 4
ra   = (x-interval,
        y-interval,
        time interval in seconds,
        linewidth scale factor)
sl   = 0
lstr = ()
str  = ()
```

The attribute information in the data record is not honored on 3270-PC/G and /GX ranges, and IBM 5080 workstations. For these workstations, the x,y interval and time interval elements are processed as described for *pet*=1. The elements are not used at other workstations.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
51	Rectangle definition is invalid
60	Polyline index is invalid
63	Linetype is equal to zero
65	Linewidth scale factor is less than zero
66	Polymarker index is invalid
67	A representation for the specified polymarker index has not been defined on this workstation
69	Marker type is equal to zero
92	Color index is less than zero
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode
144	Specified prompt and echo type is not supported on this workstation
145	Echo area is outside display space
146	Contents of input data record are invalid
152	Initial value is invalid

153	Number of points in the initial stroke is greater than the buffer size
2000	Enumeration type out of range

GINST

Purpose

GINST	**(wkid,stdnr,lstr,istr,pet,xmin, xmax,ymin,ymax,buflen, inipos,ldr,datrec)**

Function: Input function. Initializes a *string* input device at a workstation. Use this call only if your program is written in FORTRAN IV or VS FORTRAN. Otherwise, use the function Initialize string (GINSTS) instead.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.

stdnr *(specified by user) (fullword integer)*
The string device number. GDDM-GKS provides only one string device at a workstation. The string device number is always 1.

lstr *(specified by user) (fullword integer)*
The length of the initial string. This must be less than or equal to the length of the string input buffer *buflen*. The actual value used is the lesser of *lstr* and the length of the initial string *istr*.

istr *(specified by user) (character)*
The initial string to be displayed. In a VS FORTRAN program, the string can be of variable length. For FORTRAN IV, you must define the string as CHARACTER*80.

pet *(specified by user) (fullword integer)*
The prompt and echo type. GDDM-GKS provides only one prompt and echo type; this parameter should always equal 1.

xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
Echo area in device coordinates.

buflen *(specified by user) (fullword integer)*
The length of the string input buffer.

inipos *(specified by user) (fullword integer)*
Initial cursor position. It may range from 1 to the length of the initial string plus 1.

ldr *(specified by user) (fullword integer)*
Dimension of the data record array *datrec*.

datrec *(specified by user) (array of 80-byte character tokens)*
The data record.

Operating states

WSOP, WSAC, SGOP

Related functions

Request string (FORTRAN only) (GRQST), Set string mode (GSSTM)

Description

This function initializes the string device by defining the initial string, prompt and echo type, and echo area.

The function copies the initial string into the input buffer and places the cursor at the initial cursor position. Replacement of characters begins at this point.

The input buffer size is compared with the "maximum input buffer size" for the device, which is contained in the workstation description table. If the requested buffer size is greater than the maximum, the maximum buffer size is substituted. If you define an initial string longer than the buffer size, error 154 is reported. GKS defines the initial cursor position parameter, *inipos*, as a (mandatory) part of the data record for the string device. If *inipos* is invalid, error 152 is reported.

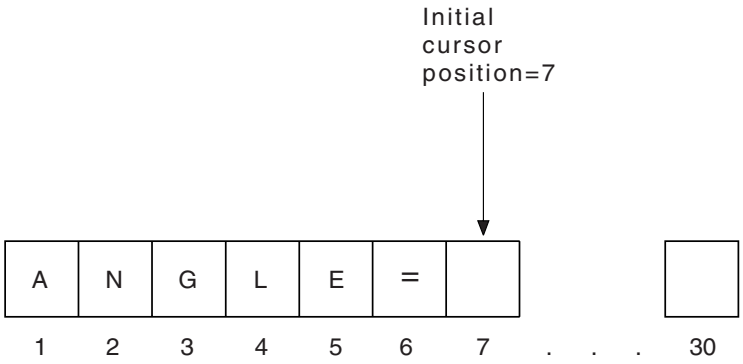
GDDM-GKS provides one prompt and echo type:

pet=1 Echo string within echo area. The string is echoed starting at the lower left corner of the echo area. No clipping to the echo area limits is performed. The data record is not used.

For example, you can use a string input device to obtain a rotation angle for a segment transformation. The initial string is "angle=," which is six characters long. The initial cursor position is set to character "7" so that the initial string is not overwritten. The echo area is defined using the maximum X and Y values for DC as returned by Inquire Maximum Display Surface Size.

```
 stdnr=1
 lstr=6
 istr='angle='
 pet=1
 xmin = 0.0
 xmax = xdc
 ymin = ydc/10.0
 ymax = ydc
 buflen = 30
 inipos = 7
 ldr=0
 CALL GINST (wkid, stdnr, lstr, istr, pet,
*xmin, xmax, ymin, ymax, buflen, inipos,
*ldr, datrec)
```

This illustration shows the input buffer containing the initial string, and indicates the location of the initial cursor position.



Principal errors

```
  7  GKS not in proper state: GKS shall be in one of the states
     WSOP, WSAC, or SGOP
 20  Specified workstation identifier is invalid
 25  Specified workstation is not open
 38  Specified workstation is neither of category INPUT nor of
     category OUTIN
 51  Rectangle definition is invalid
140  Specified input device is not present on workstation
141  Input device is not in REQUEST mode
144  Specified prompt and echo type is not supported on this
     workstation
145  Echo area is outside display space
146  Contents of input data record are invalid
152  Initial value is invalid
154  Length of the initial string is greater than the buffer size
```

GINSTS

Purpose

GINSTS	**(wkid, stdnr, lstr, istr, pet, xmin, xmax, ymin, ymax, buflen, inipos, ldr, datrec)**
APL code	1498
GKS RCP code	X'38004A00' (939543040)

Function: To initialize string.

Input function. Initializes a *string* input device at a workstation. If your program is written in FORTRAN IV or VS FORTRAN, use the function Initialize string (FORTRAN only) (GINST) instead.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
stdnr *(specified by user) (fullword integer)*
The string device number. GDDM-GKS provides only one string device at a workstation. The string device number is always 1.
lstr *(specified by user) (fullword integer)*
The length of the initial string. This must be less than or equal to the length of the string input buffer *buflen*.
istr *(specified by user) (character)*
The initial string to be displayed.
pet *(specified by user) (fullword integer)*
The prompt and echo type. GDDM-GKS provides only one prompt and echo type; this parameter should always equal 1.
xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
Echo area in device coordinates.
buflen *(specified by user) (fullword integer)*
The length of the string input buffer.
inipos *(specified by user) (fullword integer)*
Initial cursor position. It may range from 1 to the length of the initial string plus 1.
ldr *(specified by user) (fullword integer)*
Dimension of the data record array *datrec*.
datrec *(specified by user) (array of 80-byte character tokens)*
The data record.

Operating states

WSOP, WSAC, SGOP

Related functions

Request string (GRQSTS), Set string mode (GSSTM)

Description

This function initializes the string device by defining the initial string, prompt and echo type, and echo area.

The function copies the initial string into the input buffer and places the cursor at the initial cursor position. Replacement of characters begins at this point.

The input buffer size is compared with the “maximum input buffer size” for the device, which is contained in the workstation description table. If the requested buffer size is greater than the maximum, the maximum buffer size is substituted. If you define an initial string longer than the buffer size, error 154 is reported. If *inipos* is invalid, error 152 is reported.

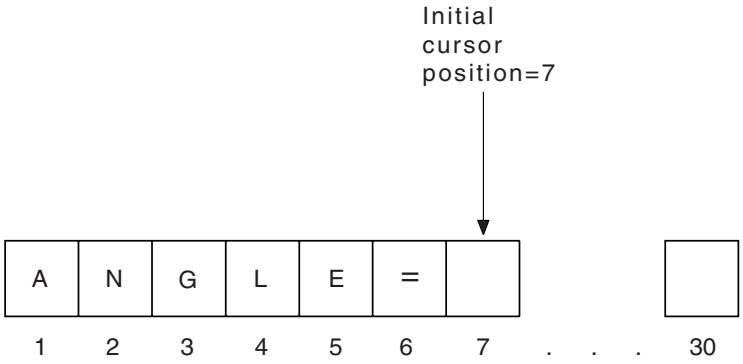
GDDM-GKS provides one prompt and echo type:

pet=1 Echo string within echo area. The string is echoed starting at the lower left corner of the echo area. No clipping to the echo area limits is performed. The data record is not used.

For example, you can use a string input device to obtain a rotation angle for a segment transformation. The initial string is “angle=,” which is six characters long. The initial cursor position is set to character “7” so that the initial string is not overwritten. The echo area is defined using the maximum X and Y values for DC as returned by the function Inquire display space size (GQDSP).

```
stdnr=1
lstr=6
istr='angle='
pet=1
xmin = 0.0
xmax = xdc
ymin = ydc/10.0
ymax = ydc
buflen = 30
inipos = 7
ldr=0
CALL GINSTS (wkid, stdnr, lstr, istr, pet,
*xmin, xmax, ymin, ymax, buflen, inipos,
*ldr, datrec)
```

This illustration shows the input buffer containing the initial string, and indicates the location of the initial cursor position.



Principal errors

	None
7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
51	Rectangle definition is invalid
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode
144	Specified prompt and echo type is not supported on this workstation
145	Echo area is outside display space
146	Contents of input data record are invalid
152	Initial value is invalid
154	Length of the initial string is greater than the buffer size

GINVL

Purpose

GINVL	(wkid, vldnr, ival, pet, xmin, xmax, ymin, ymax, loval, hival, ldr, datrec)
APL code	1495
GKS RCP code	X'38004700' (939542272)

Function: To initialize valuator.

Input function. Initializes a *valuator* input device at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
vldnr *(specified by user) (fullword integer)*
The valuator device number. GDDM-GKS provides only one valuator device at a workstation. The valuator device number is always 1.
ival *(specified by user) (short floating point)*
The initial value of the valuator device. This value must lie in the range set by the minimum and maximum values, *loval* and *hival*.
pet *(specified by user) (fullword integer)*
The prompt and echo type.
xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The echo area in device coordinates.
loval *(specified by user) (short floating point)*
hival *(specified by user) (short floating point)*
The minimum and maximum values defining the valuator range. The maximum value, *hival*, must be greater than the minimum value; *loval*; otherwise, error 146 is reported.
ldr *(specified by user) (fullword integer)*
The dimension of the data record array.
datrec *(specified by user) (array of 80-byte character tokens)*
The valuator data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Request valuator (GRQVL), Set valuator mode (GSVLM)

Description

This function initializes the valuator device by defining the initial value, prompt and echo type, echo area, and range of values.

GDDM-GKS provides one prompt and echo type:

pet=1 Echo the current valuator value within the echo area.

The initial value is displayed as a string starting in the lower left corner of the echo area. The value can be overtyped by the operator with a number within the range *loval* through *hival*.

The data record is not used.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
51	Rectangle definition is invalid
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode
144	Specified prompt and echo type is not supported on this workstation
145	Echo area is outside display space
146	Contents of input data record are invalid
152	Initial value is invalid

GMSG

Purpose

GMSG **(wkid,mess)**

Function: Control function. Displays a message at a workstation. Use this call only if your program is written in VS FORTRAN. Otherwise, use the function Message (GMSGS) instead.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier, selected when the workstation was opened.

mess *(specified by user) (character)*
The message string to be displayed.

Operating states

WSOP, WSAC, SGOP

Related functions

None

Description

This function displays the message *mess*. If the workstation *wkid* is a plotter, a printer, a GDF-file workstation, or a display, the message is displayed at the user console. If the workstation is a metafile output workstation, a MESSAGE metafile item is written to the metafile.

This function cannot send a message to workstation independent segment storage (WISS). Messages sent to metafile input workstations are ignored.

The call does not affect the GKS state list or the execution of subsequent GKS functions. When a message is displayed at the user console, the screen is temporarily cleared while the message is shown.

If *lstr* is less than 0, GDDM-GKS will report error 2001.

Principal errors

```
   7  GKS not in proper state: GKS shall be in one of the states
      WSOP, WSAC, or SGOP
  20  Specified workstation identifier is invalid
  25  Specified workstation is not open
  36  Specified workstation is Workstation Independent Segment
      Storage
2001  Output parameter size insufficient
```

GMSGS

Purpose

GMSGS	(wkid, lstr, mess)
APL code	1490
GKS RCP code	X'38000A00' (939526656)

Function: To message.

Control function. Displays a message at a workstation. If your program is written in VS FORTRAN, use the function Message (VS FORTRAN only) (GMSG) instead.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier, selected when the workstation was opened.
lstr *(specified by user) (fullword integer)*
The number of characters in *mess*.
mess *(specified by user) (character)*
The message string to be displayed.

Operating states

WSOP, WSAC, SGOP

Related functions

None

Description

This function displays the message *mess*. If the workstation *wkid* is a plotter, a printer, a GDF-file workstation, or a display, the message is displayed at the user console. If the workstation is a metafile output workstation, a MESSAGE metafile item is written to the metafile.

This function cannot send a message to workstation independent segment storage (WISS). Messages sent to metafile input workstations are ignored.

The call does not affect the GKS state list or the execution of subsequent GKS functions. When a message is displayed at the user console, the screen is temporarily cleared while the message is shown.

If *lstr* is less than 0, GDDM-GKS will report error 2001.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
36	Specified workstation is Workstation Independent Segment Storage
2001	Output parameter size insufficient

GOPKS

Purpose

GOPKS	**(errfil, bufa)**
APL code	1300
GKS RCP code	X'38000000' (939524096)

Function: To open GKS.

Control function. Opens GKS, and opens an error file for GKS error messages.

Parameters

errfil *(specified by user) (fullword integer)*
The error message file. This parameter is an integer in the range 1 through 9999 and identifies the destination of error messages output when the Error logging (GERLOG) function is called.

bufa *(specified by user) (fullword integer)*
Number of memory units for buffer area. Not used in GDDM-GKS; it should be set to 0.

Operating states

GKCL

Related functions

Close GKS (GCLKS), Emergency close GKS (GECLKS)

Description

This function opens and initializes GKS. The GKS state list is allocated and initialized. The GKS description table and the workstation description tables are made available.

The "error file" entry in the GKS error state list is set to the value specified by *errfil* and the error file is opened. The name and characteristics of the error file opened are subsystem dependent. On VM/CMS, the file ADMJnnnn ADMERLOG is created (nnnn represents the value supplied in *errfil*. On other subsystems, an appropriate equivalent file is created, as described in "Using GDDM-GKS under various subsystems" on page 60.

This call sets the GKS operating state to GKOP. It should be the first GKS function invoked by your application.

Principal errors

1	GKS not in proper state: GKS shall be in the state GKCL
200	Specified error file is invalid

GOPWK

Purpose

GOPWK	**(wkid, conid, wtype)**
APL code	1302
GKS RCP code	X'38000200' (939524608)

Function: To open workstation.

Control function. Opens a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
A unique identifier that you assign to the workstation. Use this identifier on subsequent function calls that require a workstation identifier.

conid *(specified by user) (fullword integer)*
An integer in the range 0 through 9999 indicating the device to be opened. A unique connection identifier must be given for each graphics device opened.

The value given is interpreted according to the specified workstation type as follows:

Default workstation If the workstation type parameter is 1, any valid connection identifier can be given. The default workstation, normally the user console, is opened.

WISS If the workstation type parameter is 2 (WISS), the connection identifier is ignored.

Metafile input/output For MO and MI workstations (workstation types 3 and 4), the value given identifies the file to be used.

GDF file output For GDF file workstations (workstation type 5), the value given identifies the file to be used.

Other GDDM devices If the workstation type parameter is in the range 6 through 13, GDDM-GKS searches for the entry GKWSnnnn in the nicknames defined for the user, where nnnn is the value specified by *conid*.

wtype *(specified by user) (fullword integer)*
An integer in the range 1 through 13 indicating the type of the workstation to be opened. The possible values are:

1 The default workstation (normally the user console)
2 WISS
3 Metafile output workstation
4 Metafile input workstation
5 GDF file workstation
6-13 Other supported graphics devices

The workstation types available are those that have been specified using the external default GKSWS.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Close workstation (GCLWK), Activate workstation (GACWK)

Description

This function opens a specified workstation by establishing a connection to the device, and by allocating and initializing a workstation state list for it. The workstation identifier is added to the list of open workstations held in the GKS state list. If GKS is in operating state GKOP, it is set into the state WSOP (at least one workstation open).

The workstation type and connection identifier (*wkid* and *conid*) determine which logical device will be used. See “Using GDDM-GKS workstations” on page 50 for a detailed description of workstation types and the processing of the *conid* parameter.

The workstation identifier is subsequently used in other functions to refer to the workstation.

GKS allows a maximum of five open workstations at any time; only one WISS (workstation independent segment storage) can be open.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
21	Specified connection identifier is invalid
22	Specified workstation type is invalid
23	Specified workstation type does not exist
24	Specified workstation is open
26	Specified workstation cannot be opened
28	Workstation Independent Segment Storage is already open
42	Maximum number of simultaneously open workstations would be exceeded.

GPL

Purpose

GPL	(n, px, py)
APL code	1313
GKS RCP code	X'38000C00' (939527168)

Function: To polyline.

Output function. Draws a sequence of connected straight lines between points.

Parameters

n *(specified by user) (fullword integer)*
The number of points.
px *(specified by user) (array of short floating-point numbers)*
py *(specified by user) (array of short floating-point numbers)*
The coordinates of the *n* points to be connected, in world coordinates.

Operating states

WSAC, SGOP

Related functions

Set linetype (GSLN), Set linewidth scale factor (GSLWSC), Set polyline color index (GSPLCI), Set polyline index (GSPLI), Set polyline representation (GSPLR), Set aspect source flags (GSASF)

Description

This function draws a sequence of connected straight lines, beginning with the first point and ending with the last point, given by the arrays *px* and *py*.

The current values of the *polyline* attributes, as given by the GKS state list, are bound to the polyline primitive.

If the line type is not *solid*, the line type starts at the start of the polyline and is not restarted at points in the polyline, or when clipping occurs.

Principal errors

5	GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
100	Number of points is invalid

GPM

Purpose

GPM	**(n, px, py)**
APL code	1314
GKS RCP code	X'38000D00' (939527424)

Function: To polymarker.

Output function. Draws a sequence of markers at given positions.

Parameters

n *(specified by user) (fullword integer)*
The number of markers.
px *(specified by user) (array of short floating-point numbers)*
py *(specified by user) (array of short floating-point numbers)*
The coordinates of the *n* markers, in world coordinates.

Operating states

WSAC, SGOP

Related functions

Set marker type (GSMK), Set marker size scale factor (GSMKSC), Set polymarker color index (GSPMCI), Set polymarker index (GSPMI), Set aspect source flags (GSASF)

Description

This function displays a marker at each of the points (*px, py*). The current values of the *polymarker* attributes, as given by the GKS state list, are bound to the polymarker primitive.

Principal errors

5	GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
100	Number of points is invalid

GPREC

Purpose

GPREC	**(il,ia,rl,ra,sl,lstr,str,mldr, errind,ldr,datrec)**

Function: Utility function. Packs integer, real, and character data into a data record. Use this call only if your program is written in FORTRAN IV or VS FORTRAN. Otherwise, use the function Pack data record (GPRECS) instead.

Parameters

il *(specified by user) (fullword integer)*
The number of integer entries in *ia*.

ia *(specified by user) (an array of fullword integers)*
The integer array.

rl *(specified by user) (fullword integer)*
The number of floating-point entries in *ra*.

ra *(specified by user) (array of short floating-point numbers)*
The floating-point array.

sl *(specified by user) (fullword integer)*
The number of character-string entries in *str*.

lstr *(specified by user) (an array of fullword integers)*
The individual lengths of the strings in the *str* parameter.

str *(specified by user) (character)*
The string array. In a VS FORTRAN program, the strings can be of variable length. For FORTRAN IV, you must define the strings as CHARACTER*80.

mldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.

errind *(returned by GDDM) (fullword integer)*
The error indicator.

ldr *(returned by GDDM) (fullword integer)*
The number of array elements used in the *datrec* parameter.

datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Related functions

Initialize locator (GINLC), Initialize stroke (GINSK)

Description

This function returns a packed data record when you submit the individual unpacked elements of the record. You can use this function to create data records required for initialization of input devices.

If the data record cannot be created, the error number is returned in *errind*. Error 2003 is reported if a problem is detected while GDDM-GKS is creating the data record, making the data record invalid.

Principal errors

```
2001  Output parameter size insufficient
2003  Invalid data record
```

GPRECS

Purpose

GPRECS	**(il, ia, rl, ra, sl, mstr, lstr, str, mldr, errind, ldr, datrec)**
APL code	1513
GKS RCP code	X'38006B00' (939551488)

Function: To pack data record.

Utility function. Packs integer, real, and character data into a data record. If your program is written in FORTRAN IV or VS FORTRAN, use the function Pack data record (FORTRAN only) (GPREC) instead.

Parameters

il *(specified by user) (fullword integer)*
The number of integer entries in *ia*.
ia *(specified by user) (an array of fullword integers)*
The integer array.
rl *(specified by user) (fullword integer)*
The number of floating-point entries in *ra*.
ra *(specified by user) (array of short floating-point numbers)*
The floating-point array.
sl *(specified by user) (fullword integer)*
The number of character-string entries in *str*.
mstr *(specified by user) (fullword integer)*
The maximum length of the strings in the *str* parameter.
lstr *(specified by user) (an array of fullword integers)*
The individual lengths of the strings in the *str* parameter.
str *(specified by user) (character)*
The string array.
mldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ldr *(returned by GDDM) (fullword integer)*
The number of array elements used in the *datrec* parameter.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Related functions

Initialize locator (GINLC), Initialize stroke (GINSK)

Description

This function returns a packed data record when you submit the individual unpacked elements of the record. You can use this function to create data records required for initialization of input devices.

If the data record cannot be created, the error number is returned in *errind*. Error 2003 is reported if a problem is detected while GDDM-GKS is creating the data record, making the data record invalid.

Principal errors

2001	Output parameter size insufficient
2003	Invalid data record

GQACWK

Purpose

GQACWK	(n, errind, ol, wkid)
APL code	1422
GKS RCP code	X'38008600' (939558400)

Function: To inquire set member of active workstations.

Inquiry function. Returns the number of workstations in the set of active workstations, and the identifier of a workstation in the set.

Parameters

n *(specified by user) (fullword integer)*
The set member requested. If the value specified is 0, the number of active workstations is returned but *wkid* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of active workstations.
wkid *(returned by GDDM) (fullword integer)*
Member *n* of the set of active workstations.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Activate workstation (GACWK), Deactivate workstation (GDAWK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and **errind** returns an error indicator.

If *n* is less than zero or greater than the number of active workstations, error 2002 is returned, unless the set of active workstations is empty. If the set is empty, *ol* is returned as 0.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
2002	List element or set member not available

GQASF

Purpose

GQASF	(errind, lasf)
APL code	1460
GKS RCP code	X'3800A100' (939565312)

Function: To inquire aspect source flags.

Inquiry function. Returns the current values of the aspect source flags.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.

lasf *(returned by GDDM) (an array of fullword integers)*
The aspect source flag settings. This is an array of thirteen integers. Their possible values are:

0 (GBUNDL) Bundled
1 (GINDIV) Individual

The elements of the array of aspect source flags are in the following order:

Line type
Line-width scale factor
Polyline color index
Marker type
Marker size scale factor
Polymarker color index
Text font and precision
Character expansion factor
Character spacing
Text color index
Fill area interior style
Fill area style index
Fill area color index

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set aspect source flags (GSASF)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQASWK

Purpose

GQASWK	(sgna, n, errind, ol, wkid)
APL code	1429
GKS RCP code	X'3800DE00' (939580928)

Function: To inquire set member of associated workstations.

Inquiry function. Returns the name of a workstation associated with a segment.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name.
n *(specified by user) (fullword integer)*
The set member requested. If the value specified is 0, the number of associated workstations is returned but *wkid* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of associated workstations.
wkid *(returned by GDDM) (fullword integer)*
Member *n* of the set of associated workstations.

Operating states

WSOP, WSAC, SGOP

Related functions

Associate segment with workstation (GASGWK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

You can obtain the set of all workstations associated with a segment by using the logic of the following example. Make an initial call to GQASWK with $n = 0$. The output includes the total number of associated workstations (*ol*). Now make repeated calls until you have accumulated set members equal to the total.

Here is an example:

```
      n=0
      CALL GQASWK (sgna, n, errind, ol, wkid)
      DO  55 n = 1, ol
      CALL GQASWK (sgna, n, errind, ol, wkid)
      WRITE ( *, 50) wkid
50    FORMAT (' wkid', I6)
55    CONTINUE
```

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
120	Specified segment name is invalid
122	Specified segment does not exist
2002	List element or set member not available

GQCF

Purpose

GQCF	**(wtype, errind, ncoli, cola, npci)**
APL code	1415
GKS RCP code	X'3800D000' (939577344)

Function: To inquire color facilities.

Inquiry function. Returns values giving the color facilities for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ncoli *(returned by GDDM) (fullword integer)*
The number of colors or intensities (0,2 ... n).
cola *(returned by GDDM) (fullword integer)*
The availability of color. The possible values are:
0 (GMONOC) Monochrome
1 (GCOLOR) Color
npci *(returned by GDDM) (fullword integer)*
The number of predefined color indexes (2 ... n).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set color representation (GSCR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If the number of available colors or intensities is returned as 0 (zero), the workstation supports a continuous range of colors or intensities.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQCHB

Purpose

GQCHB	**(errind, chbx, chby)**
APL code	1439
GKS RCP code	X'38008D00' (939560192)

Function: To inquire character base vector.

Inquiry function. Returns the current character base vector from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
chbx *(returned by GDDM) (short floating point)*
chby *(returned by GDDM) (short floating point)*
The character base vector in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set character up vector (GSCHUP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The character base vector is a vector giving the direction of the baseline of a character. It is set implicitly by the Set character up vector (GSCHUP) function.

Principal errors

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

GQCHH

Purpose

GQCHH	**(errind, chh)**
APL code	1431
GKS RCP code	X'38008A00' (939559424)

Function: To inquire character height.

Inquiry function. Returns the current character height from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
chh *(returned by GDDM) (short floating point)*
The current character height in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set character height (GSCHH)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQCHS

Purpose

GQCHS	**(wkid, chdnr, mldr, errind, mode, esw, istat, ichnr, pet, earea, ldr, datrec)**
APL code	1487
GKS RCP code	X'3800BE00' (939572736)

Function: To inquire choice device state.

Inquiry function. Returns the state of a *choice* device at a workstation, from the workstation state list. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
chdnr *(specified by user) (fullword integer)*
The choice device number (1 ... n).
mldr *(specified by user) (fullword integer)*
The dimension of the data record array.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
mode *(returned by GDDM) (fullword integer)*
The operating mode. The only possible value is:
0 (GREQU) Request
esw *(returned by GDDM) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo
istat *(returned by GDDM) (fullword integer)*
Initial status. The possible values are:
1 (GOK) OK
2 (GNCHOI) No choice
ichnr *(returned by GDDM) (fullword integer)*
The initial choice number.
pet *(returned by GDDM) (fullword integer)*
The prompt and echo type (always 1 in GDDM-GKS).

earea *(returned by GDDM) (array of short floating-point numbers)*
The echo area used in device coordinates (XMIN, XMAX, YMIN, YMAX).
ldr *(returned by GDDM) (fullword integer)*
The number of array elements used in the data record array.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize choice (GINCH)

Description

(See the section "GDDM-GKS restrictions" below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS reports error 2001 in *errind*.

GDDM-GKS Restrictions

GDDM-GKS ignores the *mldr* parameter; the data record returned is empty.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2001	Output parameter size insufficient

GQCHSP

Purpose

GQCHSP	(errind, chsp)
APL code	1455
GKS RCP code	X'38009C00' (939564032)

Function: To inquire character spacing.

Inquiry function. Returns the current character spacing from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
chsp *(returned by GDDM) (short floating point)*
The spacing between characters, defined as a fraction of the nominal character height.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set character spacing (GSCHSP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQCHUP

Purpose

GQCHUP	**(errind, chux, chuy)**
APL code	1433
GKS RCP code	X'38008B00' (939559680)

Function: To inquire character up vector.

Inquiry function. Returns the current character up vector from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
chux *(returned by GDDM) (short floating point)*
chuy *(returned by GDDM) (short floating point)*
The current character up vector in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set character up vector (GSCHUP)

Description

If the inquired information is available, it is returned as output, and errind is returned as 0 (zero). If the inquired information is not available, all output is invalid, and errind returns an error indicator.

Principal errors

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

GQCHW

Purpose

GQCHW	(errind, chw)
APL code	1438
GKS RCP code	X'38008C00' (939559936)

Function: To inquire character width.

Inquiry function. Returns the current character width from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
chw *(returned by GDDM) (short floating point)*
The current character width in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set character height (GSCHH)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind*

Principal errors

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

GQCHXP

Purpose

GQCHXP	(errind, chxp)
APL code	1454
GKS RCP code	X'38009B00' (939563776)

Function: To inquire character expansion factor.

Inquiry functions. Returns the current character expansion factor from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
chxp *(returned by GDDM) (short floating point)*
The character expansion factor.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set character expansion factor (GSCHXP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQCLIP

Purpose

GQCLIP	(errind, clsw, clrect)
APL code	1464
GKS RCP code	X'3800A500' (939566336)

Function: To inquire clipping indicator.

Inquiry function. Returns the current clipping indicator from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
clsw *(returned by GDDM) (fullword integer)*
The current setting of the clipping indicator. The possible values are:
0 (GNCLIP) No clipping
1 (GCLIP) Clipping
clrect *(returned by GDDM) (array of short floating-point numbers)*
The clipping rectangle. This parameter returns the clipping rectangle boundaries in normalized device coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set clipping indicator (GSCLIP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQCNTN

Purpose

GQCNTN	(errind, ctnr)
APL code	1461
GKS RCP code	X'3800A200' (939565568)

Function: To inquire current normalization transformation number.

Inquiry function. Returns the current normalization transformation number from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
ctnr *(returned by GDDM) (fullword integer)*
The current normalization transformation number.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Select normalization transformation (GSELNT)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQCR

Purpose

GQCR	(wkid, coli, type, errind, cr, cg, cb)
APL code	1482
GKS RCP code	X'3800B800' (939571200)

Function: To inquire color representation.

Inquiry function. Returns the color representation values of a color table entry in a workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
coli *(specified by user) (fullword integer)*
The color index.
type *(specified by user) (fullword integer)*
The type of returned values. The possible values are:
0 (GSET) Set
1 (GREALI) Realized
errind *(returned by GDDM) (fullword integer)*
The error indicator.
cr *(returned by GDDM) (short floating point)*
The intensity of red color (0 to 1).
cg *(returned by GDDM) (short floating point)*
The intensity of green color (0 to 1).
cb *(returned by GDDM) (short floating point)*
The intensity of blue color (0 to 1).

Operating states

WSOP, WSAC, SGOP

Related functions

Set color representation (GSCR)

Description

The color representation in red, green, and blue intensities for the color index *coli* is returned.

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If you request set values (*type* = 0), GQCR returns the values you set when you defined the color associated with color index *coli*.

If you request realized values (*type* = 1), GQCR returns the closest approximation to the set values that the device can support.

If the device is capable of representing the color you defined, set values and realized values are the same.

You can find out which color indexes are in use by calling Inquire list element of color indexes (GQECI).

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
93	Color index is invalid
94	A representation for the specified color index has not been defined on this workstation

GQDCH

Purpose

GQDCH	**(wtype, devno, n, mldr, errind, malt, ol, pet, earea, ldr, datrec)**
APL code	1503
GKS RCP code	X'3800DB00' (939580160)

Function: To inquire default choice device data.

Inquiry function. Returns the default data for a *choice* device at a workstation, from the workstation description table. (See the section “GDDM-GKS restrictions” below.)

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
devno *(specified by user) (fullword integer)*
The choice device number.
n *(specified by user) (fullword integer)*
The element requested from the list of available prompt and echo types. If the value specified is 0, the number of available prompt and echo types is returned in *ol* but *pet* is undefined.
mldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*
errind *(returned by GDDM) (fullword integer)*
The error indicator
malt *(returned by GDDM) (fullword integer)*
The maximum number of alternatives. For some choice devices, for example, the Enter key, light pen, and tablet stylus, the value returned is 1; for these devices, if the operator triggers the device during a Request choice (GRQCH) operation, the choice number returned is always 0.
ol *(returned by GDDM) (fullword integer)*
The number of available prompt and echo types.

pet *(returned by GDDM) (fullword integer)*
Element *n* of the list of available prompt and echo types.
earea *(returned by GDDM) (array of short floating-point numbers)*
The echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).
ldr *(returned by GDDM) (fullword integer)*
The number of elements used in the data record array.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Initialize choice (GINCH)

Description

(See the section "GDDM-GKS restrictions" below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS reports error 2001 in *errind*.

GDDM-GKS Restrictions

GDDM-GKS ignores the *mldr* parameter; the data record returned is empty.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2001	Output parameter size insufficient
2002	List element or set member not available

GQDDS

Purpose

GQDDS	(wtype, errind, defmod, regmod)
APL code	1394
GKS RCP code	X'3800C500' (939574528)

Function: To inquire default deferral state values.

Inquiry function. Returns the default values for the deferral state and implicit regeneration mode for a workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
defmod *(returned by GDDM) (fullword integer)*
The default value for deferral mode. The possible values are:
0 (GASAP) As soon as possible
1 (GBNIG) Before the next possible interaction globally
2 (GBNIL) Before the next possible interaction locally
3 (GASTI) At some time
regmod *(returned by GDDM) (fullword integer)*
The default value for implicit regeneration mode. The possible values are:
0 (GSUPPD) Suppressed
1 (GALLOW) Allowed

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

None.

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQDLC

Purpose

GQDLC	(wtype, devno, n, mldr, errind, dpx, dpy, ol, pet, earea, ldr, datrec)
APL code	1512
GKS RCP code	X'3800D800' (939579392)

Function: To inquire default locator device data.

Inquiry function. Returns the default data for a *locator* device at a workstation, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
devno *(specified by user) (fullword integer)*
The logical input device number.
n *(specified by user) (fullword integer)*
The element requested from the list of available prompt and echo types. If the value specified is 0, the number of available prompt and echo types is returned in *ol* but *pet* is undefined.
mldr *(specified by user) (fullword integer)*
Dimension of array *datrec*.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
dpx *(returned by GDDM) (short floating point)*
dpy *(returned by GDDM) (short floating point)*
Default initial position of locator in world coordinates.
ol *(returned by GDDM) (fullword integer)*
The number of available prompt and echo types (always 1 in GDDM-GKS).
pet *(returned by GDDM) (fullword integer)*
Element *n* of the list of available prompt and echo types.
earea *(returned by GDDM) (array of short floating-point numbers)*
The default echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).
ldr *(returned by GDDM) (fullword integer)*
The number of elements used in the data record array.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Unpack data record (GURECS), Initialize locator (GINLC)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The Unpack data record (GURECS) utility is available to unpack the data record returned by GQDLC.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS reports error 2001 in *errind*. Usually, a maximum length of 80 (*mldr*=1) is sufficient.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2001	Output parameter size insufficient
2002	List element or set member not available

GQDPK

Purpose

GQDPK	**(wtype, devno, n, mldr, errind, ol, pet, earea, ldr, datrec)**
APL code	1505
GKS RCP code	X'3800DC00' (939580416)

Function: To inquire default pick device data.

Inquiry function. Returns the default data for a *pick* device at a workstation, from the workstation description table. (See the section “GDDM-GKS restrictions” below.)

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
devno *(specified by user) (fullword integer)*
The pick device number.
n *(specified by user) (fullword integer)*
The element requested from the list of available prompt and echo types.

mldr *(specified by user) (fullword integer)*
The dimension of the data record array.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of available prompt and echo types.
pet *(returned by GDDM) (fullword integer)*
Element *n* of the list of available prompt and echo types.
earea *(returned by GDDM) (array of short floating-point numbers)*
The default echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).
ldr *(returned by GDDM) (fullword integer)*
The number of elements used in the data record array.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Initialize pick (GINPK)

Description

(See the section “GDDM-GKS restrictions” below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If the list element requested, *n*, is 0, *ol* is set to the number of available prompt and echo types but *pet* is undefined. If the list element is out of range, GKS reports error 2002 in *errind*.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS reports error 2001 in *errind*.

GDDM-GKS Restrictions

GDDM-GKS ignores the *mldr* parameter; the data record returned is empty.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2001	Output parameter size insufficient
2002	List element or set member not available

GQDSGA

Purpose

GQDSGA	(wtype, errind, sgtr, vonoff, voffon, high, sgpr, add, sgdel)
APL code	1425
GKS RCP code	X'3800D600' (939578880)

Function: To inquire dynamic modification of segment attributes.

Inquiry function. Returns dynamic modification information for segment attributes for a workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
sgtr *(returned by GDDM) (fullword integer)*
Segment transformation changes. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate
vonoff *(returned by GDDM) (fullword integer)*
Visibility from on to off. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate
voffon *(returned by GDDM) (fullword integer)*
Visibility from off to on. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate
high *(returned by GDDM) (fullword integer)*
Changes in highlighting. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate
sgpr *(returned by GDDM) (fullword integer)*
Changes in segment priority. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate
add *(returned by GDDM) (fullword integer)*
Primitive addition to an open segment. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate
sgdel *(returned by GDDM) (fullword integer)*
Segment deletion. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set detectability (GSDTEC), Set segment priority (GSSGP), Set segment transformation (GSSGT), Set visibility (GSVIS), Set highlighting (GSHLIT), Delete segment (GDSG), Delete segment from workstation (GDSGWK), Insert segment (GINSG)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The value *GIRG* means that implicit regeneration is necessary when an attribute is modified. You should call Update workstation (GUWK) or Redraw all segments on workstation (GRSGWK) to get a true representation of the current graphics picture.

The value *GIMM* means that segment attribute changes appear immediately on the display surface.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQDSK

Purpose

GQDSK	**(wtype, devno, n, mldr, errind, dbufsk, ol, pet, earea, buflen, ldr, datrec)**
APL code	1501
GKS RCP code	X'3800D900' (939579648)

Function: To inquire default stroke device data.

Inquiry function. Returns the default data for a *stroke* device at a workstation, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
devno *(specified by user) (fullword integer)*
The logical input device number.
n *(specified by user) (fullword integer)*
The element requested from the list of available prompt and echo types. If the value specified is 0, the number of available prompt and echo types is returned in *ol* but *pet* is undefined.
mldr *(specified by user) (fullword integer)*
The dimension of array *datrec*.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
dbufsk *(returned by GDDM) (fullword integer)*
The maximum input buffer size (1 ... n).
ol *(returned by GDDM) (fullword integer)*
The number of available prompt/echo types (1 ... n).
pet *(returned by GDDM) (fullword integer)*
Element *n* of the list of available prompt and echo types.
earea *(returned by GDDM) (array of short floating-point numbers)*
The default echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).
buflen *(returned by GDDM) (fullword integer)*
The default stroke buffer length.
ldr *(returned by GDDM) (fullword integer)*
The number of elements used in the data record array.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Unpack data record (GURECS), Initialize stroke (GINSK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS reports error 2001 in *errind*. Usually, a maximum of length of 80 (*mldr*=1) is sufficient.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation

2001	Output parameter size insufficient
2002	List element or set member not available

GQDSP

Purpose

GQDSP	(wtype, errind, dcunit, rx, ry, lx, ly)
APL code	1388
GKS RCP code	X'3800C300' (939574016)

Function: To inquire display space size.

Inquiry function. Returns the device units and the maximum display surface size, in both device coordinates and raster units, for a workstation, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
dcunit *(returned by GDDM) (fullword integer)*
The device coordinate system units. The possible values are:
0 (GMETRE) Meters
1 (GOTHU) Other units
rx *(returned by GDDM) (short floating point)*
ry *(returned by GDDM) (short floating point)*
The maximum display surface size in device coordinates.
lx *(returned by GDDM) (fullword integer)*
ly *(returned by GDDM) (fullword integer)*
The maximum display surface size in raster units.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

None.

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
31	Specified workstation is of category MO
33	Specified workstation is of category MI
36	Specified workstation is Workstation Independent Segment Storage

GQDST

Purpose

GQDST	**(wtype, devno, n, mldr, errind, mbuff, ol, pet, earea, buflen, ldr, datrec)**
APL code	1504
GKS RCP code	X'3800DD00' (939580672)

Function: To inquire default string device data.

Inquiry function. Returns the default data for a *string* device at a workstation, from the workstation description table. (See the section “GDDM-GKS restrictions” below.)

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
devno *(specified by user) (fullword integer)*
The string device number (workstation-dependent).
n *(specified by user) (fullword integer)*
The element requested from the list of available prompt and echo types. If the value specified is 0, the number of available prompt and echo types is returned in *ol* but *pet* is undefined.
mldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
mbuff *(returned by GDDM) (fullword integer)*
The maximum string buffer size.
ol *(returned by GDDM) (fullword integer)*
The number of available prompt and echo types.
pet *(returned by GDDM) (fullword integer)*
Element *n* of the list of available prompt and echo types.
earea *(returned by GDDM) (array of short floating-point numbers)*
The default echo in device coordinates (XMIN, XMAX, YMIN, YMAX).
buflen *(returned by GDDM) (fullword integer)*
The default string buffer length.

ldr *(returned by GDDM) (fullword integer)*
The number of elements used in the data record array.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Initialize string (GINSTS)

Description

(See the section "GDDM-GKS restrictions" below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS reports error 2001 in *errind*.

GDDM-GKS Restrictions

GDDM-GKS ignores the *mldr* parameter; the data record returned is empty.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2001	Output parameter size insufficient
2002	List element or set member not available

GQDVL

Purpose

GQDVL	**(wtype, devno, n, mldr, errind, dval, ol, pet, earea, loval, hival, ldr, datrec)**
APL code	1502
GKS RCP code	X'3800DA00' (939579904)

Function: To inquire default valuator device data.

Inquiry function. Returns the default data for a *valuator* device at a workstation, from the workstation device table. (See the section "GDDM-GKS restrictions" below.)

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
devno *(specified by user) (fullword integer)*
The logical input device number (1 ... n).
n *(specified by user) (fullword integer)*
The element requested from the list of available prompt and echo types. If the value specified is 0, the number of available prompt and echo types is returned in *ol* but *pet* is undefined.
mldr *(specified by user) (fullword integer)*
The dimension of array *datrec*.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
dval *(returned by GDDM) (short floating point)*
The default initial value.
ol *(returned by GDDM) (fullword integer)*
The number of available prompt and echo types (always *1* in GDDM-GKS).
pet *(returned by GDDM) (fullword integer)*
Element *n* of the list of available prompt and echo types.
earea *(returned by GDDM) (array of short floating-point numbers)*
The default echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).
loval *(returned by GDDM) (short floating point)*
hival *(returned by GDDM) (short floating point)*
The minimal and maximal values of the valuator.
ldr *(returned by GDDM) (fullword integer)*
The number of elements used in the data record array.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Initialize valuator (GINVL)

Description

(See the section "GDDM-GKS restrictions" below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator. If the list element requested is out of range, the closest values in range are returned.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS reports error 2001 in *errind*.

GDDM-GKS Restrictions

GDDM-GKS ignores the *mldr* parameter; the data record returned is empty.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2001	Output parameter size insufficient
2002	List element or set member not available

GQDWKA

Purpose

GQDWKA	**(wtype, errind, plbun, pmbun, txbun, fabun, parep, colrep, wktr)**
APL code	1390
GKS RCP code	X'3800C400' (939574272)

Function: To inquire dynamic modification of workstation attributes.

Inquiry function. Returns dynamic modification information for output attributes for a workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
plbun *(returned by GDDM) (fullword integer)*
The polyline bundle representation flag. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate
pmbun *(returned by GDDM) (fullword integer)*
The polymarker bundle representation flag. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate
txbun *(returned by GDDM) (fullword integer)*
The text bundle representation flag. The possible values are:
0 (GIRG) Implicit regeneration necessary

1 (GIMM) Immediate

fabun *(returned by GDDM) (fullword integer)*
The fill area bundle representation flag. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate

parep *(returned by GDDM) (fullword integer)*
The pattern representation flag. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate

colrep *(returned by GDDM) (fullword integer)*
The color representation flag. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate

wktr *(returned by GDDM) (fullword integer)*
The workstation transformation flag. The possible values are:
0 (GIRG) Implicit regeneration necessary
1 (GIMM) Immediate

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polyline representation (GSPLR), Set polymarker representation (GSPMR), Set text representation (GSTXR), Set fill area representation (GSFAR), Set pattern representation (GSPAR), Set color representation (GSCR), Set workstation window (GSWKWN), Set workstation viewport (GSWKVP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The value *GIRG* means that implicit regeneration is necessary when an attribute is modified. You should call Update workstation (GUWK) or Redraw all segments on workstation (GRSGWK) to get a true representation of the current graphics picture.

The value *GIMM* means that output attribute changes appear immediately on the display surface.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQECI

Purpose

GQECI	(wkid, n, errind, ol, colind)
APL code	1481
GKS RCP code	X'3800B700' (939570944)

Function: To inquire list element of color indexes.

Inquiry function. Returns a color index from the list of color indexes in the workstation state list for a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
n *(specified by user) (fullword integer)*
The list element requested. If the value specified is 0, the number of color table entries is returned in *ol* but *colind* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of color table entries.
colind *(returned by GDDM) (fullword integer)*
Element *n* of the list of color indexes.

Operating states

WSOP, WSAC, SGOP

Related functions

Set color representation (GSCR)

Description

The list of valid color indexes for the workstation is contained in the workstation state list. GEQCI returns one element of the list.

You can make a list of all color indexes in use on the workstation. Make an initial call to GEQCI with *n* = 0. The output includes the total number of elements in the color list (*ol*). Now make repeated calls until you have all the elements.

Here is an example:

```
      n=1
      CALL GEQCI (wkid, 0, errind, ol, colind)
      DO  55 n = 1, ol
      CALL GEQCI (wkid, n, errind, ol, colind)
      WRITE ( *, 50) colind
50    FORMAT (' colind', I6)
55    CONTINUE
```

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
2002	List element or set member not available

GQEFAI

Purpose

GQEFAI	**(wkid, n, errind, ol, faind)**
APL code	1477
GKS RCP code	X'3800B300' (939569920)

Function: To inquire list element of fill area indexes.

Inquiry function. Returns a *fill area* index from the list of fill area indexes in the workstation state list of a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
n *(specified by user) (fullword integer)*
The list element requested.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of fill area bundle table entries.
faind *(returned by GDDM) (fullword integer)*
Element *n* of the list of defined fill area indexes.

Operating states

WSOP, WSAC, SGOP

Related functions

Set fill area index (GSFAI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Only one fill area index is returned at a time by this function. The parameter *n* indicates which index is selected. If the value specified for *n* is 0, the number of fill area bundle table entries is returned in *ol* but *faind* is undefined.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
2002	List element or set member not available

GQEGDP

Purpose

GQEGDP	**(wtype, n, errind, ngdp, gdpl)**
APL code	1420
GKS RCP code	X'3800D200' (939577856)

Function: To inquire list element of available generalized drawing primitives.

Inquiry function. Returns a generalized drawing primitive (GDP) identifier from the list of generalized drawing primitives available at a workstation, from the workstation description table. (See the section “GDDM-GKS restrictions” below.)

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
n *(specified by user) (fullword integer)*
The list element requested. If the value specified is 0, the number of available generalized drawing primitives is returned in *ngdp* but *gdpl* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.

ngdp *(returned by GDDM) (fullword integer)*
The number of available generalized drawing primitives.
gdpl *(returned by GDDM) (fullword integer)*
Element *n* of the list of generalized drawing primitive identifiers.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Generalized drawing primitive (GGDP)

Description

(See the section "GDDM-GKS restrictions" below.)

The available generalized drawing primitives are contained in a list of GDP identifiers for each workstation type. This function returns one element of this list.

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

GDDM-GKS Restrictions No GDPs are currently available on the supported workstations. A value of 0 is returned in *ngdp* and *gdpl* is undefined.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQENTN

Purpose

GQENTN	**(n, errind, ol, nprio)**
APL code	1462
GKS RCP code	X'3800A300' (939565824)

Function: To inquire list element of normalization transformation numbers.

Inquiry function. Returns the number of a normalization transformation from the list of normalization transformations in the GKS state list.

Parameters

n *(specified by user) (fullword integer)*
The list element requested. If the value specified is 0, the number of normalization transformations is returned in *ol* but *nprio* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of normalization transformations.
nprio *(returned by GDDM) (fullword integer)*
Element *n* of the list of normalization transformations, ordered by decreasing viewport input priorities.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set window (GSWN), Set viewport (GSVP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQEPAI

Purpose

GQEPAI	**(wkid, n, errind, ol, paind)**
APL code	1479
GKS RCP code	X'3800B500' (939570432)

Function: To inquire list element of pattern indexes.

Inquiry function. Returns a pattern index and the number of pattern table entries, from the workstation state list for a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
n *(specified by user) (fullword integer)*
The list element requested.

errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of pattern table entries.
paind *(returned by GDDM) (fullword integer)*
Element *n* of the list of pattern indexes.

Operating states

WSOP, WSAC, SGOP

Related functions

Set pattern representation (GSPAR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Only one pattern index will be returned at a time by this function. The parameter *n* indicates which list element will be returned.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
2002	List element or set member not available

GQEPLI

Purpose

GQEPLI	**(wkid, n, errind, ol, pli)**
APL code	1470
GKS RCP code	X'3800AC00' (939568128)

Function: To inquire list element of polyline indexes.

Inquiry function. Returns a *polyline* bundle index and the number of polyline bundle table entries, from the workstation state list for a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
n *(specified by user) (fullword integer)*
The list element requested.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
Number of polyline bundle table entries.
pli *(returned by GDDM) (fullword integer)*
Element *n* of the list of defined polyline indexes.

Operating states

WSOP, WSAC, SGOP

Related functions

Set polyline index (GSPLI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Only one polyline index will be returned by this function. The parameter *n* indicates which list element will be returned. If the value specified for *n* is 0, the number of polyline bundle table entries is returned in *ol* but *pli* is undefined.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
2002	List element or set member not available

GQEPMI

Purpose

GQEPMI	**(wkid, n, errind, ol, pmi)**
APL code	1472
GKS RCP code	X'3800AE00' (939568640)

Function: To inquire list element of polymarker indexes.

Inquiry function. Returns a *polymarker* bundle index and the number of polymarker bundles at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
n *(specified by user) (fullword integer)*
The list element requested.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of polymarker bundle table entries.
pmi *(returned by GDDM) (fullword integer)*
Element *n* of the list of defined polymarker indexes.

Operating states

WSOP, WSAC, SGOP

Related functions

Set polymarker index (GSPMI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Only one *polymarker* index is returned at a time by this function. The input parameter *n* specifies which list element is returned. If the value specified for *n* is 0, the number of polymarker bundle table entries is returned in *ol* but *pmi* is undefined.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
2002	List element or set member not available

GQETXI

Purpose

GQETXI	(wkid, n, errind, ol, txind)
APL code	1474
GKS RCP code	X'3800B000' (939569152)

Function: To inquire list element of text indexes.

Inquiry function. Returns a *text* bundle index and the number of text bundles at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
n *(specified by user) (fullword integer)*
The list element requested.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of text bundle table entries.
txind *(returned by GDDM) (fullword integer)*
Element *n* of the list of defined text indexes.

Operating states

WSOP, WSAC, SGOP

Related functions

Set text index (GSTXI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Only one *text* index is returned at a time by this function. The input parameter *n* specifies which list element is returned. If the value specified for *n* is 0, the number of text bundle table entries is returned in *ol* but *txind* is undefined.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
2002	List element or set member not available

GQEWK

Purpose

GQEWK	(n, errind, number, wktyp)
APL code	1508
GKS RCP code	X'38008200' (939557376)

Function: To inquire list element of available workstation types.

Inquiry function. Returns a workstation type number from the list of available workstation types in the GKS state list.

Parameters

n *(specified by user) (fullword integer)*
The list element requested. If the value specified is 0, the number of workstation types available is returned in *number* but *wktyp* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
number *(returned by GDDM) (fullword integer)*
The number of workstation types available.
wktyp *(returned by GDDM) (fullword integer)*
Element *n* of the list of available workstation types.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Open workstation (GOPWK), Inquire workstation category (GQWKCA)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The workstation type *wktyp* can be used in a subsequent call to the function Open workstation (GOPWK).

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
2002	List element or set member not available

GQFACI

Purpose

GQFACI	(errind, coli)
APL code	1459
GKS RCP code	X'3800A000' (939565056)

Function: To inquire fill area color index.

Inquiry function. Returns the current *fill area* color index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
coli *(returned by GDDM) (fullword integer)*
The fill area color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area color index (GSFACI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQFAF

Purpose

GQFAF	(wtype, ni, nh, errind, nis, is, nhs, hs, npfai)
APL code	1408
GKS RCP code	X'3800CC00' (939576320)

Function: To inquire fill area facilities.

Inquiry function. Returns values giving the *fill area* facilities available for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
ni *(specified by user) (fullword integer)*
The element requested from the list of interior styles. If the value specified is 0, the number of available interior styles is returned in *nis* but *ni* is undefined.
nh *(specified by user) (fullword integer)*
The element requested from the list of hatch styles. If the value specified is 0, the number of available hatch styles is returned in *nhs* but *hs* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
nis *(returned by GDDM) (fullword integer)*
The number of available fill area interior styles.
is *(returned by GDDM) (fullword integer)*
The element *ni* of the list of available fill area interior styles. The possible values are:

0	(GHOLLO)	Hollow
1	(GSOLID)	Solid
2	(GPATTR)	Pattern
3	(GHATCH)	Hatch

nhs *(returned by GDDM) (fullword integer)*
The number of available hatch styles.
hs *(returned by GDDM) (fullword integer)*
The index of the element *nh* of the list of available hatch styles.
npfai *(returned by GDDM) (fullword integer)*
The number of predefined fill area indexes.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area interior style (GSFAIS), Set fill area style index (GSFASI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The list of available hatch and fill area interior styles is contained in the workstation description table made available when GKS is opened. This function returns the requested list element from each list.

You can obtain a list of all interior styles for the workstation using the logic of the following example. First call GQFAF with *ni* = 0 and *nh* = 0. The output includes the total number of styles in each list (*nis* and *nhs*). Now make repeated calls to

GQFAF until you have accumulated list elements equal to the total number of styles.

This example lists all the available fill area interior styles for a workstation:

```
       ni=0
       CALL GQFAF (wtype, ni, nh, errind,
      *nis, is, nhs, hs, npfai)
       DO  55 ni = 1, nis
       CALL GQFAF (wtype, ni, nh, errind,
      *nis, is, nhs, hs, npfai)
       WRITE ( *, 50) ni, is
50     FORMAT ('Interior Style Number', I6, 'is',I6)
55     CONTINUE
```

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
2002	List element or set member not available

GQFAI

Purpose

GQFAI	(errind, fai)
APL code	1443
GKS RCP code	X'38009000' (939560960)

Function: To inquire fill area index.

Inquiry function. Returns the current *fill area* index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
fai *(returned by GDDM) (fullword integer)*
The current fill area index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area index (GSFAI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQFAIS

Purpose

GQFAIS	**(errind, ints)**
APL code	1457
GKS RCP code	X'38009E00' (939564544)

Function: To inquire fill area interior style.

Inquiry function. Returns the current *fill area* interior style from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.

ints *(returned by GDDM) (fullword integer)*
The fill area interior style. The possible values are:

0 (GHOLLO) Hollow
1 (GSOLID) Solid
2 (GPATTR) Pattern
3 (GHATCH) Hatch

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area interior style (GSFAIS)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQFAR

Purpose

GQFAR	**(wkid, fai, type, errind, ints, styli, coli)**
APL code	1478
GKS RCP code	X'3800B400' (939570176)

Function: To inquire fill area representation.

Inquiry function. Returns the representation associated with a given *fill area* index at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
fai *(specified by user) (fullword integer)*
The fill area index.
type *(specified by user) (fullword integer)*
The type of returned values. The possible values are:
0 (GSET) Set
1 (GREALI) Realized

If *type* equals *set*, the values are the same as those originally passed by the Set fill area representation (GSFAR) function call. If *type* equals *realized*, the returned values are the closest available actual values at that device.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ints *(returned by GDDM) (fullword integer)*
The fill area interior style. The possible values are:
0 (GHOLLO) Hollow
1 (GSOLID) Solid
2 (GPATTR) Patterned
3 (GHATCH) Hatched

styli *(returned by GDDM) (fullword integer)*
The fill area interior style index. This gives the hatch style or pattern according to *ints*.
coli *(returned by GDDM) (fullword integer)*
The fill area color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area representation (GSFAR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
80	Fill area index is invalid
81	A representation for the specified fill area index has not been defined on this workstation
2000	Enumeration type out of range

GQFASI

Purpose

GQFASI	**(errind, styli)**
APL code	1458
GKS RCP code	X'38009F00' (939564800)

Function: To inquire fill area style index.

Inquiry function. Returns the current fill area style index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
styli *(returned by GDDM) (fullword integer)*
The fill area style index. This gives the hatch style or pattern according to the current fill area interior style.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area style index (GSFASI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQGDP

Purpose

GQGDP	**(wtype, gdp, errind, nbnd, bndl)**
APL code	1421
GKS RCP code	X'3800D300' (939578112)

Function: To inquire generalized drawing primitive.

Inquiry function. Returns the bundled attributes used for the specified generalized drawing primitive (GDP) for a workstation type, from the workstation description table. (See the section “GDDM-GKS restrictions” below.)

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
gdp *(specified by user) (fullword integer)*
The GDP identifier.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
nbnd *(returned by GDDM) (fullword integer)*
The number of sets of the bundled type required to produce the GDP.

bndl *(returned by GDDM) (an array of fullword integers)*
The list of bundle tables used. A maximum of four array elements can be returned. The possible values for each element are:
0 (GPLBND) Polyline bundle
1 (GPMBND) Polymarker bundle
2 (GTXBND) Text bundle
3 (GFABND) Fill area bundle

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

None.

Description

(See the section "GDDM-GKS restrictions" below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

GDDM-GKS Restrictions

No GDPs are available on the workstations supported by GDDM-GKS; if this function is invoked, error 41 is reported.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
41	Specified workstation type is not able to generate the specified generalized drawing primitive

GQIQOV

Purpose

GQIQOV	**(errind, wkid, icl, idn)**
APL code	1437
GKS RCP code	X'3800E300' (939582208)

Function: To inquire input queue overflow.

Inquiry function. Returns the identification of the logical input device that caused an input queue overflow. (See the section "GDDM-GKS restrictions" below.)

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
wkid *(returned by GDDM) (fullword integer)*
The workstation identifier.
icl *(returned by GDDM) (fullword integer)*
The input class. The possible values are:

1	(GLOCAT)	Locator
2	(GSTROK)	Stroke
3	(GVALUA)	Valuator
4	(GCHOIC)	Choice
5	(GPICK)	Pick
6	(GSTRIN)	String

idn *(returned by GDDM) (fullword integer)*
The input device number.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

None

Description

(See the section “GDDM-GKS restrictions” below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143 in *errind*.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS
148	Input queue has not overflowed since GKS was opened or the last invocation of INQUIRE INPUT QUEUE OVERFLOW
149	Input queue has overflowed, but associated workstation has been closed

---

GQLCS

Purpose

GQLCS	(wkid, lcdnr, type, mldr, errind, mode, esw, tnr, ipx, ipy, pet, earea, ldr, datrec)
APL code	1500
GKS RCP code	X'3800BB00' (939571968)

Function: To inquire locator device state.

Inquiry function. Returns the state of a specified *locator* device, from the workstation state list of a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
lcdnr *(specified by user) (fullword integer)*
The locator device number (1 ... n).
type *(specified by user) (fullword integer)*
The type of returned values. The possible values are:
0 (GSET) Set
1 (GREALI) Realized

If *type* equals *set*, the attribute values returned in the data record are the same as those originally passed by the Initialize Locator function. If *type* equals *realized*, the attribute values returned in the data record are the closest available values at the device.
mldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
mode *(returned by GDDM) (fullword integer)*
The operating mode. The only possible value is:
0 (GREQU) Request
esw *(returned by GDDM) (fullword integer)*
The echo switch setting. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo
tnr *(returned by GDDM) (fullword integer)*
The initial transformation number (0 ... n).
ipx *(returned by GDDM) (short floating point)*
ipy *(returned by GDDM) (short floating point)*
The initial locator position in world coordinates.
pet *(returned by GDDM) (fullword integer)*
The prompt and echo type used (1 ... n).
earea *(returned by GDDM) (array of short floating-point numbers)*
The echo area box in device coordinates (XMIN, XMAX, YMIN, YMAX).
ldr *(returned by GDDM) (fullword integer)*
The number of array elements used in the data record array *datrec*.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Unpack data record (GURECS), Initialize locator (GINLC)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The Unpack data record (GURECS) utility is available to unpack the data record returned by GQLCS.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS issues error 2001 in *errind*, and the function terminates to protect the program from a memory error. Usually, a maximum length of 80 (*mldr* = 1) is sufficient.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2000	Enumeration type out of range
2001	Output parameter size insufficient

GQLI

Purpose

GQLI	**(wtype, errind, nlcd, nskd, nvld, nchd, npkd, nstd)**
APL code	1426
GKS RCP code	X'3800D700' (939579136)

Function: To inquire number of available logical input devices.

Inquiry function. Returns the number of available logical input devices of each input class for a workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
nlcd *(returned by GDDM) (fullword integer)*
The number of locator devices (0 ... n).
nskd *(returned by GDDM) (fullword integer)*
The number of stroke devices (0 ... n).
nvld *(returned by GDDM) (fullword integer)*
The number of valuator devices (0 ... n).
nchd *(returned by GDDM) (fullword integer)*
The number of choice devices (0 ... n).
npkd *(returned by GDDM) (fullword integer)*
The number of pick devices (0 ... n).
nstd *(returned by GDDM) (fullword integer)*
The number of string devices (0 ... n).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Request choice (GRQCH)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
38	Specified workstation is neither of category INPUT nor of category OUTIN

GQLN

Purpose

GQLN	(errind, ltype)
APL code	1447
GKS RCP code	X'38009400' (939561984)

Function: To inquire linetype.

Inquiry function. Returns the current line type from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
ltype *(returned by GDDM) (fullword integer)*
The line type. The possible values are:
0 (GLSOLI) Solid line
1 (GLDASH) Dashed line
2 (GLDOT) Dotted line
3 (GLDASD) Dash-dotted line

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set linetype (GSLN)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQLVKS

Purpose

GQLVKS	(errind, level)
APL code	1499
GKS RCP code	X'38008100' (939557120)

Function: To inquire level of GKS.

Inquiry function. Returns the level of GKS, from the GKS description table.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
level *(returned by GDDM) (fullword integer)*
The level of GKS. The possible values are:

0	(GLOA)	0a
1	(GLOB)	0b
2	(GLOC)	0c
3	(GL1A)	1a
4	(GL1B)	1b
5	(GL1C)	1c
6	(GL2A)	2a
7	(GL2B)	2b
8	(GL2C)	2c

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

None

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Each level indicates a subset of GKS standard functions supported by a certain implementation. GDDM-GKS returns the value 7 (GKS level 2b).

Principal errors

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

GQLWK

Purpose

GQLWK	**(wtype, errind, mplbte, mpmbte, mtxbte, mfabte, mpai, mcoli)**
APL code	1423
GKS RCP code	X'3800D400' (939578368)

Function: To inquire maximum length of workstation state tables.

Inquiry function. Returns the maximum number of bundle table, pattern, and color entries available at a workstation, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
mplbte *(returned by GDDM) (fullword integer)*
The maximum number of polyline bundle representations (20 ... n).
mpmbte *(returned by GDDM) (fullword integer)*
The maximum number of polymarker bundle representations (20 ... n).
mtxbte *(returned by GDDM) (fullword integer)*
The maximum number of text bundle representations (20 ... n).
mfabte *(returned by GDDM) (fullword integer)*
The maximum number of fill area representations (10 ... n).
mpai *(returned by GDDM) (fullword integer)*
The maximum number of pattern representations (0 ... n).
mcoli *(returned by GDDM) (fullword integer)*
The maximum number of color representations.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

None.

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQLWSC

Purpose

GQLWSC	(errind, lwidth)
APL code	1448
GKS RCP code	X'38009500' (939562240)

Function: To inquire linewidth scale factor.

Inquiry function. Returns the current line width scale factor from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
lwidth *(returned by GDDM) (short floating point)*
The line width scale factor.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set linewidth scale factor (GSLWSC)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQMK

Purpose

GQMK	(errind, mtype)
APL code	1450
GKS RCP code	X'38009700' (939562752)

Function: To inquire marker type.

Inquiry function. Returns the current marker type from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
mtype *(returned by GDDM) (fullword integer)*
The marker type. The possible values and the markers associated with them are:
0 (GPOINT) Dot
1 (GPLUS) Plus
2 (GAST) Asterisk
3 (GOMARK) Circle

4 (GXMARK) Cross

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set marker type (GSMK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQMKSC

Purpose

GQMKSC	**(errind, mszsf)**
APL code	1451
GKS RCP code	X'38009800' (939563008)

Function: To inquire marker size scale factor.

Inquiry function. Returns the current marker size scale factor from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
mszsf *(returned by GDDM) (short floating point)*
The marker size scale factor.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set marker size scale factor (GSMKSC)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQMNTN

Purpose

GQMNTN	(errind, maxtnr)
APL code	1506
GKS RCP code	X'38008400' (939557888)

Function: To inquire maximum normalization transformation number.

Inquiry function. Returns the largest number that you can use to identify a normalized transformation, from the GKS description table.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
maxtnr *(returned by GDDM) (fullword integer)*
The maximum normalization transformation number.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set window (GSWN), Set viewport (GSVP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQNT

Purpose

GQNT	(ntnr, errind, window, viewpt)
APL code	1463
GKS RCP code	X'3800A400' (939566080)

Function: To inquire normalization transformation.

Inquiry function. Returns the window and viewport limits of a specified normalization transformation, from the GKS state list.

Parameters

ntnr *(specified by user) (fullword integer)*
The number of the normalization transformation requested.
errind *(returned by GDDM) (fullword integer)*
The error indicator
window *(returned by GDDM) (array of short floating-point numbers)*
The window limits, in world coordinates, of the normalization transformation given by *ntnr*. Four floating-point numbers are returned in the array: WXMIN, WXMAX, WYMIN, WYMAX.
viewpt *(returned by GDDM) (array of short floating-point numbers)*
The viewport limits, in normalized device coordinates, of the normalization transformation given by *ntnr*. Four floating-point numbers are returned in the array: VXMIN, VXMAX, VYMIN, VYMAX.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set window (GSWN), Set viewport (GSVP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
50	Transformation number is invalid

GQOPS

Purpose

GQOPS	(opsta)
APL code	1305
GKS RCP code	X'38008000' (939556864)

Function: To inquire operating state value.

Inquiry function. Returns the current operating state of GKS.

Parameters

opsta *(returned by GDDM) (fullword integer)*
The operating state value. The possible values are:

0 (GGKCL) GKS closed
1 (GGKOP) GKS open
2 (GWSOP) At least one workstation is open; no workstation active
3 (GWSAC) At least one workstation is active; no segment open
4 (GSGOP) Segment open

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Open GKS (GOPKS), Close GKS (GCLKS), Open workstation (GOPWK), Close workstation (GCLWK), Activate workstation (GACWK), Deactivate workstation (GDAWK), Create segment (GCRSG), Close segment (GCLSG), Emergency close GKS (GECLKS)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

None

GQOPSG

Purpose

GQOPSG	(errind, sgna)
APL code	1465
GKS RCP code	X'3800A600' (939566592)

Function: To inquire name of open segment.

Inquiry function. Returns the name of the currently open segment from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
sgna *(returned by GDDM) (fullword integer)*
The segment name.

Operating states

SGOP

Related functions

Create segment (GCRSG)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQOPWK

Purpose

GQOPWK	(n, errind, ol, wkid)
APL code	1419
GKS RCP code	X'38008500' (939558144)

Function: To inquire set member of open workstations.

Inquiry function. Returns the identifier of an open workstation, and the number of open workstations, from the list of open workstations in the GKS state list.

Parameters

n *(specified by user) (fullword integer)*
The set member requested. If the value specified is 0, the number of open workstations is returned in *ol* but *wkid* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of open workstations.
wkid *(returned by GDDM) (fullword integer)*
Member *n* of the set of open workstations.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Open workstation (GOPWK), Close workstation (GCLWK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If *n* is less than zero or greater than the number of open workstations, error 2002 is returned, unless the set of open workstations is empty. If the set is empty, *ol* is returned as 0.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
2002	List element or set member not available

GQPA

Purpose

GQPA	**(errind, pwx, pwy, phx, phy)**
APL code	1444
GKS RCP code	X'38009100' (939561216)

Function: To inquire pattern size.

Inquiry function. Returns the current pattern size (width and length), from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
pwx *(returned by GDDM) (short floating point)*
pwy *(returned by GDDM) (short floating point)*
The pattern width vector in world coordinates.
phx *(returned by GDDM) (short floating point)*
phy *(returned by GDDM) (short floating point)*
The pattern height vector in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set pattern size (GSPA)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQPAF

Purpose

GQPAF	(wtype, errind, nppai)
APL code	1413
GKS RCP code	X'3800CE00' (939576832)

Function: To inquire pattern facilities.

Inquiry function. Returns the number of predefined pattern indexes available for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
nppai *(returned by GDDM) (fullword integer)*
The number of predefined pattern indexes (0 ... n).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area style index (GSFASI), Inquire fill area facilities (GQFAF)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQPAR

Purpose

GQPAR	**(wkid, pai, type, dimx, dimy, errind, dx, dy, colia)**
APL code	1480
GKS RCP code	X'3800B600' (939570688)

Function: To inquire pattern representation.

Inquiry function. Returns the representation associated with a given pattern index at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.

pai *(specified by user) (fullword integer)*
The pattern index.

type *(specified by user) (fullword integer)*
The type of returned values. The possible values are:

0 (GSET) Set
1 (GREALI) Realized

If *type* equals *set*, the returned values are the same as those originally passed by the Set pattern representation (GSPAR) function call. If *type* equals *realized*, the returned values are the closest available actual values at the device.

dimx *(specified by user) (fullword integer)*
dimy *(specified by user) (fullword integer)*
The maximum dimensions of the pattern array *colia*.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
dx *(returned by GDDM) (fullword integer)*
dy *(returned by GDDM) (fullword integer)*
The dimensions of the pattern array *colia*.
colia *(returned by GDDM) (an array of fullword integers)*
The pattern array (*dimx* groups of *dimy* elements).

Operating states

WSOP, WSAC, SGOP

Related functions

Set pattern representation (GSPAR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
85	Specified pattern index is invalid
88	A representation for the specified pattern index has not been defined on this workstation
90	Interior style PATTERN is not supported on this workstation

GQPARF

Purpose

GQPARF	**(errinf, rfx, rfy)**
APL code	1445
GKS RCP code	X'38009200' (939561472)

Function: To inquire pattern reference point.

Inquiry function. Returns the current value of the pattern reference point from the GKS state list.

Parameters

errinf *(returned by GDDM) (fullword integer)*
The error indicator.
rfx *(returned by GDDM) (short floating point)*
rfy *(returned by GDDM) (short floating point)*
The current pattern reference point in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set pattern reference point (GSPARF)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQPCR

Purpose

GQPCR	(wtype, pci, errind, cr, cg, cb)
APL code	1417
GKS RCP code	X'3800D100' (939577600)

Function: To inquire predefined color representation.

Inquiry function. Returns the intensities of red, green, and blue color associated with a predefined color index for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
pci *(specified by user) (fullword integer)*
The predefined color index (0 ... n).
errind *(returned by GDDM) (fullword integer)*
The error indicator.

cr *(returned by GDDM) (short floating point)*
cg *(returned by GDDM) (short floating point)*
cb *(returned by GDDM) (short floating point)*
The color intensities (red, green, and blue).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set color representation (GSCR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
93	Color index is invalid
95	A representation for the specified color index has not been predefined on this workstation

GQPFAR

Purpose

GQPFAR	**(wtype, pfai, errind, style, stylid, coli)**
APL code	1409
GKS RCP code	X'3800CD00' (939576576)

Function: To inquire predefined fill area representation.

Inquiry function. Returns the representation associated with a predefined *fill area* index for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
pfai *(specified by user) (fullword integer)*
The predefined fill area index.

errind *(returned by GDDM) (fullword integer)*
The error indicator.
style *(returned by GDDM) (fullword integer)*
The fill area interior style. The possible values are:
0 (GHOLLO) Hollow
1 (GSOLID) Solid
2 (GPATTR) Pattern
3 (GHATCH) Hatch
stylid *(returned by GDDM) (fullword integer)*
The fill area style index.
coli *(returned by GDDM) (fullword integer)*
The fill area color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area index (GSFAI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
80	Fill area index is invalid
82	A representation for the specified fill area index has not been predefined on this workstation

GQPKID

Purpose

GQPKID	(errind, pkid)
APL code	1446
GKS RCP code	X'38009300' (939561728)

Function: To inquire pick identifier.

Inquiry function. Returns the current *pick* identifier from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
pkid *(returned by GDDM) (fullword integer)*
The current pick identifier.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set pick identifier (GSPKID)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQPKS

Purpose

GQPKS	**(wkid, pkdnr, type, mldr, errind, mode, esw, istat, isgna, ipkid, pet, earea, ldr, datrec)**
APL code	1488
GKS RCP code	X'3800BF00' (939572992)

Function: To inquire pick device state.

Inquiry function. Returns the state of a specified *pick* device at a workstation, from the workstation state list. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pkdnr *(specified by user) (fullword integer)*
The pick device number (1 ... n).
type *(specified by user) (fullword integer)*
The type of returned value. The possible values are:
0 (GSET) Set
1 (GREALI) Realized

If *type* equals *set*, the returned values are the same as those originally passed by the Initialize pick (GINPK) function. If *type* equals *realized*, the returned values are the closest available values that would be used on a Request pick (GRQPK) call.

mldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.

errind *(returned by GDDM) (fullword integer)*
The error indicator.

mode *(returned by GDDM) (fullword integer)*
The operating mode. The only possible value is:
0 (GREQU) Request

esw *(returned by GDDM) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo

istat *(returned by GDDM) (fullword integer)*
The initial status. The possible values are:
1 (GOK) OK
2 (GNPICK) No pick

isgna *(returned by GDDM) (fullword integer)*
The initial segment identifier.

ipkid *(returned by GDDM) (fullword integer)*
The initial pick identifier.

pet *(returned by GDDM) (fullword integer)*
The prompt and echo type (1 ... n).

earea *(returned by GDDM) (array of short floating-point numbers)*
The echo area in device coordinates (XMIN, XMAX, YMIN, YMAX)

ldr *(returned by GDDM) (fullword integer)*
The number of elements used in the data record array *datrec*.

datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Unpack data record (GURECS), Initialize pick (GINPK)

Description

(See the section “GDDM-GKS restrictions” below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The parameter *mldr* reserves space for the data record returned by the inquiry. If you do not specify sufficient length to accommodate the data record, GKS reports error 2001 in *errind*.

GDDM-GKS Restrictions

GDDM-GKS ignores the *mldr* parameter. The data record returned is empty.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
37	Specified workstation is not of category OUTIN
140	Specified input device is not present on workstation
2000	Enumeration type out of range
2001	Output parameter size insufficient

GQPLCI

Purpose

GQPLCI	**(errind, coli)**
APL code	1449
GKS RCP code	X'38009600' (939562496)

Function: To inquire polyline color index.

Inquiry function. Returns the current *polyline* color index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
coli *(returned by GDDM) (fullword integer)*
The polyline color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polyline color index (GSPLCI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQPLF

Purpose

GQPLF	(wtype, n, errind, nlt, lt, nlw, nomlw, rlwmin, rlwmax, nppli)
APL code	1397
GKS RCP code	X'3800C600' (939574784)

Function: To inquire polyline facilities.

Inquiry function. Returns values giving the *polyline* facilities available for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
n *(specified by user) (fullword integer)*
The element requested from the list of available line types. If the value specified is 0, the number of available line types is returned in *nlt* but *lt* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
nlt *(returned by GDDM) (fullword integer)*
The number of available line types (4 ... n).
lt *(returned by GDDM) (fullword integer)*
Element *n* of the list of available line types (1 ... n).
nlw *(returned by GDDM) (fullword integer)*
The number of available line widths (0 ... n).
nomlw *(returned by GDDM) (short floating point)*
The nominal line width in device coordinates.
rlwmin *(returned by GDDM) (short floating point)*
The minimum line width available in device coordinates.
rlwmax *(returned by GDDM) (short floating point)*
The maximum line width available in device coordinates.
nppli *(returned by GDDM) (fullword integer)*
The number of predefined polyline indexes (5 ... n).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set linetype (GSLN), Set linewidth scale factor (GSLWSC), Set polyline index (GSPLI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If the number of available line widths is returned as 0 (zero), the workstation supports a continuous range of line widths.

The available line types are contained in the workstation description table. GQPLF returns element *n* of the list. You can obtain the entire list using the logic of the following example. Make an initial call to GQPLF with *n* = 0. The output includes the total number of available line types (*nlt*). Now make repeated calls to GQPLF until you have accumulated list elements equal to the total number of types.

Here is an example:

```
       n=0
       GQPLF (wtype, n, errind, nlt, lt, nlw,
      *nomlw, rlwmin, rlwmax, nppli)
       DO  55 n = 1, nlt
       GQPLF (wtype, n, errind, nlt, lt, nlw,
      *nomlw, rlwmin, rlwmax, nppli)
       WRITE ( *, 50) lt
50     FORMAT (' lt', I6)
55     CONTINUE
```

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
2002	List element or set member not available

GQPLI

Purpose

GQPLI	**(errind, pli)**
APL code	1427
GKS RCP code	X'38008700' (939558656)

Function: To inquire polyline index.

Inquiry function. Returns the current *polyline* index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
pli *(returned by GDDM) (fullword integer)*
The current polyline index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polyline index (GSPLI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQPLR

Purpose

GQPLR	**(wkid, pli, type, errind, ltype, lwidth, coli)**
APL code	1471
GKS RCP code	X'3800AD00' (939568384)

Function: To inquire polyline representation.

Inquiry function. Returns the representation associated with a given *polyline* index at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pli *(specified by user) (fullword integer)*
The polyline index.
type *(specified by user) (fullword integer)*
The type of returned values. The possible values are:
0 (GSET) Set
1 (GREALI) Realized

If *type* equals *set*, the returned values are the same as those originally passed by the Set polyline representation (GSPLR) function call. If *type* equals *realized*, the values are the closest available actual values at the device.

errind *(returned by GDDM) (fullword integer)*
The error indicator.

ltype *(returned by GDDM) (fullword integer)*
The line type. The possible values are:

0	(GLSOLI)	Solid line
1	(GLDASH)	Dashed line
2	(GLDOT)	Dotted line
3	(GLDASD)	Dashed-dotted

lwidth *(returned by GDDM) (short floating point)*
The line-width scale factor.

coli *(returned by GDDM) (fullword integer)*
The polyline color index.

Operating states

WSOP, WSAC, SGOP

Related functions

Set polyline representation (GSPLR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
60	Polyline index is invalid
61	A representation for the specified polyline index has not been defined on this workstation
2000	Enumeration type out of range

GQPMCI

Purpose

GQPMCI	**(errind, coli)**
APL code	1452
GKS RCP code	X'38009900' (939563264)

Function: To inquire polymarker color index.

Inquiry function. Returns the current *polymarker* color index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
coli *(returned by GDDM) (fullword integer)*
The polymarker color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polymarker index (GSPMI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQPMF

Purpose

GQPMF	**(wtype, n, errind, nmt, mt, nms, nomms, rmsmin, rmsmax, nppmi)**
APL code	1402
GKS RCP code	X'3800C800' (939575296)

Function: To inquire polymarker facilities.

Inquiry function. Returns values giving the *polymarker* facilities available for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
n *(specified by user) (fullword integer)*
The element requested from the list of available marker types. If the value specified is 0, the number of available marker types is returned in *nmt* but *mt* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
nmt *(returned by GDDM) (fullword integer)*
The number of available marker types (5 ... n).
mt *(returned by GDDM) (fullword integer)*
Element *n* of the list of available marker types.
nms *(returned by GDDM) (fullword integer)*
The number of available marker sizes (0 ... n).
nomms *(returned by GDDM) (short floating point)*
The nominal marker size in device coordinates. When markers are drawn, the marker size is calculated as the nominal marker size multiplied by the marker size scale factor. Note that the nominal size is the same for all marker types and most markers are actually defined to be smaller than the nominal size.
rmsmin *(returned by GDDM) (short floating point)*
rmsmax *(returned by GDDM) (short floating point)*
The range of marker sizes available in device coordinates.
nppmi *(returned by GDDM) (fullword integer)*
The number of predefined polymarker indexes.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polymarker index (GSPMI), Set marker size scale factor (GSMKSC), Set marker type (GSMK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If the number of marker sizes is returned as 0 (zero), the workstation supports a continuous range of marker sizes.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
2002	List element or set member not available

GQPMI

Purpose

GQPMI	(errind, pmi)
APL code	1428
GKS RCP code	X'38008800' (939558912)

Function: To inquire polymarker index.

Inquiry function. Returns the current the *polymarker* index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
pmi *(returned by GDDM) (fullword integer)*
The current polymarker index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polymarker index (GSPMI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQPMR

Purpose

GQPMR	(wkid, pmi, type, errind, mtype, mszsf, coli)
APL code	1473
GKS RCP code	X'3800AF00' (939568896)

Function: To inquire polymarker representation.

Inquiry function. Returns the representation associated with a given *polymarker* index at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pmi *(specified by user) (fullword integer)*
The polymarker index.
type *(specified by user) (fullword integer)*
The type of returned values. The possible values are:
0 (GSET) Set
1 (GREALI) Realized

If *type* equals *set*, the returned values are the same as those originally passed by the Set polymarker representation (GSPMR) function call. If *type* equals *realized*, the returned values are the closest available actual values at the device.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
mtype *(returned by GDDM) (fullword integer)*
The marker type. The possible values are:
0 (GPOINT) Dot
1 (GPLUS) Plus
2 (GAST) Asterisk
3 (GOMARK) Circle
4 (GXMARK) Cross
mszsf *(returned by GDDM) (short floating point)*
The marker size scale factor.
coli *(returned by GDDM) (fullword integer)*
The polymarker color index.

Operating states

WSOP, WSAC, SGOP

Related functions

Set polymarker representation (GSPMR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage

66	Polymarker index is invalid
67	A representation for the specified polymarker index has not been defined on this workstation
2000	Enumeration type out of range

GQPPAR

Purpose

GQPPAR	(wtype, ppai, dimx, dimy, errind, dx, dy, parray)
APL code	1414
GKS RCP code	X'3800CF00' (939577088)

Function: To inquire predefined pattern representation.

Inquiry function. Returns the representation associated with a predefined pattern index for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
ppai *(specified by user) (fullword integer)*
The predefined pattern index (1 ... n).
dimx *(specified by user) (fullword integer)*
dimy *(specified by user) (fullword integer)*
The maximum pattern array dimensions.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
dx *(returned by GDDM) (fullword integer)*
dy *(returned by GDDM) (fullword integer)*
The dimensions of the pattern array returned in *parray*.
parray *(returned by GDDM) (an array of fullword integers)*
The pattern array.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set pattern representation (GSPAR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
85	Specified pattern index is invalid
89	A representation for the specified pattern index has not been predefined on this workstation
90	Interior style PATTERN is not supported on this workstation
2001	Output parameter size insufficient

GQPPLR

Purpose

GQPPLR	**(wtype, pli, errind, lntype, lwidth, coli)**
APL code	1400
GKS RCP code	X'3800C700' (939575040)

Function: To inquire predefined polyline representation.

Inquiry function. Returns the representation associated with a predefined *polyline* index for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
pli *(specified by user) (fullword integer)*
The predefined polyline index (1 ... n).
errind *(returned by GDDM) (fullword integer)*
The error indicator.
lntype *(returned by GDDM) (fullword integer)*
The line type. The possible values are:
0 (GLSOLI) Solid line
1 (GLDASH) Dashed line
2 (GLDOT) Dotted line
3 (GLDASD) Dash-dotted line
lwidth *(returned by GDDM) (short floating point)*
The line-width scale factor.

coli *(returned by GDDM) (fullword integer)*
The polyline color index (0 ... n).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polyline index (GSPLI), Set polyline representation (GSPLR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
60	Polyline index is invalid
62	A representation for the specified polyline index has not been predefined on this workstation

GQPPMR

Purpose

GQPPMR	**(wtype, pmi, errind, mktype, mksscf, coli)**
APL code	1404
GKS RCP code	X'3800C900' (939575552)

Function: To inquire predefined polymarker representation.

Inquiry function. Returns the representation associated with a predefined *polymarker* index for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
pmi *(specified by user) (fullword integer)*
The predefined polymarker index (1 ... n).
errind *(returned by GDDM) (fullword integer)*
The error indicator.

mktype *(returned by GDDM) (fullword integer)*
The marker type. The possible values are:
0 (GPOINT) Dot
1 (GPLUS) Plus
2 (GAST) Asterisk
3 (GOMARK) Circle
4 (GXMARK) Cross
mksscf *(returned by GDDM) (short floating point)*
The marker size scale factor.
coli *(returned by GDDM) (fullword integer)*
The polymarker color index (0 ... n).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polymarker index (GSPMI), Set polymarker representation (GSPMR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
66	Polymarker index is invalid
68	A representation for the specified polymarker index has not been predefined on this workstation

GQPTXR

Purpose

GQPTXR	**(wtype, ptxi, errind, font, prec, charxp, charsp, coli)**
APL code	1406
GKS RCP code	X'3800CB00' (939576064)

Function: To inquire predefined text representation.

Inquiry function. Returns the representation associated with a predefined *text* index for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
ptxi *(specified by user) (fullword integer)*
The predefined text index.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
font *(returned by GDDM) (fullword integer)*
The text font.
prec *(returned by GDDM) (fullword integer)*
The text precision. The possible values are:

0	(GSTRP)	String precision
1	(GCHARP)	Character precision
2	(GSTRKP)	Stroke precision.

charxp *(returned by GDDM) (short floating point)*
The character expansion factor.
charsp *(returned by GDDM) (short floating point)*
The character spacing.
coli *(returned by GDDM) (fullword integer)*
The text color index (0 ... n).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text representation (GSTXR), Set text index (GSTXI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
72	Text index is invalid
74	A representation for the specified text index has not been predefined on this workstation

---

GQPX

Purpose

GQPX	(wkid, px, py, errind, coli)
APL code	1436
GKS RCP code	X'3800E200' (939581952)

Function: To inquire pixel.

Inquiry function. Returns the current color index of a specific pixel on the display surface of a workstation. (See the section "GDDM-GKS restrictions" below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
px *(specified by user) (short floating point)*
py *(specified by user) (short floating point)*
A point in world coordinates.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
coli *(returned by GDDM) (fullword integer)*
The color index. If the pixel is nonexistent, the color index is –1.

Operating states

WSOP, WSAC, SGOP

Related functions

Inquire pixel array (GQPXA), Inquire pixel array dimensions (GQPXAD)

Description

(See the section "GDDM-GKS restrictions" below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

GDDM-GKS Restrictions

The workstations supported by GDDM-GKS have no pixel store read-back capability. If this function is invoked, error 40 is reported in *errind*.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
40	Specified workstation has no pixel store read-back capability

GQPXA

Purpose

GQPXA	(wkid, px, py, dimx, dimy, isc, isr, dx, dy, errind, invval, colia)
APL code	1435
GKS RCP code	X'3800E100' (939581696)

Function: To inquire pixel array.

Inquiry function. Returns the current color indexes of a specific array of pixels on the display surface of a workstation. (See the section "GDDM-GKS restrictions" below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
px *(specified by user) (short floating point)*
py *(specified by user) (short floating point)*
The upper left corner in world coordinates.
dimx *(specified by user) (fullword integer)*
dimy *(specified by user) (fullword integer)*
The dimension of the color index array *colia*.
isc *(specified by user) (fullword integer)*
isr *(specified by user) (fullword integer)*
The start column and start row within the array.
dx *(specified by user) (fullword integer)*
dy *(specified by user) (fullword integer)*
The size of the requested pixel array.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
invval *(returned by GDDM) (fullword integer)*
The presence of invalid values. The possible values are:
0 (GABSNT) Absent
1 (GPRSNT) Present
colia *(returned by GDDM) (an array of fullword integers)*
The color index array (*dimx* groups of *dimy* elements).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Inquire pixel (GQPX), Cell array (GCA)

Description

(See the section “GDDM-GKS restrictions” below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

GDDM-GKS Restrictions

The workstations supported by GDDM-GKS have no pixel store read-back capability. If this function is invoked, error 40 is reported in *errind*.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
40	Specified workstation has no pixel store read-back capability
91	Dimensions of color array are invalid

GQPXAD

Purpose

GQPXAD	**(wkid, px, py, qx, qy, errind, n, m)**
APL code	1434
GKS RCP code	X'3800E000' (939581440)

Function: To inquire pixel array dimensions.

Inquiry function. Returns the number of columns and rows of pixels within a specified rectangle on the display surface of a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
px *(specified by user) (short floating point)*
py *(specified by user) (short floating point)*
qx *(specified by user) (short floating point)*
qy *(specified by user) (short floating point)*
The upper left and lower right corners of the rectangle, in world coordinates.

errind *(returned by GDDM) (fullword integer)*
The error indicator.
n *(returned by GDDM) (fullword integer)*
m *(returned by GDDM) (fullword integer)*
The dimensions of the pixel array.

Operating states

WSOP, WSAC, SGOP

Related functions

Cell array (GCA)

Description

(See the section “GDDM-GKS restrictions” below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

GDDM-GKS Restrictions

The workstations supported by GDDM-GKS have no pixel store read-back capability. If this function is invoked, error 40 is reported in *errind*.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
40	Specified workstation has no pixel store read-back capability

GQSGA

Purpose

GQSGA	**(sgna, errind, segtm, vis, high, sgpr, det)**
APL code	1432
GKS RCP code	X'3800DF00' (939581184)

Function: To inquire segment attributes.

Inquiry function. Returns the segment attributes for a segment, from the segment state list.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name, selected when the segment was created.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
segtm *(returned by GDDM) (array of short floating-point numbers)*
The segment transformation matrix. This is a 2 * 3 array, with the elements arranged like this:

```
M(1)    M(3)    M(5)
M(2)    M(4)    M(6)
```

vis *(returned by GDDM) (fullword integer)*
The visibility attribute. The possible values are:
0 (GINVIS) Invisible.
1 (GVISI) Visible.
high *(returned by GDDM) (fullword integer)*
The highlighting attribute. The possible values are:
0 (GNORML) Normal
1 (GHILIT) Highlighted
sgpr *(returned by GDDM) (short floating point)*
The segment priority.
det *(returned by GDDM) (fullword integer)*
The detectability attribute. The possible values are:
0 (GUNDET) Undetectable
1 (GDETEC) Detectable

Operating states

WSOP, WSAC, SGOP

Related functions

Set detectability (GSDTEC), Set segment priority (GSSGP), Set segment transformation (GSSGT), Set visibility (GSVIS), Set highlighting (GSHLIT)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
120	Specified segment name is invalid
122	Specified segment does not exist

GQSGP

Purpose

GQSGP	(wtype, errind, nsg)
APL code	1424
GKS RCP code	X'3800D500' (939578624)

Function: To inquire number of segment priorities supported.

Inquiry function. Returns the number of segment priorities supported by a workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
nsg *(returned by GDDM) (fullword integer)*
The number of segment priorities supported.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set segment priority (GSSGP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If *nsg* is returned as 0 (zero), the workstation supports an infinite number of segment priorities.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQSGUS

Purpose

GQSGUS	(n, errind, ol, sgna)
APL code	1509
GKS RCP code	X'3800A700' (939566848)

Function: To inquire set member of segment names in use.

Inquiry function. Returns a segment name currently in use, from the list of segments in the GKS state list.

Parameters

n *(specified by user) (fullword integer)*
The set member requested. If the value specified is 0, the number of segment names in use is returned in *ol* but *sgna* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of segment names in use.
sgna *(returned by GDDM) (fullword integer)*
Member *n* of the set of segment names in use.

Operating states

WSOP, WSAC, SGOP

Related functions

Create segment (GCRSG), Rename segment (GRENSG)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If *n* is less than zero or greater than the number of segment names in use, error 2002 is returned, unless the set of segment names is empty. If the set is empty, *ol* is returned as 0.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
2002	List element or set member not available

GQSGWK

Purpose

GQSGWK	(wkid, n, errind, ol, sgna)
APL code	1484
GKS RCP code	X'3800BA00' (939571712)

Function: To inquire set member of segment names on workstation.

Inquiry function. Returns a segment name currently in use on a workstation, from the list of segments in the workstation state list of the workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
n *(specified by user) (fullword integer)*
The set member requested. If the value specified is 0, the number of segments at the workstation is returned in *ol* but *sgna* is undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
ol *(returned by GDDM) (fullword integer)*
The number of segments at the workstation (0 ... n).
sgna *(returned by GDDM) (fullword integer)*
Member *n* of the set of segment names on the workstation.

Operating states

WSOP, WSAC, SGOP

Related functions

Create segment (GCRSG), Associate segment with workstation (GASGWK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If *n* is less than zero or greater than the number of segments at the workstation, error 2002 is returned, unless the set of segment names is empty. If the set is empty, *ol* is returned as 0.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
2002	List element or set member not available

GQSIM

Purpose

GQSIM	(errind, flag)
APL code	1466
GKS RCP code	X'3800A800' (939567104)

Function: To inquire more simultaneous events.

Inquiry function. Returns a flag indicating whether there are any more simultaneous input events. (See the section “GDDM-GKS restrictions” below.)

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.

flag *(returned by GDDM) (fullword integer)*
This parameter indicates whether there are more simultaneous events. The possible values are:

0 (GNMORE) No more events
1 (GMORE) More events

Operating states

WSOP, WSAC, WGOP

Related functions

Await event (GWAIT)

Description

(See the section “GDDM-GKS restrictions” below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143 in *errind*.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS

GQSKS

Purpose

GQSKS	**(wkid, skdnr, type, n, mldr, errind, mode, esw, itnr, np, pxa, pya, pet, earea, buflen, ldr, datrec)**
APL code	1485
GKS RCP code	X'3800BC00' (939572224)

Function: To inquire stroke device state.

Inquiry function. Returns values giving the state of a *stroke* device at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
skdnr *(specified by user) (fullword integer)*
The stroke device number (1 ... n).
type *(specified by user) (fullword integer)*
The type of returned values. The possible values are :
0 (GSET) Set
1 (GREALI) Realized

If *type* equals *set*, the returned values are the same as those originally passed by the Initialize stroke (GINSK) function call. If *type* equals *realized*, the returned values are the closest available actual values at the device.
n *(specified by user) (fullword integer)*
The maximum number of points.
mldr *(specified by user) (fullword integer)*
The dimensions of the data record array *datrec*.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
mode *(returned by GDDM) (fullword integer)*
The operating mode. The only possible value is:
0 (GREQU) Request
esw *(returned by GDDM) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo

itnr *(returned by GDDM) (fullword integer)*
The initial transformation number (0 ... n).
np *(returned by GDDM) (fullword integer)*
The number of points (0 ... n).
pxa *(returned by GDDM) (array of short floating-point numbers)*
pya *(returned by GDDM) (array of short floating-point numbers)*
The initial points of the stroke in world coordinates.
pet *(returned by GDDM) (fullword integer)*
The prompt and echo type.
earea *(returned by GDDM) (array of short floating-point numbers)*
The echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).
buflen *(returned by GDDM) (fullword integer)*
The input buffer size.
ldr *(returned by GDDM) (fullword integer)*
The number of array elements used in the data record array *datrec*.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Unpack data record (GURECS), Initialize stroke (GINSK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2000	Enumeration type out of range
2001	Output parameter size insufficient

GQSTS

Purpose

GQSTS	**(wkid,stdnr,mldr,errind,mode,esw, lostr,istr,pet,earea, buflen,inipos,ldr,datrec)**

Function: Inquiry function. Returns values giving the state of a *string* device at a workstation, from the workstation state list. Use this call only if your program is written in FORTRAN IV or VS FORTRAN. Otherwise, use the function Inquire string device state (GQSTSS) instead.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.

stdnr *(specified by user) (fullword integer)*
The string device number.

mldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.

errind *(returned by GDDM) (fullword integer)*
The error indicator.

mode *(returned by GDDM) (fullword integer)*
The operating mode. The only possible value is:

0 (GREQU) Request

esw *(returned by GDDM) (fullword integer)*
The echo switch. The possible values are:

0 (GNECHO) No echo
1 (GECHO) Echo

lostr *(returned by GDDM) (fullword integer)*
The number of characters returned in *istr*.

istr *(returned by GDDM) (character)*
The initial string. In a VS FORTRAN program, the string can be of variable length. For FORTRAN IV, you must define the string as CHARACTER*80.

pet *(returned by GDDM) (fullword integer)*
The prompt and echo type (1 ... n).

earea *(returned by GDDM) (array of short floating-point numbers)*
The echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).

buflen *(returned by GDDM) (fullword integer)*
The input buffer size.

inipos *(returned by GDDM) (fullword integer)*
The initial cursor position.

ldr *(returned by GDDM) (fullword integer)*
The number of array elements used in the data record array *datrec*.

datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Unpack data record (FORTRAN only) (GUREC), Initialize string (FORTRAN only) (GINST)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

```
  7  GKS not in proper state: GKS shall be in one of the states
     WSOP, WSAC, or SGOP
 20  Specified workstation identifier is invalid
 25  Specified workstation is not open
 38  Specified workstation is neither of category INPUT nor of
     category OUTIN
140  Specified input device is not present on workstation
2001 Output parameter size insufficient
```

GQSTSS

Purpose

GQSTSS	**(wkid, stdnr, mstr, mldr, errind, mode, esw, lostr, istr, pet, earea, buflen, inipos, ldr, datrec)**
APL code	1489
GKS RCP code	X'3800C000' (939573248)

Function: To inquire string device state.

Inquiry function. Returns values giving the state of a *string* device at a workstation, from the workstation state list. If your program is written in FORTRAN IV or VS FORTRAN, use the function Inquire string device state (FORTRAN only) (GQSTS) instead.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
stdnr *(specified by user) (fullword integer)*
The string device number.
mstr *(specified by user) (fullword integer)*
The maximum number of characters that can be returned in the *istr* parameter.
mldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.

errind *(returned by GDDM) (fullword integer)*
The error indicator.
mode *(returned by GDDM) (fullword integer)*
The operating mode. The only possible value is:
0 (GREQU) Request
esw *(returned by GDDM) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo
lostr *(returned by GDDM) (fullword integer)*
The number of characters returned in *istr*.
istr *(returned by GDDM) (character)*
The initial string.
pet *(returned by GDDM) (fullword integer)*
The prompt and echo type (1 ... n).
earea *(returned by GDDM) (array of short floating-point numbers)*
The echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).
buflen *(returned by GDDM) (fullword integer)*
The maximum input buffer size.
inipos *(returned by GDDM) (fullword integer)*
The initial cursor position.
ldr *(returned by GDDM) (fullword integer)*
The number of array elements used in the data record array *datrec*.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Unpack data record (GURECS), Initialize string (GINSTS)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2001	Output parameter size insufficient

GQTXAL

Purpose

GQTXAL	(errind, txalh, txalv)
APL code	1442
GKS RCP code	X'38008F00' (939560704)

Function: To inquire text alignment.

Inquiry function. Returns the current text alignment from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
txalh *(returned by GDDM) (fullword integer)*
The current horizontal alignment. The possible values are:
0 (GAHNOR) Normal
1 (GALEFT) Left
2 (GACENT) Center
3 (GARITE) Right
txalv *(returned by GDDM) (fullword integer)*
The current vertical alignment. The possible values are:
0 (GAVNOR) Normal
1 (GATOP) Top
2 (GACAP) Cap
3 (GAHALF) Half
4 (GABASE) Base
5 (GABOTT) Bottom

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text alignment (GSTXAL)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQTXCI

Purpose

GQTXCI	(errind, coli)
APL code	1456
GKS RCP code	X'38009D00' (939564288)

Function: To inquire text color index.

Inquiry function. Returns the current *text* color index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
coli *(returned by GDDM) (fullword integer)*
The text color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text color index (GSTXCI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQTXF

Purpose

GQTXF	(wtype, n, errind, nfpp, font, prec, nchh, minchh, maxchh, nchx, minchx, maxchx, nptxi)
APL code	1405
GKS RCP code	X'3800CA00' (939575808)

Function: To inquire text facilities.

Inquiry function. Returns values giving the *text* facilities available for a given workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
n *(specified by user) (fullword integer)*
The element requested from the list of text font and precision pairs. If the value specified is 0, the number of available text font and precision pairs is returned in *nfpp* but *font* and *prec* are undefined.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
nfpp *(returned by GDDM) (fullword integer)*
The number of text font and precision pairs (1 ... n).
font *(returned by GDDM) (fullword integer)*
Element *n* of the list of available text fonts (1 ... n).
prec *(returned by GDDM) (fullword integer)*
The text font precision. The possible values are:
0 (GSTRP) String precision
1 (GCHARP) Character precision
2 (GSTRKP) Stroke precision
nchh *(returned by GDDM) (fullword integer)*
The number of available character heights (0 ... n).
minchh *(returned by GDDM) (short floating point)*
The minimum character height in device coordinates.
maxchh *(returned by GDDM) (short floating point)*
The maximum character height in device coordinates.
nchx *(returned by GDDM) (fullword integer)*
The number of available expansion factors (0 ... n).
minchx *(returned by GDDM) (short floating point)*
The minimum character expansion factor.
maxchx *(returned by GDDM) (short floating point)*
The maximum character expansion factor.
nptxi *(returned by GDDM) (fullword integer)*
The number of predefined text indexes (2 ... n).

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text font and precision (GSTXFP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator. If the list element requested is out of range, the closest values in range are returned.

If the number of available character heights is returned as 0 (zero), the workstation supports a continuous range of character heights.

If the number of available character expansion factors is returned as 0 (zero), the workstation supports a continuous range of character expansion factors.

The list of available text font and precision pairs is contained in the workstation description table. GQTXF returns the font and precision given by element *n* of the list.

You can obtain the entire lists using the logic of the following example. Make an initial call to GQTXF with *n* equal to 0. The output includes the total number of pairs (*nfpp*). Now make repeated calls to GQTXF until you have accumulated list elements equal to the total number of pairs.

```
       n=0
       CALL GQTXF (wtype, n, errind, nfpp, font,
      *prec, nchh, minchh, maxchh, nchx, minchx,
      *maxchx, nptxi)
       DO  55 n = 1, nfpp
       CALL GQTXF (wtype, n, errind, nfpp, font,
      *prec, nchh, minchh, maxchh, nchx, minchx,
      *maxchx, nptxi)
       WRITE ( *, 50) font
50     FORMAT (' font', I6)
       WRITE ( *, 52) prec
52     FORMAT (' prec',I6)
55     CONTINUE
```

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
2002	List element or set member not available

GQTXFP

Purpose

GQTXFP	**(errind, font, prec)**
APL code	1453
GKS RCP code	X'38009A00' (939563520)

Function: To inquire text font and precision.

Inquiry function. Returns the current text font and precision from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
font *(returned by GDDM) (fullword integer)*
The integer representing the text font.
prec *(returned by GDDM) (fullword integer)*
The text precision. The possible values are:
0 (GSTRP) String precision
1 (GCHARP) Character precision
2 (GSTRKP) Stroke precision

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text font and precision (GSTXFP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQTXI

Purpose

GQTXI	(errind, txi)
APL code	1430
GKS RCP code	X'38008900' (939559168)

Function: To inquire text index.

Inquiry function. Returns the current text index from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
txi *(returned by GDDM) (fullword integer)*
The current text index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text index (GSTXI)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQTXP

Purpose

GQTXP	**(errind, txp)**
APL code	1511
GKS RCP code	X'38008E00' (939560448)

Function: To inquire text path.

Inquiry function. Returns the current text path from the GKS state list.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.

txp *(returned by GDDM) (fullword integer)*
The current text path. The possible values are:

0 (GRIGHT) Right
1 (GLEFT) Left
2 (GUP) Up
3 (GDOWN) Down

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text path (GSTXP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQTXR

Purpose

GQTXR	**(wkid, txi, type, errind, font, prec, chxp, chsp, coli)**
APL code	1475
GKS RCP code	X'3800B100' (939569408)

Function: To inquire text representation.

Inquiry function. Returns the representation associated with a given *text* index at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
txi *(specified by user) (fullword integer)*
The text index.
type *(specified by user) (fullword integer)*
The type of returned values. The possible values are:
0 (GSET) Set
1 (GREALI) Realized

If *type* equals *set*, the returned values are the same as those originally passed by the Set text representation (GSTXR) function call. If *type* equals *realized*, the returned values are the closest available actual values at the device.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
font *(returned by GDDM) (fullword integer)*
The text font.
prec *(returned by GDDM) (fullword integer)*
The text precision. The possible values are:
0 (GSTRP) String precision
1 (GCHARP) Character precision
2 (GSTRKP) Stroke precision

chxp *(returned by GDDM) (short floating point)*
The character expansion factor.
chsp *(returned by GDDM) (short floating point)*
The character spacing.
coli *(returned by GDDM) (fullword integer)*
The text color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text representation (GSTXR)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
72	Text index is invalid
73	A representation for the specified text index has not been defined on this workstation
2000	Enumeration type out of range

GQTXX

Purpose

GQTXX	**(wkid,px,py,str,errind,cpx,cpy, txexpx,txexpy)**

Function: Inquiry function. Returns *text* extent values for a given text string. Use this call only if your program is written in VS FORTRAN. Otherwise, use the function Inquire text extent (GQTXXS) instead. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.

px *(specified by user) (short floating point)*
py *(specified by user) (short floating point)*
The text position in world coordinates.

str *(specified by user) (character)*
The character string.

errind *(returned by GDDM) (fullword integer)*
The error indicator.

cpx *(returned by GDDM) (short floating point)*
cpy *(returned by GDDM) (short floating point)*
The concatenation point in world coordinates.

txexpx *(returned by GDDM) (array of short floating-point numbers)*
txexpy *(returned by GDDM) (array of short floating-point numbers)*
The text extent parallelogram that the text string occupies, in world coordinates. The parallelogram is returned as four corner points in counterclockwise order, starting at the bottom-left corner.

Operating states

WSOP, WSAC, SGOP

Related functions

Set character height (GSCHH), Set character up vector (GSCHUP), Set character expansion factor (GSCHXP), Set character spacing (GSCHSP), Set text font and precision (GSTXFP), Set text path (GSTXP), Set text alignment (GSTXAL), Set text index (GSTXI), Set text representation (GSTXR), Set aspect source flags (GSASF)

Description

(See the section “GDDM-GKS restrictions” below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The extent of the specified character string is computed using the following current text attributes:

- Set individually or using a bundle table, depending on the corresponding aspect source flags:

  Text font and precision
  Character expansion factor
  Character spacing.

- Set individually:

  Character height
  Character width
  Character up vector
  Character base vector

Text path
Text alignment.

If the current text index is not present in the text bundle table, text index 1 is used.

For *string* and *character* precisions, the text extent parallelogram is the minimum figure that completely encloses the character bodies of the whole string. For *up* and *down* text paths, the widest character body in the font is enclosed. For *stroke* precision, if the character width vector is perpendicular to the character base vector, the text extent parallelogram is a rectangle.

Where **concatenation** is meaningful, you can concatenate a further text output primitive to the character string specified in this function. To do so, use the concatenation point returned by this function as the text position specified in a subsequent Text function call. The concatenation point is affected by the text path and text alignment as follows:

- If the text path is *right* or *left*, the concatenation point is displaced from the text position, in a direction determined by the horizontal component of text alignment. If this component is *left*, the direction is to the right; if it is *right*, the direction is to the left. The magnitude of the displacement is the width of the text extent parallelogram plus one additional character spacing. (The width of the text extent parallelogram is the length of the sides parallel to the character base vector.)
- If the text path is *up* or *down*, the concatenation point is displaced from the text position, in a direction determined by the vertical component of text alignment. If this component is *top* or *cap*, the direction is down; if it is *base* or *bottom*, the direction is up. The magnitude of the displacement is the height of the text extent parallelogram plus one additional character spacing. (The height of the text extent parallelogram is the length of the sides parallel to the character up vector.)

For some combinations of text path and text alignment, concatenation is not meaningful and the returned concatenation point is the same as the text position. These combinations are:

- Text path *left* or *right* and text alignment *center*
- Text path *up* or *down* and text alignment *half*.

Control characters in the string have a workstation-dependent effect, which is the same as their effect in the Text function.

GDDM-GKS restrictions

On the workstations supported by GDDM-GKS, only one control character, X'15', is valid; it is interpreted as a “new line” character. Other control characters included in the string can cause error 101 to be reported in *errind*.

Principal errors

```
  7  GKS not in proper state: GKS shall be in one of the states
     WSOP, WSAC, or SGOP
 20  Specified workstation identifier is invalid
 25  Specified workstation is not open
 39  Specified workstation is neither of category OUTPUT nor of
     category OUTIN
101  Invalid code in string
```

GQTXXS

Purpose

GQTXXS	**(wkid, px, py, lstr, str, errind, cpx, cpy, txexpx, txexpy)**
APL code	1476
GKS RCP code	X'3800B200' (939569664)

Function: To inquire text extent.

Inquiry function. Returns *text* extent values for a given text string. If your program is written in VS FORTRAN, use the function Inquire text extent (VS FORTRAN only) (GQTXX) instead. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
px *(specified by user) (short floating point)*
py *(specified by user) (short floating point)*
The text position in world coordinates.
lstr *(specified by user) (fullword integer)*
The length of string given in the *str* parameter.
str *(specified by user) (character)*
The character string.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
cpx *(returned by GDDM) (short floating point)*
cpy *(returned by GDDM) (short floating point)*
The concatenation point in world coordinates.
txexpx *(returned by GDDM) (array of short floating-point numbers)*
txexpy *(returned by GDDM) (array of short floating-point numbers)*
The text extent parallelogram that the text string occupies, in world coordinates. The parallelogram is returned as four corner points in counterclockwise order, starting at the bottom-left corner.

Operating states

WSOP, WSAC, SGOP

Related functions

Set character height (GSCHH), Set character up vector (GSCHUP), Set character expansion factor (GSCHXP), Set character spacing (GSCHSP), Set text font and precision (GSTXFP), Set text path (GSTXP), Set text alignment (GSTXAL), Set text index (GSTXI), Set text representation (GSTXR), Set aspect source flags (GSASF)

Description

(See the section “GDDM-GKS restrictions” below.)

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

The extent of the specified character string is computed using the following current text attributes:

- Set individually or using a bundle table, depending on the corresponding aspect source flags:

  Text font and precision
  Character expansion factor
  Character spacing

- Set individually:

  Character height
  Character width
  Character up vector
  Character base vector
  Text path
  Text alignment.

If the current text index is not present in the text bundle table, text index 1 is used.

For *string* and *character* precisions, the text extent parallelogram is the minimum figure that completely encloses the character bodies of the whole string. For *up* and *down* text paths, the widest character body in the font is enclosed. For *stroke* precision, if the character width vector is perpendicular to the character base vector, the text extent parallelogram is a rectangle.

Where **concatenation** is meaningful, you can concatenate a further text output primitive to the character string specified in this function. To do so, use the concatenation point returned by this function as the text position specified in a subsequent Text function call. The concatenation point is affected by the text path and text alignment as follows:

- If the text path is *right* or *left*, the concatenation point is displaced from the text position, in a direction determined by the horizontal component of text alignment. If this component is *left*, the direction is to the right; if it is *right*, the direction is to the left. The magnitude of the displacement is the width of the text extent parallelogram plus one additional character spacing. (The width of the text extent parallelogram is the length of the sides parallel to the character base vector.)

- If the text path is *up* or *down*, the concatenation point is displaced from the text position, in a direction determined by the vertical component of text alignment. If this component is *top* or *cap*, the direction is down; if it is *base* or *bottom*, the direction is up. The magnitude of the displacement is the height of the text extent parallelogram plus one additional character spacing. (The height of the text extent parallelogram is the length of the sides parallel to the character up vector.)

For some combinations of text path and text alignment, concatenation is not meaningful and the returned concatenation point is the same as the text position. These combinations are:

- Text path *left* or *right* and text alignment *center*
- Text path *up* or *down* and text alignment *half*.

Control characters in the string have a workstation-dependent effect, which is the same as their effect in the Text (GTXS) function.

GDDM-GKS Restrictions

On the workstations supported by GDDM-GKS, only one control character, X'15', is valid; it is interpreted as a “new line” character. Other control characters included in the string can cause error 101 to be reported in *errind*.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
39	Specified workstation is neither of category OUTPUT nor of category OUTIN
101	Invalid code in string

GQVLS

Purpose

GQVLS	**(wkid, vldnr, mldr, errind, mode, esw, ival, pet, earea, loval, hival, ldr, datrec)**
APL code	1486
GKS RCP code	X'3800BD00' (939572480)

Function: To inquire valuator device state.

Inquiry function. Returns values giving the state of a *valuator* device at a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
vldnr *(specified by user) (fullword integer)*
The valuator device number (1 ... n).
mldr *(specified by user) (fullword integer)*
The dimensions of the data record array.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
mode *(returned by GDDM) (fullword integer)*
The operating mode. The only possible value is:
0 (GREQU) Request
esw *(returned by GDDM) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo
ival *(returned by GDDM) (short floating point)*
The initial value of the device.
pet *(returned by GDDM) (fullword integer)*
The prompt and echo type (1 ... n).
earea *(returned by GDDM) (array of short floating-point numbers)*
The echo area in device coordinates (XMIN, XMAX, YMIN, YMAX).
loval *(returned by GDDM) (short floating point)*
The minimum valuator value.
hival *(returned by GDDM) (short floating point)*
The maximum valuator value.
ldr *(returned by GDDM) (fullword integer)*
The number of array elements used in the data record array.
datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array.

Operating states

WSOP, WSAC, SGOP

Related functions

Unpack data record (GURECS), Initialize valuator (GINVL)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
2001	Output parameter size insufficient

GQWKC

Purpose

GQWKC	(wkid, errind, conid, wtype)
APL code	1467
GKS RCP code	X'3800A900' (939567360)

Function: To inquire workstation connection and type.

Inquiry function. Returns workstation connection and type values from the workstation state list of a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
conid *(returned by GDDM) (fullword integer)*
The connection identifier.
wtype *(returned by GDDM) (fullword integer)*
The workstation type.

Operating states

WSOP, WSAC, SGOP

Related functions

Open workstation (GOPWK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open

GQWKCA

Purpose

GQWKCA	(wtype, errind, wkcat)
APL code	1386
GKS RCP code	X'3800C100' (939573504)

Function: To inquire workstation category.

Inquiry function. Returns the category of the specified type of workstation, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
wkcat *(returned by GDDM) (fullword integer)*
The workstation category. The possible values are:

0	(GOUTPT)	Output
1	(GINPUT)	Input
2	(GOUTIN)	Output/input
3	(GWISS)	Workstation independent segment storage
4	(GMO)	Metafile output
5	(GMI)	Metafile input

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Open workstation (GOPWK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist

GQWKCL

Purpose

GQWKCL	(wtype, errind, vrtype)
APL code	1387
GKS RCP code	X'3800C200' (939573760)

Function: To inquire workstation classification.

Inquiry function. Returns the output class of the specified workstation type, from the workstation description table.

Parameters

wtype *(specified by user) (fullword integer)*
The workstation type.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
vrtype *(returned by GDDM) (fullword integer)*
The workstation category. The possible values are:
0 (GVECTR) Vector
1 (GRASTR) Raster
2 (GOTHWK) Other workstation

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Open workstation (GOPWK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
22	Specified workstation type is invalid
23	Specified workstation type does not exist
39	Specified workstation is neither of category OUTPUT nor of category OUTIN

GQWKDU

Purpose

GQWKDU	(wkid, errind, defmod, regmod, dempty, nframe)
APL code	1469
GKS RCP code	X'3800AB00' (939567872)

Function: To inquire workstation deferral and update states.

Inquiry function. Returns the deferral and implicit regeneration modes of a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
defmod *(returned by GDDM) (fullword integer)*
The deferral mode. The possible values are:
0 (GASAP) As soon as possible
1 (GBNIG) Before the next interaction globally
2 (GBNIL) Before the next interaction locally
3 (GASTI) At some time
regmod *(returned by GDDM) (fullword integer)*
The implicit regeneration mode. The possible values are:
0 (GSUPPD) Suppressed
1 (GALLOW) Allowed
dempty *(returned by GDDM) (fullword integer)*
This parameter indicates whether the display surface is empty. The possible values are:
0 (GNEMPT) Not empty
1 (GEMPTY) Empty
nframe *(returned by GDDM) (fullword integer)*
This parameter indicates whether a new frame action is necessary at update. The possible values are:
0 (GNO) No
1 (GYES) Yes

Operating states

WSAP, WSAC, SGOP

Related functions

Set deferral state (GSDS)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage

GQWKM

Purpose

GQWKM	**(errind, mxopwk, mxacwk, mxwkas)**
APL code	1507
GKS RCP code	X'38008300' (939557632)

Function: To inquire workstation maximum numbers.

Inquiry function. Returns the maximum number of workstations that can be simultaneously open and active, from the GKS description table.

Parameters

errind *(returned by GDDM) (fullword integer)*
The error indicator.
mxopwk *(returned by GDDM) (fullword integer)*
The maximum number of simultaneously open workstations.
mxacwk *(returned by GDDM) (fullword integer)*
The maximum number of simultaneously active workstations.
mxwkas *(returned by GDDM) (fullword integer)*
The maximum number of workstations associated with a segment.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

None

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

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

GQWKS

Purpose

GQWKS	**(wkid, errind, state)**
APL code	1468
GKS RCP code	X'3800AA00' (939567616)

Function: To inquire workstation state.

Inquiry function. Returns the activity state of a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
state *(returned by GDDM) (fullword integer)*
The workstation state. The possible values are:
0 (GINACT) Inactive
1 (GATIV) Active

Operating states

WSOP, WSAC, SGOP

Related functions

Activate workstation (GACWK), Deactivate workstation (GDAWK)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT

GQWKT

Purpose

GQWKT	**(wkid, errind, tus, rwindo, cwindo, rviewp, cviewp)**
APL code	1483
GKS RCP code	X'3800B900' (939571456)

Function: To inquire workstation transformation.

Inquiry function. Returns workstation transformation values for a workstation, from the workstation state list.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
tus *(returned by GDDM) (fullword integer)*
The workstation transformation update state. The possible values are:
0 (GNPEND) Not pending
1 (GPEND) Pending
rwindo *(returned by GDDM) (array of short floating-point numbers)*
The requested workstation window in normalized device coordinates (XMIN, XMAX, YMIN, YMAX).
cwindo *(returned by GDDM) (array of short floating-point numbers)*
The current workstation window in normalized device coordinates (XMIN, XMAX, YMIN, YMAX).
rviewp *(returned by GDDM) (array of short floating-point numbers)*
The requested workstation window in device coordinates (XMIN, XMAX, YMIN, YMAX).
cviewp *(returned by GDDM) (array of short floating-point numbers)*
The current workstation viewport in device coordinates (XMIN, XMAX, YMIN, YMAX).

Operating states

WSOP, WSAC, SGOP

Related functions

Set workstation window (GSWKWN), Set workstation viewport (GSWKVP)

Description

If the inquired information is available, it is returned as output, and *errind* is returned as 0 (zero). If the inquired information is not available, all output is invalid, and *errind* returns an error indicator.

If the workstation transformation update state is *pending*, a workstation transformation change has been requested but not yet performed.

The values for requested and current windows or viewports are different only if the update state is *pending*.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
36	Specified workstation is Workstation Independent Segment Storage

GRDITM

Purpose

GRDITM	**(wkid, midrl, mldr, datrec)**
APL code	1411
GKS RCP code	X'38006700' (939550464)

Function: To read item from GKSM.

Metafile function. Reads the current item from an input *metafile*.

Parameters

wkid *(specified by user) (fullword integer)*
The identifier of the metafile input workstation.
midrl *(specified by user) (fullword integer)*
The maximum item data record length in bytes. If the length of the item data record is greater than this value, the excess parts of the item are lost. You can specify 0 to skip the metafile item.
mldr *(specified by user) (fullword integer)*
The dimension of the item data record array *datrec*.

datrec *(returned by GDDM) (array of 80-byte character tokens)*
The data record array. The contents of metafile item data records are described in Appendix C, “Metafile structure” on page 383.

Operating states

WSOP, WSAC, SGOP

Related functions

Get item type from GKSM (GGTITM).

Description

A GKS *metafile* (GKSM) is a sequential file that behaves like a workstation. This function reads the current item from an input metafile and returns the data to your application program in the data record array. After the item is read, the next item in the metafile becomes the current item.

Note: The content of each GKSM item is not defined as part of the GKS standard, but is specific to GDDM-GKS.

If *datrec* is not large enough to contain the item data record requested, error 2001 is reported.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
34	Specified workstation is not of category MI
162	No item is left in GKS Metafile input
163	Metafile item is invalid
165	Content of item data record is invalid for the specified item type
166	Maximum item data record length is invalid
2001	Output parameter size insufficient

GRENSG

Purpose

GRENSG	(old, new)
APL code	1359
GKS RCP code	X'38003A00' (939538944)

Function: To rename segment.

Segment function. Replaces each occurrence of a segment name with a new name.

Parameters

old *(specified by user) (fullword integer)*
The old segment name.
new *(specified by user) (fullword integer)*
The new segment name.

Operating states

WSOP, WSAC, SGOP

Related functions

Create segment (GCRSG)

Description

This function replaces segment name *old* with segment name *new* in the list of segments in the workstation state list of each open workstation and in the list of segments in the GKS state list. If the old name is the name of the current open segment, the open segment is renamed.

The old segment name may be reused by the application program.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
120	Specified segment name is invalid
121	Specified segment name is already in use
122	Specified segment does not exist

GRQCH

Purpose

GRQCH	(wkid, chdnr, stat, chnr)
APL code	1380
GKS RCP code	X'38005400' (939545600)

Function: To request choice.

Input function. Obtains *request* mode input from a *choice* device at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
chdnr *(specified by user) (fullword integer)*
The choice device identifier. The possible values are

1 The Enter key. The operator presses the enter key to trigger the device. The choice number returned is always 1.

2 The PF keys. The operator presses a PF key to trigger the device. The choice number returned is the number of the PF key pressed. Note that only PF keys 1 to 12 are supported by GDDM-GKS. If the operator presses a PF key in the range 13 through 24 on a workstation equipped with 24 PF keys then the value is mapped to the range 1 through 12.

3 The alphanumeric light pen. The operator selects a light-pen-detectable field by using either a light pen attached to the display, or the cursor select key. You must have created the light-pen-detectable fields by using the appropriate GDDM alphanumeric functions. If the operator selects a field, the choice number returned is 1. You must use GDDM alphanumeric functions to determine which field was selected.

4 The data keys. If data keys are available as a choice device, the device is triggered when the operator presses a data key. The choice number returned is in the range 1 through 255, and is the character code corresponding to the key that was pressed.

5 The mouse or tablet buttons. If a mouse or tablet is available at the workstation, the device is triggered when the operator presses a button on the mouse or tablet puck, or presses the stylus. For a tablet stylus, the choice number returned is 1; for a mouse or tablet puck, it is the number of the button pressed.

stat *(returned by GDDM) (fullword integer)*
The device status. The possible values are (see the section “Description” below, for an explanation of these values):
0 (GNONE) None
1 (GOK) OK
2 (GNCHOI) No choice

chnr *(returned by GDDM) (fullword integer)*
The choice number selected by the operator. This value is returned only if *stat* is returned as *OK*. It is in the range 1 through the maximum number of choice alternatives for the choice device specified by *chdnr*.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize choice (GINCH), Set choice mode (GSCHM), Inquire number of available logical input devices (GQLI), Inquire default choice device data (GQDCH)

Description

This function obtains request mode input from the choice device *chdnr* at workstation *wkid*.

Choice input can be terminated normally by the corresponding trigger (for example, a button on the choice device), or abnormally by a break signal. The status values have the following meanings:

None is returned if the operation is not successful. Either the operator has used the break signal, or an error was detected when you called the function.

OK is returned if the operator uses the trigger, and the choice is valid. The parameter *chdnr* is set to the logical input value that is the current measure of the choice device.

No choice is returned if the operator terminates the input operation without using either the device trigger or the break signal. The input operation can be terminated in this way as follows:

For the PF key, data key, mouse, and tablet choice devices: if the Enter key is pressed.

For the alphanumeric light pen: if the operator presses the Enter key, or a PF or PA key.

Not all choice devices are available at all workstations. You can use the function Inquire number of available logical input devices (GQLI) to determine which devices are available at a particular workstation. You can use the function Inquire default choice device data (GQDCH) to obtain the maximum number of choice alternatives for a particular choice device.

You can control the choice input process by using the functions Initialize choice (GINCH) and Set choice mode (GSCHM). The default mode of a choice device is *request*.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode

GRQLC

Purpose

GRQLC	**(wkid, lcdnr, stat, tnr, px, py)**
APL code	1377
GKS RCP code	X'38005100' (939544832)

Function: To request locator.

Input function. Obtains *request* mode input from a *locator* input device at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
lcdnr *(specified by user) (fullword integer)*
The locator device number.
stat *(returned by GDDM) (fullword integer)*
The status of the device. The possible values are:
0 (GNONE) None
1 (GOK) OK
tnr *(returned by GDDM) (fullword integer)*
The normalization transformation number.
px *(returned by GDDM) (short floating point)*
py *(returned by GDDM) (short floating point)*
The locator position in world coordinates.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize locator (GINLC), Set locator mode (GSLCM).

Description

This function obtains request mode input from the locator device associated with workstation *wkid*.

If the operation is successful, *OK* is returned together with the logical input value that is the current measure of the locator device.

The measure of a locator device consists of a locator position in world coordinates and the normalization transformation number that was used in the conversion to world coordinates.

The normalization transformation is the one with the highest viewport input priority whose window contains the locator position.

If the operation invokes the break facility, the status is returned as *none*. The locator position entered by the operator must be within the workstation viewport.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode

GRQPK

Purpose

GRQPK	(wkid, pkdnr, stat, sgna, pkid)
APL code	1381
GKS RCP code	X'38005500' (939545856)

Function: To request pick.

Input function. Obtains *request* mode input from a *pick* input device at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pkdnr *(specified by user) (fullword integer)*
The pick device number.
stat *(returned by GDDM) (fullword integer)*
The pick device status. The possible values are:
0 (GNONE) None
1 (GOK) OK
2 (GNPICK) No pick
sgna *(returned by GDDM) (fullword integer)*
The name of the segment containing the picked primitive.
pkid *(returned by GDDM) (fullword integer)*
The pick identifier associated with the picked primitive.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize pick (GINPK), Set pick mode (GSPKM).

Description

This function obtains request mode input from the pick device associated with workstation *wkid*.

If the measure of the pick device indicates that a pick was made, status *OK* is returned together with a segment name and a pick identifier that are set according to the current measure of the pick device.

Status *no pick* is returned when the operator has triggered the pick device without selecting a primitive.

If the break facility is invoked by the operator, status *none* is returned.

The pick identifier returned is the one associated with the primitive within the segment that was picked.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
37	Specified workstation is not of category OUTIN
140	Specified input device is not present on workstation

GRQSK

Purpose

GRQSK	(wkid, skdnr, n, stat, tnr, np, pxa, pya)
APL code	1378
GKS RCP code	X'38005200' (939545088)

Function: To request stroke.

Input function. Obtains *request* mode input from a *stroke* input device at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
skdnr *(specified by user) (fullword integer)*
The stroke device number.
n *(specified by user) (fullword integer)*
The maximum number of points that can be returned by the stroke device, in the arrays *px*, *py*.
stat *(returned by GDDM) (fullword integer)*
The device status. The possible values are:
0 (GNONE) None
1 (GOK) OK
tnr *(returned by GDDM) (fullword integer)*
The normalization transformation number that was used for transforming the returned position from normalized device coordinates into world coordinates. This is the normalization transformation with the highest viewport input priority that contains the stroke positions within its viewport.
np *(returned by GDDM) (fullword integer)*
The number of points returned from the stroke device.
pxa *(returned by GDDM) (array of short floating-point numbers)*
pya *(returned by GDDM) (array of short floating-point numbers)*
The world coordinates of the points returned. The number of elements used is given by *np*.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize stroke (GINSK), Set stroke mode (GSSKM)

Description

This function obtains request mode input from the stroke device associated with workstation *wkid*. If the operation is successful, status *OK* is returned together with the logical input value that is the current measure of the stroke device. If the operator invokes the break facility, the status *none* is returned.

The stroke entered by the operator must lie within the workstation viewport; otherwise, an alarm is sounded, and the stroke is rejected and must be reentered. On some devices, the stroke is checked as each point in the stroke is indicated, but on others, the points are checked only after the whole stroke has been entered by the operator.

On the 3270-PC/G and /GX ranges of workstations, if the prompt and echo type is 1 or 4, and points in the stroke are generated automatically as the cursor is moved (as detailed in the decription of the Initialize stroke (GINSK) function), it is possible for the operator to suspend the stream of points, by using the mouse or tablet buttons, then move the cursor to a new position and resume the stroke. While the stream is suspended, no echo is displayed and no points are generated. When the stream is resumed the new points are echoed and added to the stroke. The coordinates of all the points in the stroke are returned in *pxa* and *pya* when the stroke is completed.

On 3179-G workstations, the stroke is terminated by pressing Enter twice with the cursor at the same position.

This function applies the inverse workstation transformation to the position of the device to get the point positions in normalized device coordinates. The stroke point coordinates lie within the viewport of the returned transformation. If viewports overlap so that more than one transformation contains points in the stroke, the transformation with the highest viewport input priority is used. If the priority of the viewport has not been set by using the Set viewport input priority (GSVPIP) function, normalization transformation number zero will be used.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode

GRQST

Purpose

GRQST **(wkid,stdnr,stat,lostr,str)**

Function: Input function. Obtains *request* mode input from a *string* input device at a workstation. Use this call only if your program is written in FORTRAN IV or VS FORTRAN. Otherwise, use the function Request string (GRQSTS) instead. (See the section "GDDM-GKS restrictions" below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.

stdnr *(specified by user) (fullword integer)*
The string device number. GDDM-GKS provides only one string device at a workstation. The string device number is always 1.

stat *(returned by GDDM) (fullword integer)*
The device status. The possible values are:

0 (GNONE) None
1 (GOK) OK

lostr *(returned by GDDM) (fullword integer)*
The length of the returned string.

str *(returned by GDDM) (character)*
The character string. In a VS FORTRAN program, the string can be of variable length. For FORTRAN IV, you must define the string as CHARACTER*80. This string consists of the characters entered by the application user. The length of the string entered must be less than or equal to the input buffer size, which is defined in the Initialize String function call.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize string (FORTRAN only) (GINST), Set string mode (GSSTM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function obtains request mode input from the string device associated with workstation *wkid*.

If the operation is successful, status *OK* is returned together with the logical input value that is the current measure of the string device.

If the operator invokes the break facility, the status is returned as *none*.

GDDM-GKS restrictions

This function can report error 304 (I/O error occurred while sending data to a workstation) if the echo area defined, or used by default, for the string device overlaps an alphanumeric field created by calls to GDDM Base alphanumeric functions.

Principal errors

```
  7  GKS not in proper state: GKS shall be in one of the states
     WSOP, WSAC, or SGOP
 20  Specified workstation identifier is invalid
 25  Specified workstation is not open
 38  Specified workstation is neither of category INPUT nor of
     category OUTIN
140  Specified input device is not present on workstation
141  Input device is not in REQUEST mode
```

GRQSTS

Purpose

GRQSTS	(wkid, stdnr, mstr, stat, lostr, str)
APL code	1382
GKS RCP code	X'38005600' (939546112)

Function: To request string.

Input function. Obtains *request* mode input from a *string* input device at a workstation. If your program is written in FORTRAN IV or VS FORTRAN, use the function Request string (FORTRAN only) (GRQST) instead. (See the section "GDDM-GKS restrictions" below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.

stdnr *(specified by user) (fullword integer)*
The string device number. GDDM-GKS provides only one string device at a workstation. The string device number is always 1.

mstr *(specified by user) (fullword integer)*
The maximum string length. This is the maximum number of characters that can be returned in *str*.

stat *(returned by GDDM) (fullword integer)*
The device status. The possible values are:

0 (GNONE) None
1 (GOK) OK

lostr *(returned by GDDM) (fullword integer)*
The length of the returned string.

str *(returned by GDDM) (character)*
The character string. This string consists of the characters entered by the application user starting at the initial cursor position. The length of the string entered must be less than or equal to the input buffer size, which is defined in the Initialize string (GINSTS) function call.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize string (GINSTS), Set string mode (GSSTM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function obtains request mode input from the string device associated with workstation *wkid*.

If the operation is successful, status *OK* is returned together with the logical input value that is the current measure of the string device.

If the operator invokes the break facility, the status is returned as *none*.

GDDM-GKS Restrictions

This function can report error 304 (I/O error occurred while sending data to a workstation) if the echo area defined, or used by default, for the string device overlaps an alphanumeric field created by calls to GDDM Base alphanumeric functions.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode

GRQVL

Purpose

GRQVL	**(wkid, vldnr, stat, val)**
APL code	1379
GKS RCP code	X'38005300' (939545344)

Function: To request valuator.

Input function. Obtains *request* mode input from a *valuator* input device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
vldnr *(specified by user) (fullword integer)*
The valuator device number.
stat *(returned by GDDM) (fullword integer)*
The status of the device. The possible values are:
0 (GNONE) None
1 (GOK) OK
val *(returned by GDDM) (short floating point)*
The value returned. This is a real number in the range given by the lower and upper limit specified in the Initialize valuator (GINVL) function.

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize valuator (GINVL), Set valuator mode (GSVLM)

Description

(See the section “GDDM-GKS restrictions” below.)

This function obtains request mode input from the valuator device associated with workstation *wkid*. If the operation is successful, status *OK* is returned together with the logical input value that is the current measure of the valuator device. If the operator invokes the break facility, the status is returned as *none*. The value delivered is within the range defined by the Initialize valuator (GINVL) function.

You can control the valuator input process by using the functions Initialize valuator (GINVL) and Set valuator mode (GSVLM).

GDDM-GKS Restrictions

This function can report error 304 (I/O error while sending data to a workstation) if the echo area defined, or used by default, for the valuator device overlaps an alphanumeric field created by calls to GDDM Base alphanumeric functions.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
141	Input device is not in REQUEST mode

GRSGWK

Purpose

GRSGWK	(wkid)
APL code	1310
GKS RCP code	X'38000700' (939525888)

Function: To redraw all segments on workstation.

Control function. Redisplays all stored segments at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier, specified when the workstation was opened.

Operating states

WSOP, WSAC, SGOP

Related functions

Update workstation (GUWK), Inquire dynamic modification of segment attributes (GQDSGA), Inquire dynamic modification of workstation attributes (GQDWKA)

Description

This function executes all pending output and clears the screen. Then, using the last requested workstation window and viewport, it redisplays all visible segments stored for the workstation.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage

GSASF

Purpose

GSASF	(lasf)
APL code	1341
GKS RCP code	X'38002900' (939534592)

Function: To set aspect source flags.

Attribute function. Sets the aspect source flags.

Parameters

lasf *(specified by user) (an array of fullword integers)*
The individual aspect source flags. This is an array of thirteen integers. The possible values for each element are:
0 (GBUNDL) Bundled
1 (GINDIV) Individual

The elements of the list of aspect source flags are in the following order:
Line type
Line-width scale factor
Polyline color index
Marker type
Marker size scale factor
Polymarker color index
Text font and precision
Character expansion factor
Character spacing
Text color index
Fill area interior style
Fill area style index
Fill area color index

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polyline index (GSPLI), Set polymarker index (GSPMI), Set fill area index (GSFAI), Set text index (GSTXI), Inquire aspect source flags (GQASF), Set polyline representation (GSPLR), Set polymarker representation (GSPMR), Set fill area representation (GSFAR), Set text representation (GSTXR)

Description

The values of the aspect source flags determine how GKS selects the attributes for an output primitive. If the value of an aspect source flag is *bundled*, the corresponding attribute is selected from a bundle table of a workstation state list. The bundles contain either default values or values you create using the Set *** representation functions for the various output primitives. If the value of an aspect source flag is *individual*, the corresponding attribute is taken from the GKS state list. The initial (default) value for all the aspect source flags is *individual*.

Principal errors

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
2000 Enumeration type out of range

GSCHH

Purpose

GSCHH	(chh)
APL code	1331
GKS RCP code	X'38001F00' (939532032)

Function: To set character height.

Attribute function. Sets the character height for *text* primitives in world coordinate units.

Parameters

chh *(specified by user) (short floating point)*
The character height in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set character expansion factor (GSCHXP), Inquire character height (GQCHH), Inquire character width (GQCHW)

Description

This function sets the character height for output primitives, in world coordinate units. The “current character height” and “current character width” entries in the GKS state list are set to the value specified by *chh*. The width-to-height ratio of characters when they are displayed at a workstation can be varied using the Set character expansion factor (GSCHXP) function.

Principal errors

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
78 Character height is less than or equal to zero

GSCHM

Purpose

GSCHM	(wkid, chdnr, mode, esw)
APL code	1374
GKS RCP code	X'38004E00' (939544064)

Function: To set choice mode.

Input function. Sets the operating mode and echo switch as required for a *choice* device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
chdnr *(specified by user) (fullword integer)*
The choice device number.
mode *(specified by user) (fullword integer)*
The operating mode. The possible values are:
0 (GREQU) Request
1 (GSAMPL) Sample
2 (GEVENT) Event
esw *(specified by user) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize choice (GINCH), Request choice (GRQCH)

Description

(See the section “GDDM-GKS restrictions” below.)

This function sets the operating mode and echo switch for a given choice input device; these are stored in the workstation state list.

GDDM-GKS Restrictions

Event and *sample* modes are not supported at GKS level 2b. If one of these modes is selected, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
143	EVENT and SAMPLE input mode are not available at this level of GKS
2000	Enumeration type out of range

GSCHSP

Purpose

GSCHSP	(chsp)
APL code	1329
GKS RCP code	X'38001D00' (939531520)

Function: To set character spacing.

Attribute function. Sets the current character spacing.

Parameters

chsp *(specified by user) (short floating point)*
The character spacing, specified as a fraction of the current character height.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set aspect source flags (GSASF), Set character height (GSCHH), Set text font and precision (GSTXFP), Inquire character spacing (GQCHSP)

Description

This function sets the “current character spacing” entry in the GKS state list to the value given by *chsp*. If the aspect source flag for character spacing is *individual*, the value given by *chsp* is used for the display of subsequent text output primitives.

The character spacing value specifies overlap or additional spacing between characters. If the character spacing is zero (the default), characters are drawn one after the other, according to the text path, without any additional space between them. Positive values for *chsp* will insert additional space between characters; negative values will cause adjacent characters to overlap.

The operation of this function depends on the text mode (*string*, *character*, or *stroke* precision). For *string* precision, the character spacing is not used. For *character* and *stroke* precisions, the character spacing is evaluated exactly.

Principal errors

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

GSCHUP

Purpose

GSCHUP	(chux, chuy)
APL code	1332
GKS RCP code	X'38002000' (939532288)

Function: To set character up vector.

Attribute function. Establishes an up direction for *text* string primitive output.

Parameters

chux *(specified by user) (short floating point)*
chuy *(specified by user) (short floating point)*
The x and y values for the character up vector in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text font and precision (GSTXFP), Inquire character up vector (GQCHUP), Inquire character base vector (GQCHB)

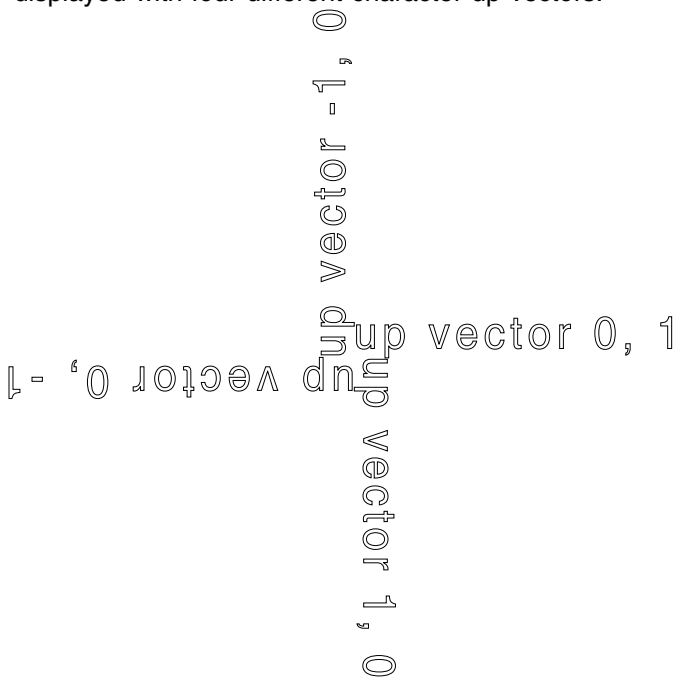
Description

This function defines the up direction for text string primitive output by a vector from the origin (0,0) to the point (*chux, chuy*). The “current character up vector” entry in the GKS state list is set to the value specified. The entry “current character base vector” is set to a vector at right angles, in the clockwise direction, to the character up vector.

If you do not set the character up vector in your program before calling the Text (GTXS) function, a default (0,1) is used.

The operation of this function depends on the text mode (*string*, *character*, or *stroke* precision). For *string* precision, the character up vector is not used. For *character* precision, the vector is used to determine the position of each character but the characters themselves are drawn without rotation. For *stroke* precision, the

attribute is evaluated exactly and the characters are rotated so as to be normal to the character base vector, as in the following illustration, where the string is displayed with four different character up vectors.



Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
79	Length of character up vector is zero

GSCHXP

Purpose

GSCHXP	(chxp)
APL code	1328
GKS RCP code	X'38001C00' (939531264)

Function: To set character expansion factor.

Output attribute. Sets the character expansion factor.

Parameters

chxp *(specified by user) (short floating point)*
The character expansion factor. The font width-to-height ratio for a character is multiplied by the value specified, to give the output width-to-height ratio of that character.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set character height (GSCHH), Inquire character expansion factor (GQCHXP), Inquire character width (GQCHW)

Description

The "current character expansion factor" entry in the GKS state list is set to the value given by *chxp*. The value is used for the display of subsequent text output primitives created when the character expansion factor aspect source flag (ASF) is *individual*. (If the ASF is *bundled*, the value has no effect.)

The character expansion factor affects the width of characters but not the height. When characters are displayed at a workstation, the width is defined by the product of the transformed character width attribute, the character expansion factor, and the font width-to-height ratio for the character. The font width-to-height ratio can vary from character to character according to the font design. The character width attribute is set implicitly when the character height is set using the function Set character height (GSCHH).

Principal errors

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

77 Character expansion factor is less than or equal to zero

GSCLIP

Purpose

GSCLIP	**(clsw)**
APL code	1354
GKS RCP code	X'38003500' (939537664)

Function: To set clipping indicator.

Transformation function. Sets the clipping indicator for clipping to the normalization transformation viewport.

Parameters

clsw *(specified by user) (fullword integer)*
The clipping indicator. The possible values are:
0 (GNCLIP) No clipping
1 (GCLIP) Clipping

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Select normalization transformation (GSELNT), Set viewport (GSVP), Set window (GSWN)

Description

The “clipping indicator” entry in the GKS state list is set to the value given by *clsw*. The clipping indicator determines whether or not subsequent output primitives are clipped to the viewport of the current normalization transformation.

If clipping is on, output primitives are clipped to the intersection of the normalization transformation viewport and the workstation transformation window when they are displayed. If the primitives are in a segment, the segment transformation is applied before clipping.

If clipping is off, output primitives are not clipped at the viewport boundaries. Output primitives that exceed the limits of the workstation window are always clipped.

Output primitives sent to a *metafile output (MO)* workstation are not clipped. If clipping is on, the clipping rectangle defined by the viewport boundary is stored as a metafile item. If clipping is off, the clipping rectangle (0,1,0,1) NDC is stored.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
2000	Enumeration type out of range

GSCR

Purpose

GSCR	**(wkid, ci, cr, cg, cb)**
APL code	1349
GKS RCP code	X'38003000' (939536384)

Function: To set color representation.

Attribute function. Associates a color with a color index at a workstation, using the RGB (red, green, blue) color model.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier
ci *(specified by user) (fullword integer)*
The color table index, at the workstation, defined by this function.
cr *(specified by user) (short floating point)*
A number in the range 0.0 to 1.0 representing the red intensity.
cg *(specified by user) (short floating point)*
A number in the range 0.0 to 1.0 representing the green intensity.
cb *(specified by user) (short floating point)*
A number in the range 0.0 to 1.0 representing the blue intensity.

Operating states

WSOP, WSAC, SGOP

Related functions

Inquire list element of color indexes (GQECI), Inquire color representation (GQCR), Inquire color facilities (GQCF), Inquire predefined color representation (GQPCR), Inquire maximum length of workstation state tables (GQLWK), Inquire dynamic modification of workstation attributes (GQDWKA)

Description

This function associates a color with a color table index at the workstation specified by *wkid*. The function uses the RGB color model, defining a color in terms of red, green, and blue intensities.

If you specify a color that is not available at the workstation, the closest available color is used. The function Inquire color representation (GQCR) returns the realized values on any device.

When you change the color associated with an index, all primitives with that index are drawn in the new color. If you draw a primitive with an undefined color index, the color associated with color index 1 at the workstation is used.

GKS supports the indexes 0 (background) and 1 (foreground) as initial preset values for all devices. You can reset the values for background and foreground by resetting the representations for these indexes to the desired values.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
93	Color index is invalid
96	Color is outside range (0,1)

GSDS

Purpose

GSDS	(wkid, defmod, regmod)
APL code	1312
GKS RCP code	X'38000900' (939526400)

Function: To set deferral state.

Control function. Sets the deferral mode and regeneration mode for a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
defmod *(specified by user) (fullword integer)*
The deferral mode. The possible values are:
0 (GASAP) As soon as possible
1 (GBNIG) Before the next interaction globally
2 (GBNIL) Before the next interaction locally
3 (GASTI) At some time
regmod *(specified by user) (fullword integer)*
The implicit regeneration mode. The possible values are:
0 (GSUPPD) Suppressed
1 (GALLOW) Allowed

Operating states

WSOP, WSAC, SGOP

Related functions

Update workstation (GUWK), Redraw all segments on workstation (GRSGWK), Inquire default deferral state values (GQDDS), Inquire dynamic modification of workstation attributes (GQDWKA), Inquire dynamic modification of segment attributes (GQDSGA), Inquire workstation deferral and update states (GQWKDU)

Description

This function sets the values given by *defmod* and *regmod* in the workstation state list of the workstation specified by *wkid*. Depending on the new value for deferral mode, deferred output at the workstation may be unblocked. If the new value for implicit regeneration mode is *allowed* and a new frame action is required at the workstation, the display is regenerated as in the function Redraw all segments on workstation (GRSGWK).

The deferral mode controls the time at which output functions have a visual effect at a workstation. You can set the deferral mode to *GASAP* or *GBNIG* to ensure that the display is updated before the operator responds to a request for input data by using a logical input device at any workstation. *GBNIL* is used to ensure that the display is updated before the operator responds to a request for input data from

a device at the workstation *wkid*. Use *GASTI* if the display need not be updated before operator interactions. The deferral mode *GASAP* can be used to ensure that the visual effect occurs as soon as possible, but this is workstation-dependent.

The implicit regeneration mode controls the time at which functions requiring an implicit regeneration have their visual effect. If the implicit regeneration mode is *suppressed* and a function requiring implicit regeneration at the workstation is invoked, the “new frame necessary at update” entry in the workstation state list is set to *yes*. The display is not regenerated until you explicitly update the display using Update workstation (GUWK) or Redraw all segments on workstation (GRSGWK), or change the implicit regeneration mode to *allowed*.

If the implicit regeneration mode is *allowed* and an implicit regeneration is required, then it is performed immediately. An implicit regeneration is equivalent to an invocation of Redraw all segments on workstation (GRSGWK).

For example, color changes made by Set color representation (GSCR) can be controlled by this parameter. If the implicit regeneration mode is *suppressed*, a new frame will be redrawn only when an explicit function call to Redraw all segments on workstation (GRSGWK) or Update workstation (GUWK) is issued. However, when the implicit regeneration mode is *allowed*, the workstation will be immediately updated.

For each workstation type, the workstation description table indicates which changes lead to an implicit regeneration and which can be performed immediately. You can use the functions Inquire dynamic modification of workstation attributes (GQDWKA) and Inquire dynamic modification of segment attributes (GQDSGA) to determine dynamic modification capabilities. The following functions can cause an implicit regeneration to be required:

Set polyline representation (GSPLR)
Set polymarker representation (GSPMR)
Set text representation (GSTXR)
Set fill area representation (GSFAR)
Set pattern representation (GSPAR)
Set color representation (GSCR)
Set workstation window (GSWKWN)
Set workstation viewport (GSWKVP)
Set segment transformation (GSSGT)
Set visibility (GSVIS)
Set highlighting (GSHLIT)
Delete segment (GDSG)
Delete segment from workstation (GDSGWK)
Interpret item (GIITM)

In addition, an implicit regeneration may be required when adding primitives to an open segment overlapping a segment of higher priority, using the output primitive functions or Insert segment (GINSG), or if segment priorities affect the execution of the functions Associate segment with workstation (GASGWK) and Set segment priority (GSSGP).

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
2000	Enumeration type out of range

GSDTEC

Purpose

GSDTEC	**(sgna, det)**
APL code	1369
GKS RCP code	X'38004400' (939541504)

Function: To set detectability.

Segment function. Marks a segment as detectable or undetectable.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name, selected when the segment was created.
det *(specified by user) (fullword integer)*
The detectability flag. The possible values are:
0 (GUNDET) Undetectable
1 (GDETEC) Detectable

Operating states

WSOP, WSAC, SGOP

Related functions

Create segment (GCRSG), Set visibility (GSVIS)

Description

This function sets the “detectability” entry in the segment state list of the segment *sgna*. If the segment is marked as *detectable* and *visible*, the primitives in it are available for *pick* input. Segments that are *detectable* but *invisible* cannot be picked. (See Set visibility (GSVIS).)

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
120	Specified segment name is invalid
122	Specified segment does not exist
2000	Enumeration type out of range

GSELNT

Purpose

GSELNT	(tnr)
APL code	1353
GKS RCP code	X'38003400' (939537408)

Function: To select normalization transformation.

Transformation function. Selects the current normalization transformation number.

Parameters

tnr *(specified by user) (fullword integer)*
The transformation number.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set window (GSWN), Set viewport (GSVP), Set clipping indicator (GSCLIP)

Description

This function selects the current normalization transformation number. The current normalization transformation number in the GKS state list is set to the value given by *tnr*. The clipping rectangle in the GKS state list is set to the viewport limits of the selected normalization transformation.

There are 11 transformation numbers defined, from 0 through 10. Transformation 0 is the unity transformation, which always maps WC (0.0, 1.0) x (0.0, 1.0) to NDC (0.0, 1.0) x (0.0, 1.0); the whole of the world coordinate system maps onto the whole of the NDC coordinate system. Transformations 1 through 10 initially default to be the same as transformation 0, and can be reset to other values with the Set window (GSWN) and Set viewport (GSVP) functions.

Once you have defined the normalization transformations you need, GSELNT lets you choose one of them to be the current one. The current transformation is used to transform subsequent output primitives.

Principal errors

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
50 Transformation number is invalid

GSFACI

Purpose

GSFACI	(coli)
APL code	1338
GKS RCP code	X'38002600' (939533824)

Function: To set fill area color index.

Attribute function. Sets the current *fill area* color index for fill area output primitives.

Parameters

coli *(specified by user) (fullword integer)*
The fill area color index

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set aspect source flags (GSASF), Set color representation (GSCR), Fill area (GFA), Inquire fill area color index (GQFACI)

Description

The current fill area color index in the GKS state list is set to the value given by *coli*. This value is used for the display of subsequent fill area output primitives when the fill area color index ASF is *individual*.

You define the color represented by a color index at a particular workstation by calling Set color representation (GSCR). The color index is a pointer into the workstation color table. If the color index *coli* has not been defined at a workstation, color index 1 is used on that workstation.

Principal errors

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
92 Color index is less than zero

GSFAI

Purpose

GSFAI	(index)
APL code	1335
GKS RCP code	X'38002300' (939533056)

Function: To set fill area index.

Output attribute. Sets the current fill area index.

Parameters

index *(specified by user) (fullword integer)*
The fill area index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area representation (GSFAR), Fill area (GFA), Set aspect source flags (GSASF)

Description

This function sets the current fill area index in the GKS state list to the value given by *index*. The fill area index is used for subsequent fill area primitives.

The appearance of fill areas is determined by their color, interior style, and the style of hatch or pattern that fills the interior. Using Set fill area representation (GSFAR), you can define a number of bundles of values for these three attributes at any workstation. The bundles are stored in the fill area bundle table for the workstation. The fill area index is a pointer into the fill area bundle table at each workstation. When a fill area primitive is displayed, attributes from the bundle specified by the fill area index are used according to the corresponding aspect source flag (ASF). You use Set aspect source flags (GSASF) to define whether or not fill area attributes are to be taken from bundles on the output workstations. If the ASF for an attribute is *bundled*, the bundled attribute is used. An aspect source flag exists for each of these fill area attributes:

Fill area color index
Fill area interior style
Fill area style index.

If you do not call this function in your program before calling the Fill area (GFA) function, or if the index you request is not defined or is not available on the workstation, fill area index 1 is used.

Principal errors

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

80 Fill area index is invalid

GSFAIS

Purpose

GSFAIS	(ints)
APL code	1336
GKS RCP code	X'38002400' (939533312)

Function: To set fill area interior style.

Attribute function. Selects the current *fill area* interior style for fill area output primitives.

Parameters

ints *(specified by user) (fullword integer)*
The interior fill area style. The possible values are:

0 (GHOLLO) Hollow
1 (GSOLID) Solid
2 (GPATTR) Pattern
3 (GHATCH) Hatch

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set fill area color index (GSFACI), Set fill area style index (GSFASI), Set aspect source flags (GSASF), Fill area (GFA), Set pattern representation (GSPAR), Inquire fill area interior style (GQFAIS)

Description

This function sets the current fill area interior style in the GKS state list to the value given by *ints*. This value is used for subsequent fill area primitives created when the current fill area style index aspect source flag (ASF) in the GKS state list is *individual*.

There are four available fill area interior styles:

Hollow There is no filling but the boundary of the polygon is closed. If the fill area color index ASF is *individual*, the color indicated by the current fill area color index is used. If the fill area color index ASF is *bundled*, the color indicated by the current fill area index is used. Line type is solid, and line width is the default size.

Solid The interior of the polygon (excluding the boundary) is filled with a single color. If the fill area color index ASF is *individual*, the current fill area color index is used. If the fill area color index ASF is *bundled*, the color indicated by the current fill area index is used.

Pattern Fills the polygon using a pattern. If the fill area style index ASF is *individual*, the pattern indicated by the current fill area style index is used. If the fill area style index ASF is *bundled*, the pattern indicated by the current fill area index is used.

Hatch Fills the polygon using a hatch style. If the fill area style index ASF is *individual*, the hatch style indicated by the current fill area style index is used. If the fill area style index ASF is *bundled*, the hatch style indicated by the current fill area index is used. If the fill area color index ASF is *individual*, the current fill area color index is used. If the fill area color index ASF is *bundled*, the color indicated by the current fill area index is used.

If you do not call this function before using the Fill area (GFA) output function, and the fill area interior style index ASF is *individual*, the default value *hollow* is used. If your program requests an interior style that is not available on a workstation, *hollow* will be used on that workstation.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
2000	Enumeration type out of range

GSFAR

Purpose

GSFAR	**(wkid, fai, ints, styli, coli)**
APL code	1347
GKS RCP code	X'38002E00' (939535872)

Function: To set fill area representation.

Output attribute. Associates a bundle of *fill area* characteristics with a fill area index at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier
fai *(specified by user) (fullword integer)*
The fill area index.
ints *(specified by user) (fullword integer)*
The fill area interior style. The possible values are:
0 (GHOLLO) Hollow
1 (GSOLID) Solid
2 (GPATTR) Pattern

3 (GHATCH) Hatch

styli *(specified by user) (fullword integer)*
The fill area style index. For interior styles *hollow* and *solid* this value is not used. For interior styles *pattern* and *hatch* the value indicates a particular pattern or hatch style at the workstation.

coli *(specified by user) (fullword integer)*
A color index from the color table of the workstation.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Inquire fill area representation (GQFAR), Inquire fill area facilities (GQFAF), Set fill area index (GSFAI), Set aspect source flags (GSASF)

Description

This function associates the fill area index *fai* at workstation *wkid* with the fill area characteristics given by the other parameters. The index and group of characteristics are stored in the fill area bundle table at the workstation. The changes in the fill area bundle table are made visible in the displayed primitives on the workstation.

When a fill area is displayed, the current fill area index refers to an entry in the fill area bundle table. If fill areas are displayed with a fill area index that is not present in the fill area bundle table, fill area index 1 is used. Which of the attributes in the bundle table entry are used depends on the setting of the following fill area attribute aspect source flags:

Fill area interior style ASF
Fill area style index ASF
Fill area color index ASF.

If the fill area interior style ASF is *individual*, the current fill area interior style is used. If the fill area interior style ASF is *bundled*, the fill area interior style from the bundle table entry indicated by the current fill area index is used.

If this function is used to define a representation for the **current** fill area index, the fill area interior style is set by this function. The fill area is then displayed as follows, depending on the fill area interior style determined:

Hollow There is no filling but the boundary of the polygon is closed. If the fill area color index ASF is *individual*, the color indicated by the current fill area color index is used. If the fill area color index ASF is *bundled*, the color index set by this function is used. Line type is solid, and line width is the default size.

Solid The interior of the polygon (excluding the boundary) is filled with a single color. If the fill area color index ASF is *individual*, the current fill area color index is used. If the fill area color index ASF is *bundled*, the color index set by this function is used.

Pattern The polygon is filled with a pattern. If the fill area style index ASF is *individual*, the pattern indicated by the current fill area style index is used. If the fill area style index ASF is *bundled*, the pattern index (*styli*) set by this function is used. The index is a pointer into the pattern table of the

workstation. You can set the referenced pattern table entry using Set pattern representation (GSPAR).

Hatch Fills the polygon using a hatch style. If the fill area style index ASF is *individual*, the hatch style indicated by the current fill area style index is used. If the fill area style index ASF is *bundled*, the hatch style (*styli*) set by this function is used. If the fill area color index ASF is *individual*, the current fill area color index is used. If the fill area color index ASF is *bundled*, the color index set by this function is used.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
80	Fill area index is invalid
83	Specified fill area interior style is not supported on this workstation
85	Specified pattern index is invalid
86	Specified hatch style is not supported on this workstation
93	Color index is invalid
2000	Enumeration type out of range

GSFASI

Purpose

GSFASI	(styli)
APL code	1337
GKS RCP code	X'38002500' (939533568)

Function: To set fill area style index.

Attribute function. Selects the current *fill area* style index for fill area output primitives.

Parameters

styli *(specified by user) (fullword integer)*
The fill area style index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Inquire fill area style index (GQFASI), Set pattern representation (GSPAR)

Description

This function sets the fill area style index for fill area output primitives. The current fill area style index in the GKS state list is set to the value given by *styli*. This value is used for fill area primitives created when the current fill area style index ASF (aspect source flag) in the GKS state list is *individual*.

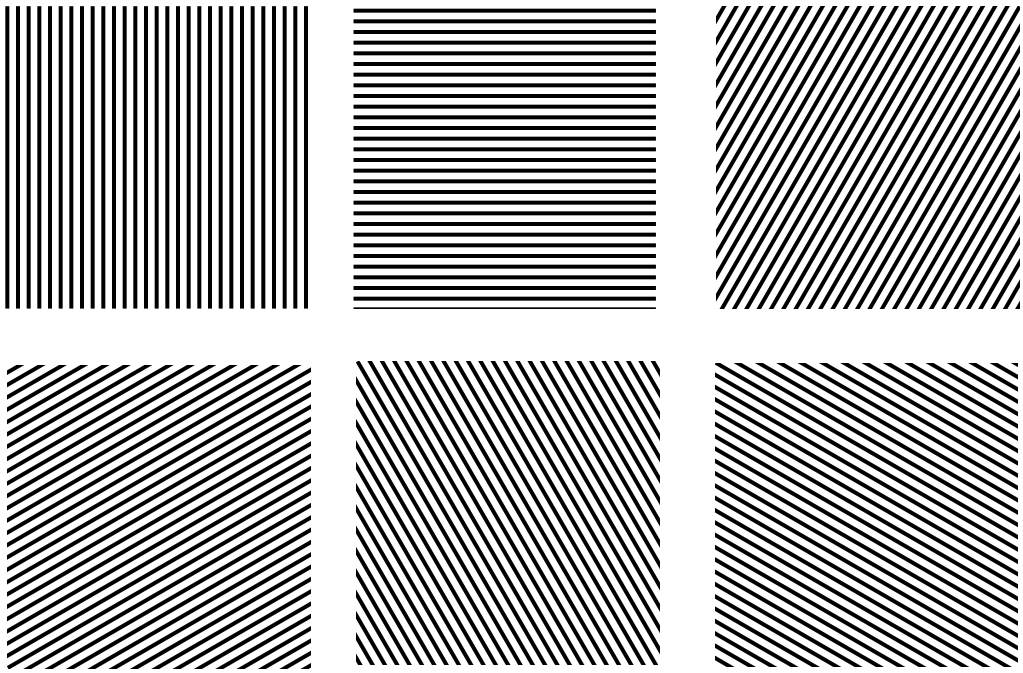
For fill areas that are displayed with interior styles *hollow* and *solid*, the index is not used.

For fill areas displayed with interior style *pattern*, the index *styli* is a pointer into the pattern table of the workstation. You can set the pattern table entry to be referenced at a workstation using Set pattern representation (GSPAR).

For fill areas displayed with interior style *hatch*, the index determines the hatch style used.

GDDM-GKS provides the following hatch styles:

-1 Narrow-spaced vertical lines
-2 Narrow-spaced horizontal lines
-3 Widely-spaced +45 degree lines
-4 Narrow-spaced +45 degree lines
-5 Widely-spaced -45 degree lines
-6 Narrow-spaced -45 degree lines



If you request a style that is not available when the fill area is displayed at a workstation, style index 1 is used at that workstation. If style 1 is not available, the result is workstation-dependent.

Principal errors

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

84 Style (pattern or hatch) index is equal to zero

GSHLIT

Purpose

GSHLIT	(sgna,hil)
APL code	1367
GKS RCP code	'38004200'X (939540992)

Function: Segment function. Marks a segment for normal or highlighted display.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name

hil *(specified by user) (fullword integer)*
The highlighting value. The possible values are:

0 (GNORML) Normal
1 (GHILIT) Highlight

Operating states

WSOP, WSAC, SGOP

Related functions

Create segment (GCRSG), Inquire segment attributes (GQSGA)

Description

This function marks a segment to be drawn with or without highlighting. The highlighting attribute in the segment state list for the segment *sgna* is set to the value given by *hil*.

If the segment is marked as *highlighted* and *visible*, the primitives in it are highlighted when displayed. (See Set visibility (GSVIS).) Highlighting color is dependent upon the type of workstation: on displays it is white, on printers it is black, and on plotters it takes the color associated with the highest available pen number.

Principal errors

```
  7  GKS not in proper state: GKS shall be in one of the states
     WSOP, WSAC, or SGOP
120  Specified segment name is invalid
122  Specified segment does not exist
2000 Enumeration type out of range
```

GSLCM

Purpose

GSLCM	(wkid, skdnr, mode, esw)
APL code	1371
GKS RCP code	X'38004B00' (939543296)

Function: To set locator mode.

Input function. Sets the operating mode and echo switch for a *locator* device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
skdnr *(specified by user) (fullword integer)*
The locator device number.
mode *(specified by user) (fullword integer)*
The operating mode. The possible values are:
0 (GREQU) Request
1 (GSAMPL) Sample
2 (GEVENT) Event
esw *(specified by user) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize locator (GINLC), Request locator (GRQLC)

Description

(See the section “GDDM-GKS restrictions” below.)

This function sets the operating mode and echo switch for a given locator input device; these are stored in the workstation state list.

GDDM-GKS Restrictions

Event and *sample* modes are not supported at GKS level 2b. If one of these modes is selected, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
143	EVENT and SAMPLE input mode are not available at this level of GKS
2000	Enumeration type out of range

GSLN

Purpose

GSLN	(ltype)
APL code	1318
GKS RCP code	X'38001300' (939528960)

Function: To set linetype.

Attribute function. Sets the current line type for *polyline* output primitives.

Parameters

ltype *(specified by user) (fullword integer)*
The identifier for a specific line type. This should be either a negative value indicating an implementation-dependent line type or one of the following:

1 (GLSOLI) Solid
2 (GLDASH) Dash
3 (GLDOT) Dotted
4 (GLDASD) Dash-dotted

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Polyline (GPL), Set aspect source flags (GSASF), Inquire polyline facilities (GQPLF), Inquire linetype (GQLN)

Description

This function selects the current polyline line type. The current line type in the GKS state list is set to the value given by *ltype*. This value is used for the display of subsequent polyline primitives created when the current *linetype* aspect source flag (ASF) is *individual*.

The line types specified by negative values of *ltype* are implementation-dependent and availability is workstation-dependent. The following table shows the values for *ltype* supported by GDDM-GKS.

Value	Line type
-4	Solid line (same as 1)
-3	Dash-double-dot line
-2	Long-dashed line
-1	Double-dotted line
1	Solid
2	Dashed line (short dashes)
3	Dotted line
4	dashed-dotted line

Here are the four line types guaranteed by GKS:

1. solid
2. long-dashed
3. dotted
4. dashed-dotted

If you choose a line type that is not supported at a workstation, line type 1 is used on that workstation.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
63	Linetype is equal to zero

GSLWSC

Purpose

GSLWSC	(lwidth)
APL code	1319
GKS RCP code	X'38001400' (939529216)

Function: To set linewidth scale factor.

Attribute function. Sets the current line width scale factor for *polyline* output primitives.

Parameters

lwidth *(specified by user) (short floating point)*
The line width scale factor.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Polyline (GPL), Inquire linewidth scale factor (GQLWSC), Set aspect source flags (GSASF), Inquire polyline facilities (GQPLF)

Description

This function sets the width scale factor for polyline output primitives. The current line width scale factor in the GKS state list is set to the value given *lwidth*. This value is used for the display of subsequent polyline primitives created when the current linewidth scale factor aspect source flag (ASF) is *individual*. (If the ASF is *bundled*, this value has no effect.) If the scale factor specified is less than zero, error 65 is reported.

When the primitive is displayed at a workstation, the line width scale factor is applied to the nominal line width. This value is then mapped to the nearest line width available on the workstation.

If you do not call this function in your program before you call the Polyline (GPL) function, GKS uses a default value of 1.0.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
65	Linewidth scale factor is less than zero

GSMCH

Purpose

GSMCH	**(wkid, chdnr, stat, chnr)**
APL code	1385
GKS RCP code	X'38005A00' (939547136)

Function: To sample choice.

Input function. Obtains *sample* mode input from a *choice* input device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
chdnr *(specified by user) (fullword integer)*
The choice device number.
stat *(returned by GDDM) (fullword integer)*
The device status. The possible values are:
1 (GOK) OK
2 (GNCHOI) No Choice
chnr *(returned by GDDM) (fullword integer)*
The choice number selected.

Operating states

WSOP, WSAC, SGOP

Related functions

Set choice mode (GSCHM)

Description

(See the section “GDDM-GKS restrictions” below.)

This function obtains sample mode input from the choice device associated with workstation *wkid*. The logical input value, which is the current measure of the choice device, is returned.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
142	Input device is not in SAMPLE mode
143	EVENT and SAMPLE input mode are not available at this level of GKS

GSMK

Purpose

GSMK	(mtype)
APL code	1322
GKS RCP code	X'38001700' (939529984)

Function: To set marker type.

Attribute function. Selects the current marker type for *polymarker* output primitives.

Parameters

mtype *(specified by user) (fullword integer)*
The marker type identifier. This should be either a negative value indicating an implementation-dependent marker type, or one of the following:

1 (GPOINT) Small dot
2 (GPLUS) Plus
3 (GAST) Asterisk
4 (GOMARK) Circle
5 (GXMARK) Diagonal cross

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Polymarker (GPM), Inquire marker type (GQMK), Set aspect source flags (GSASF), Inquire polymarker facilities (GQPMF)

Description

The current marker type in the GKS state list is set to the value given by *mtype*. This value is used for the display of subsequent polymarker output primitives created when the current marker type aspect source flag (ASF) is *individual*. (If the ASF is *bundled*, this value has no effect.)

The marker types specified by negative values of *mtype* are implementation-dependent and availability is workstation-dependent. The following table shows the values for *mtype* supported by GDDM-GKS.

rules=no split=yes.

Value	Symbol
-6	Shaded box
-5	Shaded diamond
-4	Asterisk (6 points)
-3	Hollow box
-2	Hollow diamond
-1	Scalable dot
1	Dot
2	Plus
3	Asterisk (8 points)
4	Circle
5	Diagonal cross

Marker type 1 is not scaled; when it is drawn at a workstation, GDDM-GKS ensures that the smallest clearly-visible dot is used.

Here are the five marker types guaranteed by GKS:

1. dot
2. plus 
3. asterisk 
4. circle 
5. diagonal cross 

If you choose a marker type that is unavailable at a workstation, marker type 3 is used at that workstation.

Principal errors

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

69 Marker type is equal to zero

GSMKSC

Purpose

GSMKSC	(mszsf)
APL code	1323
GKS RCP code	X'38001800' (939530240)

Function: To set marker size scale factor.

Attribute function. Sets the current marker size scale factor for *polymarker* output primitives.

Parameters

mszsf *(specified by user) (short floating point)*
The marker size scale factor.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Polymarker (GPM), Inquire polymarker facilities (GQPMF), Inquire marker size scale factor (GQMKSC)

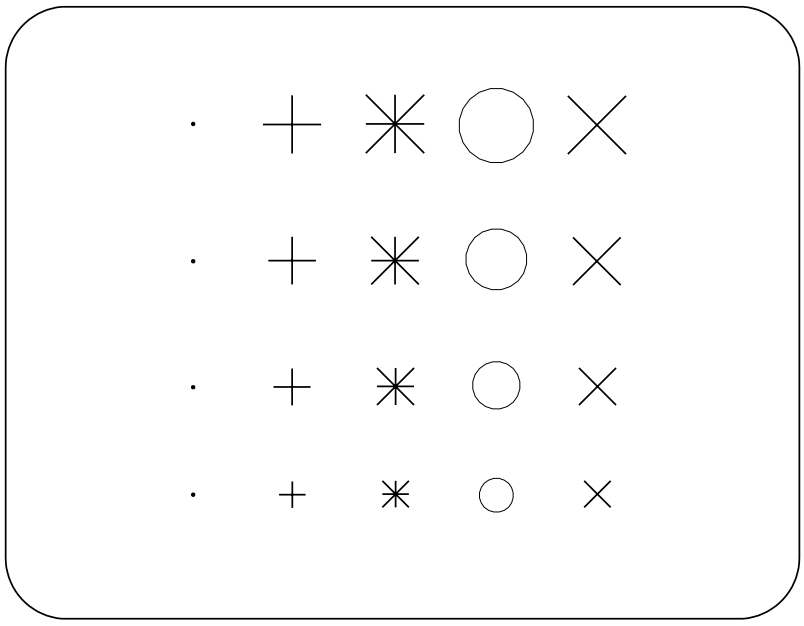
Description

The current marker size scale factor in the GKS state list is set to the value given by *mszsf*. This value is used for the display of subsequent *polymarker* output primitives created when the current marker size scale factor aspect source flag (ASF) is *individual*. (If the ASF is *bundled*, this value has no effect.) If you set the scale factor to less than zero, error 71 is reported.

The number of available marker sizes is workstation-dependent. When the primitive is displayed at a workstation, the marker size scale factor is applied to the nominal marker size. If the resulting marker size is not available on the workstation, the value is mapped to the closest supported value.

You can find the nominal marker size by calling Inquire polymarker facilities (GQPMF).

In this illustration, each of the five marker types is displayed with four different marker size scale factors.



If you do not call this function in your program before you call the Polymarker (GPM) function, GKS uses a default value of 1.0.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
71	Marker size scale factor is less than zero

GSMLC

Purpose

GSMLC	(wkid, lcdnr, tnr, lpx, lpy)
APL code	1510
GKS RCP code	X'38005700' (939546368)

Function: To sample locator.

Input function. Obtains *sample* mode input from a *locator* input device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The identifier selected when the workstation was opened.
lcdnr *(specified by user) (fullword integer)*
The locator device number.
tnr *(returned by GDDM) (fullword integer)*
The normalization transformation number.
lpx *(returned by GDDM) (short floating point)*
lpy *(returned by GDDM) (short floating point)*
The locator position in world coordinates.

Operating states

WSOP, WSAC, SGOP

Related functions

Set locator mode (GSLCM)

Description

(See the section “GDDM-GKS restrictions” below.)

This function obtains sample mode input from the locator device associated with workstation *wkid*. The logical input value, which is the current measure of the locator device, is returned.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
142	Input device is not in SAMPLE mode
143	EVENT and SAMPLE input mode are not available at this level of GKS

GSMPK

Purpose

GSMPK	**(wkid, pkdnr, stat, sgna, pkid)**
APL code	1403
GKS RCP code	X'38005B00' (939547392)

Function: To sample pick.

Input function. Obtains *sample* mode input from a *pick* input device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pkdnr *(specified by user) (fullword integer)*
The pick device number.
stat *(returned by GDDM) (fullword integer)*
The pick device status. The possible values are:
0 (GOK) OK
1 (GNPICK) No pick
sgna *(returned by GDDM) (fullword integer)*
The segment name.
pkid *(returned by GDDM) (fullword integer)*
The pick identifier.

Operating states

WSOP, WSAC, SGOP

Related functions

Set pick mode (GSPKM)

Description

(See the section “GDDM-GKS restrictions” below.)

This function obtains sample mode input from the pick device associated with workstation *wkid*.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
37	Specified workstation is not of category OUTIN
140	Specified input device is not present on workstation
142	Input device is not in SAMPLE mode
143	EVENT and SAMPLE input mode are not available at this level of GKS

GSMSK

Purpose

GSMSK	**(wkid, skdnr, n, tnr, np, px, py)**
APL code	1383
GKS RCP code	X'38005800' (939546624)

Function: To sample stroke.

Input function. Obtains *sample* mode input from a *stroke* input device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
skdnr *(specified by user) (fullword integer)*
The stroke device number.
n *(specified by user) (fullword integer)*
The maximum number of points that can be returned.
tnr *(returned by GDDM) (fullword integer)*
The normalization transformation number.
np *(returned by GDDM) (fullword integer)*
The number of points that have been returned.

px *(returned by GDDM) (array of short floating-point numbers)*
py *(returned by GDDM) (array of short floating-point numbers)*
The points in world coordinates.

Operating states

WSOP, WSAC, SGOP

Related functions

Set stroke mode (GSSKM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function obtains sample mode input from the stroke device associated with workstation *wkid*. The logical input value, which is the current measure of the stroke device, is returned.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
142	Input device is not in SAMPLE mode
143	EVENT and SAMPLE input mode are not available at this level of GKS

GSMST

Purpose

GSMST **(wkid,stdnr,lostr,str)**

Function: Input function. Obtains *sample* mode input from a *string* input device at a workstation. Use this call only if your program is written in FORTRAN IV or VS FORTRAN. Otherwise, use the function Sample string (GSMSTS) instead. (See the section "GDDM-GKS restrictions" below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.

stdnr *(specified by user) (fullword integer)*
The string device number.

lostr *(returned by GDDM) (fullword integer)*
The number of characters returned.

str *(returned by GDDM) (character)*
The string. In a VS FORTRAN program, the string can be of variable length. For FORTRAN IV, you must define the string as CHARACTER*80.

Operating states

WSOP, WSAC, SGOP

Related functions

Set string mode (GSSTM)

Description

(See the section “GDDM-GKS restrictions” below.)

This function obtains sample mode input from the string device associated with workstation *wkid*.

GDDM-GKS restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

```
  7  GKS not in proper state: GKS shall be in one of the states
     WSOP, WSAC, or SGOP
 20  Specified workstation identifier is invalid
 25  Specified workstation is not open
 38  Specified workstation is neither of category INPUT nor of
     category OUTIN
140  Specified input device is not present on workstation
142  Input device is not in SAMPLE mode
143  EVENT and SAMPLE input mode are not available at this level
     of GKS
```

GSMSTS

Purpose

GSMSTS	(wkid, stdnr, mstr, lostr, str)
APL code	1389
GKS RCP code	X'38005C00' (939547648)

Function: To sample string.

Input function. Obtains *sample* mode input from a *string* input device at a workstation. If your program is written in FORTRAN IV or VS FORTRAN, use the function Sample string (FORTRAN only) (GSMST) instead. (See the section "GDDM-GKS restrictions" below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
stdnr *(specified by user) (fullword integer)*
The string device number.
mstr *(specified by user) (fullword integer)*
The maximum number of characters in *str.*
lostr *(returned by GDDM) (fullword integer)*
The number of characters returned.
str *(returned by GDDM) (character)*
The string.

Operating states

WSOP, WSAC, SGOP

Related functions

Set string mode (GSSTM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function obtains sample mode input from the string device associated with workstation *wkid*.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN

140	Specified input device is not present on workstation
142	Input device is not in SAMPLE mode
143	EVENT and SAMPLE input mode are not available at this level of GKS

GSMVL

Purpose

GSMVL	(wkid, vldnr, val)
APL code	1384
GKS RCP code	X'38005900' (939546880)

Function: To sample valuator.

Input function. Obtains *sample* mode input from a *valuator* input device. (See the section "GDDM-GKS restrictions" below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
vldnr *(specified by user) (fullword integer)*
The valuator device number.
val *(returned by GDDM) (short floating point)*
The value returned.

Operating states

WSOP, WSAC, SGOP

Related functions

Set valuator mode (GSVLM)

Description

(See the section "GDDM-GKS restrictions" below.)

This function obtains sample mode input from the valuator device associated with workstation *wkid*.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
142	Input device is not in SAMPLE mode
143	EVENT and SAMPLE input mode are not available at this level of GKS

GSPA

Purpose

GSPA	(szx, szy)
APL code	1339
GKS RCP code	X'38002700' (939534080)

Function: To set pattern size.

Attribute function. Sets the pattern height and width. (See the section "GDDM-GKS restrictions" below.)

Parameters

szx *(specified by user) (short floating point)*
szy *(specified by user) (short floating point)*
The pattern size in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set pattern reference point (GSPARF), Set pattern representation (GSPAR), Fill area (GFA), Inquire pattern size (GQPA)

Description

(See the section "GDDM-GKS restrictions" below.)

This function sets the pattern height and width for *fill area* output primitives. The current pattern width vector in the GKS state list is set to the vector (*szx*,0). The current pattern height vector in the GKS state list is set to the vector (0,*szy*).

When the fill area interior style is *pattern*, these vectors are used with the current pattern reference point for displaying the patterned fill area output primitives. The pattern width and height vector form the two sides of a pattern box. This lower left

corner of the box is positioned on the pattern reference point that you can set in the Set pattern reference point (GSPARF) function. The box is divided into several color cells defined in the Set pattern representation (GSPAR) function. GKS replicates the pattern box to cover the entire interior of the fill area.

GDDM-GKS Restrictions

On the graphics devices supported by GDDM-GKS, patterns are not transformed, and the pattern size and pattern reference point attributes are not honored.

Principal errors

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
87 Pattern size value is not positive

GSPAR

Purpose

GSPAR	**(wkid, pai, dimx, dimy, isc, isr, dx, dy, colia)**
APL code	1348
GKS RCP code	X'38002F00' (939536128)

Function: To set pattern representation.

Attribute function. Defines the pattern array associated with a pattern index at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pai *(specified by user) (fullword integer)*
The pattern index number.
dimx *(specified by user) (fullword integer)*
dimy *(specified by user) (fullword integer)*
The dimensions of the color index array *colia*.
isc *(specified by user) (fullword integer)*
isr *(specified by user) (fullword integer)*
The start column and start row of the pattern array in the color index array *colia*.
dx *(specified by user) (fullword integer)*
dy *(specified by user) (fullword integer)*
The number of columns and rows of the pattern array.
colia *(specified by user) (an array of fullword integers)*
The color index array (*dimx* groups of *dimy* elements).

Operating states

WSOP, WSAC, SGOP

Related functions

Inquire pattern representation (GQPAR), Inquire fill area facilities (GQFAF), Set fill area representation (GSFAR), Set fill area style index (GSFASI)

Description

The pattern index number and pattern array are stored in the pattern table (in the workstation state list) of the workstation *wkid*. The pattern array is used when *fill area* output primitives are displayed at the workstation with fill area interior style *pattern*, and the fill area style index is *pai*.

The pattern array is the array of color indexes with dimensions (*dx*,*dy*) passed in the color index array *colia*. The parameters *isc* and *isr* give the start column and start row of the pattern array in *colia*. To pass the entire color index array as the pattern array, *isc* and *isr* should be 1 and *dx* and *dy* should be equal to the dimensions of the color index array, *dimx* and *dimy*.

If fill areas are displayed with a pattern index that is not available at a workstation, pattern index 1 is used. If pattern index 1 is not available (that is, patterns are not supported at the workstation) the fill area is shaded with cross hatching using the fill area color index and default line type and width.

If a pattern representation is set by this function, the changes may affect the displayed primitives by causing an implicit regeneration of the display, depending on the workstation implicit regeneration mode.

GDDM-GKS Restrictions

Patterns are not available on plotter workstations supported by GDDM-GKS.

Patterns are not available on IBM 5080 workstations.

On other graphics devices, the pattern size is fixed (patterns are not transformed and the pattern size attribute is ignored) and patterns have a fixed number of rows and columns. The values you specify for the pattern array are truncated or replicated at the workstation to fit the fixed pattern size of the device.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
85	Specified pattern index is invalid
90	Interior style PATTERN is not supported on this workstation
91	Dimensions of color array are invalid
93	Color index is invalid

GSPARF

Purpose

GSPARF	(rfx, rfy)
APL code	1340
GKS RCP code	X'38002800' (939534336)

Function: To set pattern reference point.

Attribute function. Sets the pattern reference point for *fill area* output primitives. (See the section “GDDM-GKS restrictions” below.)

Parameters

rfx *(specified by user) (short floating point)*
rfy *(specified by user) (short floating point)*
The pattern reference point in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Fill area (GFA), Inquire pattern reference point (GQPARF), Set pattern size (GSPA)

Description

(See the section “GDDM-GKS restrictions” below.)

This function sets the location of the lower left corner of the pattern rectangle for subsequent fill area output primitives. The current pattern reference point in the GKS state list is set to the coordinates given by *rfx* and *rfy*.

When the fill area interior style is *pattern*, the pattern reference point is used with the pattern width and height vectors for displaying the patterned fill area output primitives. The pattern width and height vectors form two sides of a pattern box. The lower left corner of this box is positioned on the pattern reference point. The box is several color cells defined in the Set pattern representation (GSPAR) function. GKS replicates the pattern box to cover the entire interior of the fill area. The pattern reference point is subject to the same transformations as the fill area primitive that it is bound to.

GDDM-GKS Restrictions

The pattern reference point attribute is not used when fill areas are displayed at any of the GDDM-GKS supported graphics devices.

Principal errors

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

GSPKID

Purpose

GSPKID	(pkid)
APL code	1342
GKS RCP code	X'38002A00' (939534848)

Function: To set pick identifier.

Attribute function. Sets the current *pick* identifier.

Parameters

pkid *(specified by user) (fullword integer)*
The pick identifier.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set pick mode (GSPKM)

Description

This function sets the current pick identifier. The “current pick identifier” entry in the GKS state list is set to the value given by *pkid*. During pick input, the application program identifies a primitive by the pick identifier that was current when the primitive was created.

Primitives in the same segment or in other segments may have the same pick identifier. If this function is not called before output primitive functions in your program, the default pick identifier value of zero is used.

Principal errors

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
97 Pick identifier is invalid

GSPKM

Purpose

GSPKM	(wkid, pkdnr, mode, esw)
APL code	1375
GKS RCP code	X'38004F00' (939544320)

Function: To set pick mode.

Input function. Sets the operating mode and echo switch for a *pick* device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pkdnr *(specified by user) (fullword integer)*
The pick device number.
mode *(specified by user) (fullword integer)*
The operating mode. The possible values are:
0 (GREQU) Request
1 (GSAMPL) Sample
2 (GEVENT) Event
esw *(specified by user) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize pick (GINPK), Request pick (GRQPK)

Description

(See the section “GDDM-GKS restrictions” below.)

This function sets the operating mode and echo switch for a given pick input device; these are stored in the workstation state list.

GDDM-GKS Restrictions

Event and *sample* modes are not supported at GKS level 2b. If one of these modes is selected, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
37	Specified workstation is not of category OUTIN
140	Specified input device is not present on workstation
143	EVENT and SAMPLE input mode are not available at this level of GKS
2000	Enumeration type out of range

GSPLCI

Purpose

GSPLCI	(coli)
APL code	1320
GKS RCP code	X'38001500' (939529472)

Function: To set polyline color index.

Attribute function. Sets the current *polyline* color index for polyline output primitives.

Parameters

coli *(specified by user) (fullword integer)*
The polyline color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Polyline (GPL), Set aspect source flags (GSASF), Inquire polyline color index (GQPLCI), Set color representation (GSCR)

Description

This function sets the current polyline color index in the GKS state list to the value given by *coli*. This value is used for the display of all subsequent polyline primitives created when the polyline color index aspect source flag (ASF) is *individual*. (If the ASF is *bundled*, the value has no effect.)

You can define the color indicated by a particular color index at a workstation by calling Set color representation (GSCR).

If you choose a color index that is not available at a workstation, color index 1 is used on that workstation.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
92	Color index is less than zero

GSPLI

Purpose

GSPLI	(pli)
APL code	1317
GKS RCP code	X'38001200' (939528704)

Function: To set polyline index.

Attribute function. Selects a bundle of *polyline* attributes to be used for polyline output primitives.

Parameters

pli *(specified by user) (fullword integer)*
The polyline index. A number that represents a bundle of polyline output attributes at the workstation.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Inquire polyline index (GQPLI), Set polyline representation (GSPLR)

Description

The current polyline index in the GKS state list is set to the value given by *pli*. This value is used when creating subsequent polyline primitives.

The appearance of polylines is determined by their color, and by the current line type and line width scale factor. Using Set polyline representation (GSPLR), you can define a number of bundles of values for these three attributes at any workstation. The bundles are stored in the polyline bundle table for the workstation. The polyline index is a pointer into the polyline bundle table at each workstation. When a polyline primitive is displayed, attributes from the bundle specified by the polyline index are used according to the corresponding aspect source flag (ASF). You use Set aspect source flags (GSASF) to define whether or not polyline attributes are to be taken from bundles on the output workstations. If the ASF for an attribute is *bundled*, the bundled attribute is used. An aspect source flag exists for each of these attributes:

Polyline color index
Line type

Line width scale factor.

If you do not call this function in your program before calling the Polyline (GPL) function, or if the index you request is not defined or is not available on the workstation, polyline index 1 is used.

Principal errors

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
60 Polyline index is invalid

GSPLR

Purpose

GSPLR	(wkid, pli, ltype, lwidth, coli)
APL code	1344
GKS RCP code	X'38002B00' (939535104)

Function: To set polyline representation.

Attribute function. Creates a bundle of *polyline* attributes and associates them with a polyline index at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
pli *(specified by user) (fullword integer)*
The polyline index number to represent the group of characteristics specified in the other parameters of this function.
ltype *(specified by user) (fullword integer)*
The identifier for a specific line type. This should be either a negative value, indicating an implementation-dependent line type, or one of the following:
1 (GLSOLI) Solid
2 (GLDASH) Dash
3 (GLDOT) Dotted
4 (GLDASD) Dash-dotted
lwidth *(specified by user) (short floating point)*
The line-width scale factor. A scale factor applied to the workstation nominal line width. The result is mapped by the workstation to the nearest line width available.
coli *(specified by user) (fullword integer)*
The polyline color index. A pointer into the color table at the workstation.

Operating states

WSOP, WSAC, SGOP

Related functions

Inquire polyline representation (GQPLR), Polyline (GPL), Set polyline index (GSPLI), Set aspect source flags (GSASF), Inquire polyline facilities (GQPLF)

Description

This function creates a bundle of attribute values and associates them with polyline index *pli* at workstation *wkid*. The bundle is stored in the polyline bundle table of the workstation.

The polyline bundle table in the workstation state list has predefined entries taken from the workstation description table when the workstation is opened. Any table entry (including the predefined entries) may be redefined with this function.

When a polyline is displayed, the current polyline index refers to an entry in the polyline bundle table. If polylines are displayed with a polyline index that is not present in the polyline bundle table, polyline index 1 is used. Which of the attributes in the bundle table entry are used depends on the setting of the following polyline attribute aspect source flags:

Line type ASF
Line-width scale factor ASF
Polyline color index ASF.

If a polyline representation is set by this function, the changes may affect the displayed primitives by causing an implicit regeneration of the display, depending on the workstation implicit regeneration mode.

The line types specified by negative values of *ltype* are implementation-dependent and availability is workstation-dependent. The following table shows the values for *ltype* supported by GDDM-GKS.

Value	Line type
-4	Solid line (same as 1)
-3	Dash-double-dot line
-2	Long-dashed line
-1	Double-dotted line
1	Solid
2	Dashed line (short dashes)
3	Dotted line
4	dash-dotted line

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
60	Polyline index is invalid

63	Linetype is equal to zero
64	Specified linetype is not supported on this workstation
65	Linewidth scale factor is less than zero
93	Color index is invalid

GSPMCI

Purpose

GSPMCI	(coli)
APL code	1324
GKS RCP code	X'38001900' (939530496)

Function: To set polymarker color index.

Attribute function. Sets the current *polymarker* color index for polymarker output primitives.

Parameters

coli *(specified by user) (fullword integer)*
The polymarker color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Polymarker (GPM), Inquire polymarker color index (GQPMCI), Set aspect source flags (GSASF), Set color representation (GSCR)

Description

This function sets the current polymarker color index in the GKS state list to the value given by *coli*. This value is used for the display of all subsequent polymarker primitives created when the polymarker color index aspect source flag is *individual*.

You can define the color indicated by a particular color index at a workstation by calling Set color representation (GSCR). If you choose a color index that is not available at a workstation, color index 1 is used on that workstation.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
92	Color index is less than zero

GSPMI

Purpose

GSPMI	(pmi)
APL code	1321
GKS RCP code	X'38001600' (939529728)

Function: To set polymarker index.

Attribute function. Selects a bundle of *polymarker* attributes to be used for subsequent polymarker output primitives.

Parameters

pmi *(specified by user) (fullword integer)*
The polymarker index. A number that represents a bundle of polymarker output attributes at the workstation.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set polymarker representation (GSPMR), Inquire polymarker index (GQPMI)

Description

The current polymarker index in the GKS state list is set to the value given by *pmi*. This value is used when creating subsequent polymarker primitives.

The appearance of polymarkers is determined by their color, and by the current marker type and marker size scale factor. Using Set polymarker representation (GSPMR), you can define a number of bundles of values for these three attributes at any workstation. The bundles are stored in the polymarker bundle table for the workstation. The polymarker index is a pointer into the polymarker bundle table at each workstation. When a polymarker primitive is displayed, attributes from the bundle specified by the polymarker index are used according to the corresponding aspect source flag (ASF). You use Set aspect source flags (GSASF) to define whether or not polymarker attributes are to be taken from bundles for the output workstations. If the ASF for an attribute is *bundled*, the bundled attribute is used. An aspect source flag exists for each of these polymarker attributes:

Polymarker color index
Marker type
Marker size scale factor.

If you do not call this function in your program before calling the Polymarker (GPM) function, or if the index you request is not defined or is not available on the workstation, polymarker index 1 is used.

Principal errors

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

66 Polymarker index is invalid

GSPMR

Purpose

GSPMR	(wkid, pmi, mtype, mszsf, coli)
APL code	1345
GKS RCP code	X'38002C00' (939535360)

Function: To set polymarker representation.

Attribute function. Creates a bundle of *polymarker* attributes and associates them with a polymarker index at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier

pmi *(specified by user) (fullword integer)*
The polymarker index number to represent the group of characteristics specified in the other parameters of this function.

mtype *(specified by user) (fullword integer)*
The marker type identifier. This should be either a negative value, indicating an implementation-dependent marker type, or one of the following:

1 (GPOINT) Small dot
2 (GPLUS) Plus
3 (GAST) Asterisk
4 (GOMARK) Circle
5 (GXMARK) Diagonal cross

mszsf *(specified by user) (short floating point)*
The marker size scale factor. A scale factor applied to the workstation nominal marker size. The result is mapped by the workstation to the nearest available marker size.

coli *(specified by user) (fullword integer)*
The polymarker color index. A pointer into the color table at the workstation.

Operating states

WSOP, WSAC, SGOP

Related functions

Inquire polymarker representation (GQPMR), Polymarker (GPM), Set polymarker index (GSPMI), Set aspect source flags (GSASF), Inquire polymarker facilities (GQPMF)

Description

This function creates a bundle of attribute values and associates them with polymarker index *pmi* at workstation *wkid*. The bundle is stored in the polymarker bundle table of the workstation.

The polymarker bundle table in the workstation state list has predefined entries taken from the workstation description table when the workstation is opened. Any table entry (including the predefined entries) may be redefined using this function.

When a polymarker is displayed, the current polymarker index refers to an entry in the polymarker bundle table. If polymarkers are displayed with a polymarker index that is not present in the polymarker bundle table, polymarker index 1 is used. Which of the attributes in the bundle table entry are used depends on the setting of the following polymarker attribute aspect source flags:

Marker type ASF
Marker size scale factor ASF
Polymarker color index ASF.

If a polymarker representation is set by this function, the changes can affect the displayed primitives, possibly causing an implicit regeneration of the display, depending on the workstation implicit regeneration mode.

The marker types specified by negative values of *mtype* are implementation-dependent and availability is workstation-dependent. The following table shows the values for *mtype* supported by GDDM-GKS.

Value	Symbol
-6	Shaded box
-5	Shaded diamond
-4	Asterisk (6 points)
-3	Hollow box
-2	Hollow diamond
-1	Scalable dot
1	Dot
2	Plus
3	Asterisk (8 points)
4	Circle
5	Diagonal cross

Marker type 1 is not scaled; when it is drawn at a workstation, GDDM-GKS ensures that the smallest clearly-visible dot is used.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI

35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
66	Polymarker index is invalid
69	Marker type is equal to zero
70	Specified marker type is not supported on this workstation
71	Marker size scale factor is less than zero
93	Color index is invalid

GSSGP

Purpose

GSSGP	(sgna, prior)
APL code	1368
GKS RCP code	X'38004300' (939541248)

Function: To set segment priority.

Segment function. Sets the priority for a segment.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name.
prior *(specified by user) (short floating point)*
The segment priority.

Operating states

WSOP, WSAC, SGOP

Related functions

Create segment (GCRSG), Inquire segment attributes (GQSGA), Inquire dynamic modification of segment attributes (GQDSGA)

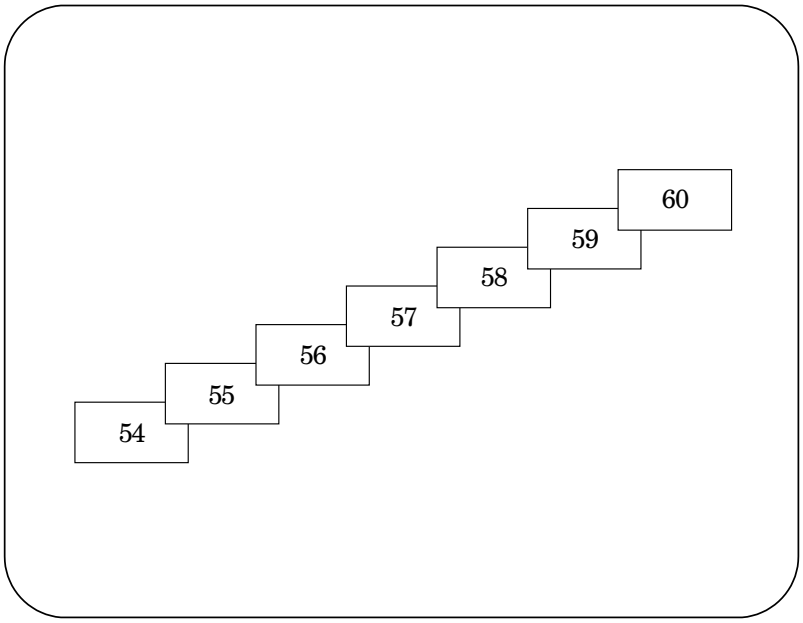
Description

This function sets “front-to-back” display priority for segment *sgna*. The segment priority in the segment state list for the segment is set to the value given by *prior*.

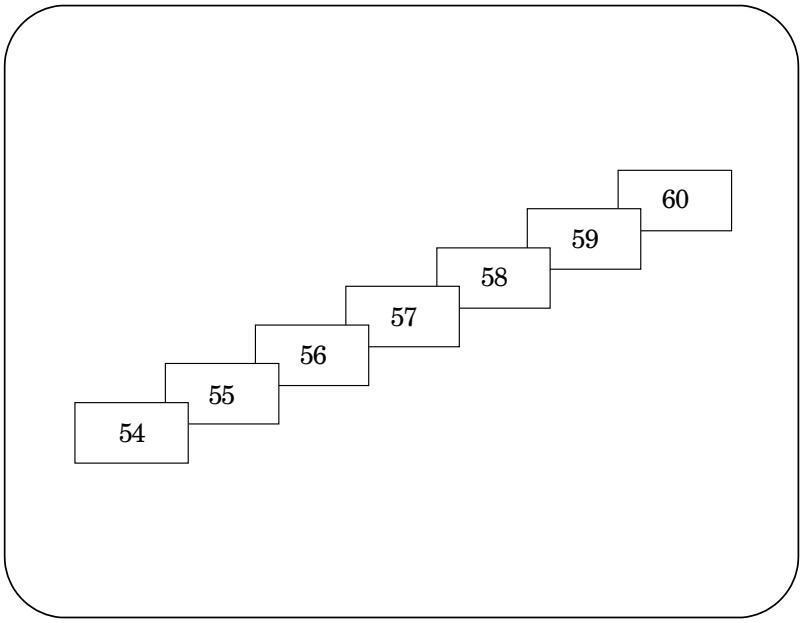
If displayed segments overlap, precedence is given to segments with higher priority. Segments with lower priority are displayed as being behind overlapping segments with a greater priority. If segments with the same priority overlap, the last segment drawn is placed on top.

Segment priority also affects pick input. When overlapping or intersecting segments are picked, the segment with the higher priority is delivered.

The following illustrations show the effect of the hierarchy of segment priorities. In the first display, the segments are drawn in the order in which they were created. Higher-numbered panels overlap lower-numbered ones.



The second display shows the effect of GSSGP when lower-numbered segments are assigned higher priorities. When segment priorities are set using this function, an implicit regeneration of the display may be required at workstations where the segment is displayed. If the implicit regeneration mode of a workstation is *suppressed*, you can call Redraw all segments on workstation (GRSGWK) to ensure a true representation of the graphics picture, including segment priority.



Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
120	Specified segment name is invalid
122	Specified segment does not exist
126	Segment priority is outside the range (0,1)

GSSGT

Purpose

GSSGT	(sgna, m)
APL code	1365
GKS RCP code	X'38004000' (939540480)

Function: To set segment transformation.

Segment function. Assigns a predefined transformation to a segment.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name.
m *(specified by user) (array of short floating-point numbers)*
The transformation matrix.

Operating states

WSOP, WSAC, SGOP

Related functions

Evaluate transformation matrix (GEVTM), Accumulate transformation matrix (GACTM), Create segment (GCRSG), Inquire segment attributes (GQSGA), Inquire dynamic modification of segment attributes (GQDSGA)

Description

This function applies segment transformation matrix *m* to segment *sgna*. The segment transformation matrix in the segment state list for the segment is set to the matrix *m*.

Segment transformations scale, translate, and rotate the coordinates of the primitives in the displayed segment. You define the transformation values by calling Evaluate transformation matrix (GEVTM) or Accumulate transformation matrix (GACTM).

Application of the transformation is not cumulative. It does not affect the contents of the segment. The transformation is always applied to the segment as originally

created. Applying the same segment transformation more than once to the segment gives identical results.

GSSGT transforms a segment stored on a workstation. It applies to all workstations where the segment is stored, even if they are not all active.

When a segment transformation is set by this function, an implicit regeneration of the display may be required at workstations where the segment is displayed. If the implicit regeneration mode of a workstation is *suppressed*, you can call Redraw all segments on workstation (GRSGWK) to ensure a true representation of the graphics picture.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
120	Specified segment name is invalid
122	Specified segment does not exist

GSSKM

Purpose

GSSKM	**(wkid, skdnr, mode, esw)**
APL code	1372
GKS RCP code	X'38004C00' (939543552)

Function: To set stroke mode.

Input function. Sets the operating mode and echo switch for a *stroke* device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
skdnr *(specified by user) (fullword integer)*
The stroke device number.
mode *(specified by user) (fullword integer)*
The operating mode. The possible values are:
0 (GREQU) Request
1 (GSAMPL) Sample
2 (GEVENT) Event
esw *(specified by user) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize stroke (GINSK), Request stroke (GRQSK)

Description

(See the section “GDDM-GKS restrictions” below.)

This function sets the operating mode and echo switch for the stroke input device *skdnr* at workstation *wkid*.

The echo switch determines whether or not an echo will be visible for the stroke device. The default values for all stroke devices is *echo*. The echo switch only controls whether the echo is on or off. Prompting is not affected.

GDDM-GKS Restrictions

Event and *sample* modes are not supported at GKS level 2b. If one of these modes is selected, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
143	EVENT and SAMPLE input mode are not available at this level of GKS
2000	Enumeration type out of range

GSSTM

Purpose

GSSTM	**(wkid, stdnr, mode, esw)**
APL code	1376
GKS RCP code	X'38005000' (939544576)

Function: To set string mode.

Input function. Sets the operating mode and echo switch for a *string* device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
stdnr *(specified by user) (fullword integer)*
The string device number.
mode *(specified by user) (fullword integer)*
The operating mode. The possible values are:
0 (GREQU) Request
1 (GSAMPL) Sample
2 (GEVENT) Event
esw *(specified by user) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No Echo
1 (GECHO) Echo

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize string (GINSTS), Request string (GRQSTS)

Description

(See the section “GDDM-GKS restrictions” below.)

This function sets the operating mode and echo switch for the string input device *stdnr* at workstation *wkid*.

The echo switch determines whether or not an echo will be visible for the string device. The default values for all string devices is *echo*. The echo switch only controls whether the echo is on or off. Prompting is not affected.

GDDM-GKS Restrictions

Event and *sample* modes are not supported at GKS level 2b. If one of these modes is selected, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
143	EVENT and SAMPLE input mode are not available at this level of GKS
2000	Enumeration type out of range

GSTXAL

Purpose

GSTXAL	(txalh, txalv)
APL code	1334
GKS RCP code	X'38002200' (939532800)

Function: To set text alignment.

Attribute function. Defines the horizontal and vertical text alignment components used for text justification. (See the section “GDDM-GKS restrictions” below.)

Parameters

txalh *(specified by user) (fullword integer)*
The horizontal alignment. The possible values are:
0 (GAHNOR) Normal
1 (GALEFT) Left
2 (GACENT) Center
3 (GARITE) Right

txalv *(specified by user) (fullword integer)*
The vertical alignment. The possible values are:
0 (GAVNOR) Normal
1 (GATOP) Top
2 (GACAP) Cap
3 (GAHALF) Half
4 (GABASE) Base
5 (GABOTT) Bottom

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Text (GTXS), Inquire text alignment (GQTXAL), Set character up vector (GSCHUP), Set text path (GSTXP), Inquire text extent (GQTXXS)

Description

(See the section “GDDM-GKS restrictions” below.)

This function defines the horizontal and vertical guides for text string justification. The horizontal and vertical components of the current text alignment in the GKS state list are set to the values given by *txalh* and *txalv*. These values are used when subsequent text output primitives are created.

When text output primitives are displayed, the text alignment determines how the character string is aligned with respect to the text position. The text alignment has two components: horizontal and vertical. Either component can be specified as

normal, which is equivalent to specifying one of the other values of the component, depending on the text path:

Right For horizontal alignment, *normal=left*;

for vertical alignment, *normal=base*

Left For horizontal alignment, *normal=right*;

for vertical alignment, *normal=base*

Up For horizontal alignment, *normal=center*;

for vertical alignment, *normal=base*

Down For horizontal alignment, *normal=center*;

for vertical alignment, *normal=top*

When the text attributes other than alignment have been applied to a character string, you can consider the characters in the string to define an (imaginary) text extent parallelogram.

If text path equals *left* or *right*, the left side of the parallelogram is the left side of the character body of the leftmost character. The right side of the parallelogram is the right side of the character body of the rightmost character. Consider the top and bottom of the parallelogram to be defined by the font *topline* and *bottomline* (for *top* or *bottom* alignment) or the font *capline* and *baseline* (for *cap*, *half*, or *base* alignment).

If text path equals *up* or *down*, the left and right sides of the parallelogram are defined by the width of the widest character in the font. The top of the parallelogram is the top (or cap) of the topmost character and the bottom is the bottom (or base) of the lowermost character.

GDDM-GKS Restrictions

In the fonts provided by GDDM-GKS, the capline is the same as the font topline and the baseline is the same as the font bottomline. This affects the vertical alignment as follows:

Cap is treated as *top* alignment.

Base is treated as *bottom* alignment.

Half the font *halfline* is midway between the font topline and bottomline.

The resulting values for the text alignment horizontal component have the following effects:

Left The left side of the text extent parallelogram passes through the text position.

Center The text position lies midway between the left and right sides of the text extent parallelogram.

Right The right side of the text extent parallelogram passes through the text position.

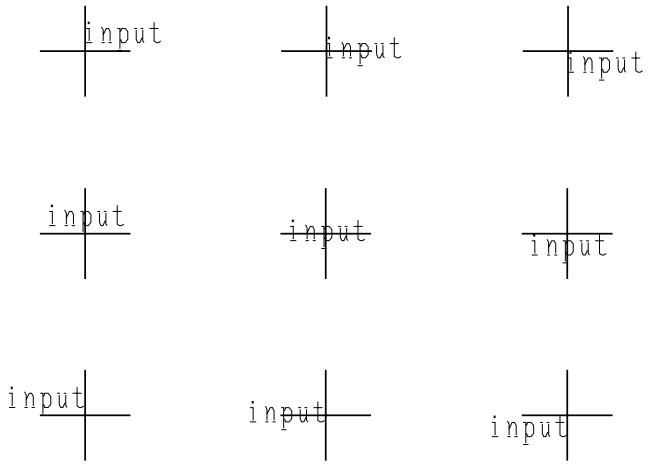
The resulting values for the text alignment vertical component have the following effects:

Top The top of the text extent parallelogram passes through the text position.

Half The text position lies midway between the top and bottom of the text extent parallelogram

Bottom The bottom of the text extent parallelogram passes through the text position.

This illustration shows the text alignment positions available in GDDM-GKS:



Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
2000	Enumeration type out of range

GSTXCI

Purpose

GSTXCI	(coli)
APL code	1330
GKS RCP code	X'38001E00' (939531776)

Function: To set text color index.

Attribute function. Sets the current *text* color index for text output primitives.

Parameters

coli *(specified by user) (fullword integer)*
The text color index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Text (GTXS), Inquire text color index (GQTXCI), Set color representation (GSCR), Set aspect source flags (GSASF)

Description

The current text color index in the GKS state list is set to the value given by *coli*. This value is used for the display of subsequent text output primitives when the text color index ASF is *individual*.

You define the color represented by a color index at a particular workstation by calling Set color representation (GSCR). The color index is a pointer into the workstation color table. If the color index *coli* has not been defined at a workstation, color index 1 is used on that workstation.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
92	Color index is less than zero

GSTXFP

Purpose

GSTXFP	**(font, prec)**
APL code	1326
GKS RCP code	X'38001B00' (939531008)

Function: To set text font and precision.

Attribute function. Sets the current text font and precision for *text* output primitives.

Parameters

font *(specified by user) (fullword integer)*
The text font. Identifies a particular text font. This can be either a negative value, indicating an implementation-dependent font, or 1, the default font.

prec *(specified by user) (fullword integer)*
The text precision. The possible values are:

0	(GSTRP)	String precision
1	(GCHARP)	Character precision
2	(GSTRKP)	Stroke precision

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Text (GTXS), Inquire text font and precision (GQTXFP), Set aspect source flags (GSASF)

Description

The current text font and precision in the GKS state list are set to the values given by *font* and *prec*. These values are used for subsequent text output primitives created when the current text font and precision aspect source flag (ASF) is *individual*.

The text precision type governs the closeness of text appearance to that defined by the text attributes, transformation, and clipping in effect when text primitives are displayed. There are three types of text precision:

String precision means that the text is positioned as an entire string. If the text primitive is clipped (because the text position lies outside the workstation window or, if clipping is on, outside the normalization transformation viewport), the entire string is discarded. If the primitive is not clipped, the entire string is displayed.

The attributes: character height, character width, and character expansion factor are evaluated as closely as possible, given the workstation capabilities. The attributes: character up vector, character base vector, text path, text alignment and character spacing are not used.

Character precision means that the text is treated on a character-by-character basis. If the text primitive is clipped, each character is treated separately; no individual character is clipped. The character height and character width aspects are evaluated as closely as possible, given the workstation capabilities. The character up vector and character base vector aspects are used to position the individual characters in the string but characters are not rotated. The other attributes are evaluated exactly.

Stroke precision means that GKS applies all text attributes to the character string being displayed.

If the specified text font and precision is not available at a workstation, text font 1 and precision *string* is used on that workstation. Note that GDDM-GKS may use a higher precision than the one requested. If a font is provided at *stroke* or *character* precision, it is implied that any lower precisions are available. For example, if a font is provided at *stroke* precision, you can set *character* or *string* precision, but the workstation may use *stroke* precision to display text.

The fonts specified by negative values of *font* are implementation-dependent and availability is workstation-dependent. The following table shows the values for *font* supported by GDDM-GKS.

Font number	Precisions provided	Description
1	All	Default font
-1	Character, stroke	Italic characters
	GDDM national language symbol sets:	

Font number	Precisions provided	Description
-2	Stroke	Brazilian
-3	Stroke	Danish
-4	Stroke	English
-5	Stroke	French
-6	Stroke	German
-7	Stroke	Italian
-8	Stroke	Japanese (Katakana)
-9	Stroke	Norwegian
-10	Stroke	Spanish
-11	Stroke	Swedish
	Proportionally-spaced symbol sets:	
-20	Stroke	Area Filled Roman Principal
-21	Stroke	Outline Roman Principal
-22	Stroke	Complex Italic Principal
-23	Stroke	Complex Roman Principal
-24	Stroke	Complex Script Principal
-25	Stroke	Duplex Roman Principal
-26	Stroke	Simplex Roman Principal
-27	Stroke	Triplex Italic Principal
-28	Stroke	Triplex Roman Principal
-29	Stroke	Gothic English Principal
-30	Stroke	Gothic German Principal
-31	Stroke	Gothic Italian Principal
-40	Stroke	Shadow
-41	Stroke	Thin Filled
-42	Stroke	Thin Outline
-43	Stroke	Thick Square Filled
-44	Stroke	Thick Square Outlined
-45	Stroke	Thick Round Filled
-46	Stroke	Thick Round Outlined
-47	Stroke	Moderne
	Double-byte character symbol sets:	
-65	Character, Stroke	DBCS Kanji symbols

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
75	Text font is equal to zero
2000	Enumeration type out of range

GSTXI

Purpose

GSTXI	(txi)
APL code	1325
GKS RCP code	X'38001A00' (939530752)

Function: To set text index.

Attribute function. Selects a bundle of *text* attributes to be used for text output primitives.

Parameters

txi *(specified by user) (fullword integer)*
The text index.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Set text representation (GSTXR), Inquire text index (GQTXI), Set aspect source flags (GSASF)

Description

The current text entry in the GKS state list is set to the value given by *txi*. This value is used when creating subsequent text primitives.

The appearance of text primitives, at a workstation, is determined by the text font and precision, character expansion factor, character spacing, and color. Using Set text representation (GSTXR), you can define a number of bundles of values for these four attributes at any workstation. The bundles are stored in the text bundle table of the workstation. The text index is a pointer into the text bundle table at each workstation. When a text primitive is displayed, attributes from the bundle specified by the text index are used according to the corresponding aspect source flag (ASF). You use Set aspect source flags (GSASF) to define whether or not text attributes are to be taken from bundles for the output workstations. If the ASF for an attribute is *bundled*, the bundled attribute is used. An aspect source flag exists for each of these text attributes:

Character expansion factor
Character spacing
Text color index
Text font and precision

If you do not call this function in your program before calling the Text (GTXS) function, or if the index you request is not defined or is not available on the workstation, text index 1 is used.

Principal errors

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

72 Text index is invalid

GSTXP

Purpose

GSTXP	(txp)
APL code	1333
GKS RCP code	X'38002100' (939532544)

Function: To set text path.

Attribute function. Sets the current text path for *text* output primitives.

Parameters

txp *(specified by user) (fullword integer)*
The direction in which characters are placed, relative to the previous character position. The possible values are:

0 (GRIGHT) Right
1 (GLEFT) Left
2 (GUP) Up
3 (GDOWN) Down

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Inquire text path (GQTXP)

Description

The “current text path” entry in the GKS state list is set to the value given by *txp*. This value is used when creating subsequent text output primitives. The text path is the direction of placement of characters relative to the preceding character on the workstation.

If this function is not used, GDDM-GKS uses the default value *right*.

Principal errors

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

GSTXR

Purpose

GSTXR	(wkid, txi, font, prec, chxp, chsp, coli)
APL code	1346
GKS RCP code	X'38002D00' (939535616)

Function: To set text representation.

Attribute function. Creates a bundle of text attribute values and associates them with a text index at a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.

txi *(specified by user) (fullword integer)*
The text index. This identifies a bundle of text attributes that are specified in the other parameters of this function.

font *(specified by user) (fullword integer)*
Text font. This value selects a particular font at the workstation. It can be either a negative value, indicating an implementation-dependent font, or 1, the default font. A particular text font can be available at some, but not necessarily all, precisions.

prec *(specified by user) (fullword integer)*
The precision of the font. The possible values are:
0 (GSTRP) String precision
1 (GCHARP) Character precision
2 (GSTRKP) Stroke precision.
The text precision value determines the fidelity with which the other text aspects are used.

chxp *(specified by user) (short floating point)*
The character expansion factor. This specifies the deviation of the width-to-height ratio of the characters from that of the original font.

chsp *(specified by user) (short floating point)*
The character spacing. This specifies how much additional space is to be inserted between two adjacent character bodies. Character spacing is specified as a fraction of the nominal character height.

coli *(specified by user) (fullword integer)*
The text color index.

Operating states

WSOP, WSAC, SGOP

Related functions

Text (GTXS), Inquire text representation (GQTXR), Set text index (GSTXI), Set text font and precision (GSTXFP), Inquire text facilities (GQTXF)

Description

This function creates a bundle of attribute values and associates them with text index *txi* at workstation *wkid*. The bundle is stored in the text bundle table of the workstation.

The text bundle table in the workstation state list has predefined entries taken from the workstation description table when the workstation is opened. Several bundles are predefined for every output and output/input workstation. Any table entry (including the predefined entries) may be redefined using this function.

When a text output primitive is displayed, the current text index refers to an entry in the text bundle table. If text is displayed with a text index that is not present in the text bundle table, text index 1 is used. Which of the attributes in the bundle table entry are used depends on the setting of the following text attribute aspect source flags:

Character expansion factor ASF
Character spacing ASF
Text color index ASF
Text font and precision ASF

If a text representation is set by this function, the changes may affect the displayed primitives by causing an implicit regeneration of the display, depending on the workstation implicit regeneration mode.

See the Set text font and precision (GSTXFP) function for details of the available text fonts and precisions.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
72	Text index is invalid
75	Text font is equal to zero
76	Requested text font is not supported for the specified precision on this workstation
77	Character expansion factor is less than or equal to zero
93	Color index is invalid
2000	Enumeration type out of range

GSVIS

Purpose

GSVIS	(sgna, vis)
APL code	1366
GKS RCP code	X'38004100' (939540736)

Function: To set visibility.

Segment function. Makes a segment visible or invisible on the display surface.

Parameters

sgna *(specified by user) (fullword integer)*
The segment name.
vis *(specified by user) (fullword integer)*
The visibility flag. The possible values are:
0 (GINVIS) Invisible
1 (GVISI) Visible

Operating states

WSOP, WSAC, SGOP

Related functions

Redraw all segments on workstation (GRSGWK), Update workstation (GUWK), Create segment (GCRSG), Inquire segment attributes (GQSGA), Inquire dynamic modification of segment attributes (GQDSGA)

Description

This function sets the visibility attribute for segment *sgna*. The value given by *vis* is set in the segment state list for the segment.

When segment visibility is set using this function, an implicit regeneration may be required at the workstation where the segment is displayed. To ensure an accurate representation, call Redraw all segments on workstation (GRSGWK) or Update workstation (GUWK).

The visibility of a segment is valid for all workstations associated with the segment, even if they are not active. The default visibility of a segment is *visible* when you create the segment. This function can also be applied to an open segment.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
120	Specified segment name is invalid
122	Specified segment does not exist
2000	Enumeration type out of range

GSVLM

Purpose

GSVLM	(wkid, vldnr, mode, esw)
APL code	1373
GKS RCP code	X'38004D00' (939543808)

Function: To set valuator mode.

Input function. Sets operating mode and echo switch for a *valuator* device at a workstation. (See the section “GDDM-GKS restrictions” below.)

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
vldnr *(specified by user) (fullword integer)*
The valuator device number.
mode *(specified by user) (fullword integer)*
The operating mode. The possible values are:
0 (GREQU) Request
1 (GSAMPL) Sample
2 (GEVENT) Event
esw *(specified by user) (fullword integer)*
The echo switch. The possible values are:
0 (GNECHO) No echo
1 (GECHO) Echo

Operating states

WSOP, WSAC, SGOP

Related functions

Initialize valuator (GINVL), Request valuator (GRQVL), Sample valuator (GSMVL)

Description

(See the section “GDDM-GKS restrictions” below.)

This function sets the operating mode and echo switch for the valuator input device *vldnr* at workstation *wkid*.

GDDM-GKS Restrictions

Event and *sample* modes are not supported at GKS level 2b. If one of these modes is selected, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
38	Specified workstation is neither of category INPUT nor of category OUTIN
140	Specified input device is not present on workstation
143	EVENT and SAMPLE input mode are not available at this level of GKS
2000	Enumeration type out of range

GSVP

Purpose

GSVP	(tnr, xmin, xmax, ymin, ymax)
APL code	1351
GKS RCP code	X'38003200' (939536896)

Function: To set viewport.

Transformation function. Defines the viewport NDC coordinates for a normalization transformation.

Parameters

tnr *(specified by user) (fullword integer)*
The transformation number.
xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The viewport's limits in normalized device coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Select normalization transformation (GSELNT), Set window (GSWN), Set viewport input priority (GSVPIP), Inquire normalization transformation (GQNT)

Description

This function defines the dimensions of a normalization transformation viewport. The viewport is the portion of NDC space to which you will map your world coordinate window. The viewport boundaries are limited by the range of NDC coordinates (0.0, 1.0) x (0.0, 1.0). Once selected by the Select normalization transformation (GSELNT) function, the viewport remains in effect until redefined.

GDDM-GKS supports up to 11 transformations. Transformation 0 is the unity transformation, and maps WC (0.0, 1.0) x (0.0, 1.0) to NDC (0.0, 1.0) x (0.0, 1.0). Transformations 1 to 10 default initially to be the same as transformation 0. You can redefine these by calling Set viewport (GSVP) and Set window (GSWN). You can select among the transformations by calling Select normalization transformation (GSELNT).

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
50	Transformation number is invalid
51	Rectangle definition is invalid
52	Viewport is not within the Normalized Device Coordinate unit square

GSVPIP

Purpose

GSVPIP	**(tnr, rtnr, relpri)**
APL code	1352
GKS RCP code	X'38003300' (939537152)

Function: To set viewport input priority.

Transformation function. Sets input priority for the viewport if more than one normalization transformation is defined.

Parameters

tnr *(specified by user) (fullword integer)*
The normalization transformation number.

rtnr *(specified by user) (fullword integer)*
The reference transformation number. This is a normalization transformation number that is used as the target by the *relpri* parameter.

relpri *(specified by user) (fullword integer)*
The priority relative to the reference transformation number. The possible values are:

0 (GHIGHR) Higher
1 (GLOWER) Lower

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Request locator (GRQLC), Request stroke (GRQSK), Inquire list element of normalization transformation numbers (GQENTN)

Description

During *stroke* and *locator* input, GKS transforms input points by inverse transformation, first by an inverse workstation transformation from DC to NDC, and then by an inverse normalization transformation from NDC to WC. If multiple normalization transformations are defined, and more than one transformation contains the same input point, the transformation with the highest priority number is used. This function sets the priority of transformation *tnr* to the next higher or lower priority relative to transformation *rtnr*. For example:

```
GSVPIP (5,2,1)
```

sets the transformation 5 viewport to one place below the transformation 2 viewport in relative priority.

If you call GSVPIP more than once, the most recent priority setting has precedence. For example, if you call

```
GSVPIP (5,2,1)
```

and then

```
GSVPIP (4,2,1)
```

transformation 4 viewport has a priority one less than transformation 2 viewport, and transformation viewport 5 has a priority one less than transformation 4 viewport.

You can use the function Inquire list element of normalization transformation numbers (GQENTN) to help decide the reference transformation number to be used.

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
50	Transformation number is invalid
2000	Enumeration type out of range

GSWKVP

Purpose

GSWKVP	**(wkid, xmin, xmax, ymin, ymax)**
APL code	1356
GKS RCP code	X'38003700' (939538176)

Function: To set workstation viewport.

Transformation function. Defines the device coordinate viewport for a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The workstation viewport limits in device coordinates.

Operating states

WSOP, WSAC, SGOP

Related functions

Set workstation window (GSWKWN), Inquire dynamic modification of workstation attributes (GQDWKA), Inquire workstation deferral and update states (GQWKDU), Inquire workstation transformation (GQWKT)

Description

This function defines the device coordinate viewport at the workstation *wkid*. This workstation viewport is the portion of the display surface into which the workstation window is mapped.

The “requested workstation viewport” entry in the workstation state list of workstation *wkid* is set to the values given in the function call.

If the “dynamic modification accepted for workstation transformation” entry in the workstation description table is set to *imm*, or if the “display surface empty” entry in the workstation state list is set to *empty*, the “current workstation viewport” entry in the workstation state list is set to the values given in the function call, and the “workstation transformation update state” entry is set to *notpending*. Otherwise, the “current workstation viewport” entry is not changed, and the “workstation transformation update state” entry is set to *pending*.

The workstation window will be mapped into the largest rectangle within the workstation viewport that will maintain a one-to-one aspect ratio between the workstation window and workstation viewport.

The extents of the workstation viewport are limited to the extents of the workstation display surface. Points extending beyond the workstation viewport limit are clipped.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
36	Specified workstation is Workstation Independent Segment Storage
51	Rectangle definition is invalid
54	Workstation viewport is not within the display space

GSWKWN

Purpose

GSWKWN	(wkid, xmix, xmax, ymin, ymax)
APL code	1355
GKS RCP code	X'38003600' (939537920)

Function: To set workstation window.

Transformation function. Defines the NDC coordinate workstation window for a workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
xmix *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The workstation window limits in normalized device coordinates.

Operating states

WSOP, WSAC, SGOP

Related functions

Set workstation viewport (GSWKVP), Inquire dynamic modification of workstation attributes (GQDWKA), Inquire workstation deferral and update states (GQWKDU), Inquire workstation transformation (GQWKT)

Description

This function defines the workstation window extents for the workstation *wkid*. The workstation window encloses the portion of NDC space that is mapped to the extents of the workstation viewport. The coordinates of the workstation window are limited to the range of NDC coordinates, (0,1) x (0,1).

The “requested workstation window” entry in the workstation state list of workstation *wkid* is set to the values given in the function call.

If the “dynamic modification accepted for workstation transformation” entry in the workstation description table is set to *imm*, or if the “display surface empty” entry in the workstation state list is set to *empty*, the “current workstation window” entry in the workstation state list is set to the values given in the function call, and the “workstation transformation update state” entry is set to *notpending*. Otherwise, the “current workstation window” entry is not changed, and the “workstation transformation update state” entry is set to *pending*.

If the aspect ratio of the workstation window to the workstation viewport is not one-to-one, the workstation window is mapped to the largest rectangle that maintains a one-to-one ratio without exceeding the extents specified in the Set workstation viewport (GSWKVP) function call.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
36	Specified workstation is Workstation Independent Segment Storage
51	Rectangle definition is invalid
53	Workstation window is not within the Normalized Device Coordinate unit square

GSWN

Purpose

GSWN	(tnr, xmin, xmax, ymin, ymax)
APL code	1350
GKS RCP code	X'38003100' (939536640)

Function: To set window.

Transformation function. Defines the world coordinate window for a normalization transformation.

Parameters

tnr *(specified by user) (fullword integer)*
The transformation number.
xmin *(specified by user) (short floating point)*
xmax *(specified by user) (short floating point)*
ymin *(specified by user) (short floating point)*
ymax *(specified by user) (short floating point)*
The window limits in world coordinates.

Operating states

GKOP, WSOP, WSAC, SGOP

Related functions

Select normalization transformation (GSELNT), Set viewport (GSVP), Set viewport input priority (GSVPIP), Inquire normalization transformation (GQNT), Set clipping indicator (GSCLIP)

Description

This function defines the world coordinate window for the normalization transformation *tnr*. The window limits given by the other parameters are stored in the entry for the transformation in the GKS state list.

When you use output primitives or geometric attributes, you specify positions and dimensions in world coordinates (WC). You can make up a picture from separate parts, each of which is defined in its own world coordinates. You control the relative positioning of the separate parts by defining normalization transformations, which map each part into a single coordinate system - normalized device coordinate space (NDC). The normalization transformation window and viewport determine the transformation used to transform WC into NDC. Window and viewport limits specify rectangles parallel to the coordinate axes in WC and NDC. The normalization transformation maps the entire window to the entire viewport, performing any scaling required, separately, for the x and y axes. Coordinates outside the window of the current normalization transformation are also transformed. If clipping is on, primitives are clipped to the viewport after any segment transformations have been applied.

GDDM-GKS supports up to 11 normalization transformations. Transformation 0 is the unity transformation, and maps WC (0.0, 1.0) x (0.0, 1.0) to NDC (0.0, 1.0) x (0.0, 1.0). Transformations 1 to 10 default initially to be the same as transformation 0. You can redefine these by calling Set viewport (GSVP) and Set window (GSWN). You can select among the transformations by calling Select normalization transformation (GSELNT).

Principal errors

8	GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
50	Transformation number is invalid
51	Rectangle definition is invalid

GTX

Purpose

GTX **(px,py,chars)**

Function: Output function. Draws a text string at a given position. Use this call only if your program is written in VS FORTRAN. Otherwise, use the function Text (GTXS) instead. (See the section “GDDM-GKS restrictions” below.)

Parameters

px *(specified by user) (short floating point)*
py *(specified by user) (short floating point)*
The position of the string in world coordinates.

chars *(specified by user) (character)*
The string of characters to be drawn.

Operating states

WSAC, SGOP

Related functions

Set text font and precision (GSTXFP), Set text color index (GSTXCI), Set character height (GSCHH), Set character up vector (GSCHUP), Set text alignment (GSTXAL), Inquire text color index (GQTXCI), Inquire text font and precision (GQTXFP), Inquire text alignment (GQTXAL), Inquire text extent (GQTXXS), Inquire text facilities (GQTXF), Set aspect source flags (GSASF), Set text representation (GSTXR), Set text index (GSTXI)

Description

(See the section “GDDM-GKS restrictions” below.)

This function displays a string of text at the position (*px, py*) in world coordinates.

The current text attributes are bound to the primitive.

If the character string contains control characters or undefined characters, the effect is workstation-dependent. Control characters included in the string can cause error 101 to be reported. Even if error 101 is reported, the character string is generated on the active workstations that do not report the error. If these are WISS or metafile output workstations, error 101 can be reported when the string is later displayed at a workstation.

GDDM-GKS restrictions

On the workstations supported by GDDM-GKS, only one control character, X'15', is valid; it is interpreted as a “new line” character. Character code X'FF' has a reserved meaning and must not be used.

The symbols displayed depend on the text font used when the string is displayed at a workstation. If the text font is -65 (DBSC Kanji or Hangeul symbols) or if the GDDM external defaults specify MIXSOSI=YES, the character strings can contain DBCS characters. If text font -65 has not been set but MIXSOSI=YES, DBCS character strings should be delimited by shift-out (SO) (X'0E') and shift-in (SI) (X'0F') control codes. Invalid use of DBCS code points may cause error 304 to be reported.

If *lstr* is less than zero, GDDM-GKS will report error 2001.

Principal errors

```
   5  GKS not in proper state: GKS shall be either in the state
      WSAC or in the state SGOP
 101  Invalid code in string
2001  Output parameter size insufficient
```

GTXS

Purpose

GTXS	(px, py, lstr, chars)
APL code	1492
GKS RCP code	X'38000E00' (939527680)

Function: To text.

Output function. Draws a text string at a given position. If your program is written in VS FORTRAN, use the function Text (VS FORTRAN only) (GTX) instead. (See the section “GDDM-GKS restrictions” below.)

Parameters

px *(specified by user) (short floating point)*
py *(specified by user) (short floating point)*
The position of the string in world coordinates.
lstr *(specified by user) (fullword integer)*
The length of the string in the *chars* parameter.
chars *(specified by user) (character)*
The string of characters to be drawn.

Operating states

WSAC, SGOP

Related functions

Set text font and precision (GSTXFP), Set text color index (GSTXCI), Set character height (GSCHH), Set character up vector (GSCHUP), Set text alignment (GSTXAL), Inquire text color index (GQTXCI), Inquire text font and precision (GQTXFP), Inquire text alignment (GQTXAL), Inquire text extent (GQTXXS), Inquire text facilities (GQTXF), Set aspect source flags (GSASF), Set text representation (GSTXR), Set text index (GSTXI)

Description

(See the section “GDDM-GKS restrictions” below.)

This function displays a string of text at the position (*px, py*) in world coordinates.

The current text attributes are bound to the primitive.

If the character string contains control characters or undefined characters, the effect is workstation-dependent. Control characters included in the string can cause error 101 to be reported. Even if error 101 is reported, the character string is generated on the active workstations that do not report the error. If these are WISS or metafile output workstations, error 101 can be reported when the string is later displayed at a workstation.

GDDM-GKS Restrictions

On the workstations supported by GDDM-GKS, only one control character, X'15', is valid; it is interpreted as a "new line" character. Character code X'FF' has a reserved meaning and must not be used.

The symbols displayed depend on the text font used when the string is displayed at a workstation. If the text font is -65 (DBSC Kanji or Hangeul symbols) or if the GDDM external defaults specify MIXSOSI=YES, the character strings can contain DBCS characters. If text font -65 has not been set but MIXSOSI=YES, DBCS character strings should be delimited by shift-out (SO) (X'0E') and shift-in (SI) (X'0F') control codes.

If *lstr* is less than zero, GDDM-GKS will report error 2001.

Principal errors

5	GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
101	Invalid code in string
2001	Output parameter size insufficient

GUREC

Purpose

GUREC **(ldr,datrec,iil,irl,isl,errind,il, ia,rl,ra,sl,lstr,str)**

Function: Utility function. Unpacks a data record. Use this call only if your program is written in FORTRAN IV or VS FORTRAN. Otherwise, use the function Unpack data record (GURECS) instead.

Parameters

ldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.

datrec *(specified by user) (array of 80-byte character tokens)*
The data record array.

iil *(specified by user) (fullword integer)*
The dimension of integer array.

irl *(specified by user) (fullword integer)*
The dimension of floating-point array.

isl *(specified by user) (fullword integer)*
The number of character-string entries in the string array.

errind *(returned by GDDM) (fullword integer)*
The error indicator.

il *(returned by GDDM) (fullword integer)*
The number of integer entries returned in *ia*.

ia *(returned by GDDM) (an array of fullword integers)*
The integer array.

rl *(returned by GDDM) (fullword integer)*
The number of floating-point entries returned in *ra*.

ra *(returned by GDDM) (array of short floating-point numbers)*
The floating-point array.

sl *(returned by GDDM) (fullword integer)*
The number of character-string entries returned in *str*.

lstr *(returned by GDDM) (fullword integer)*
The length of the string returned in the *str* parameter.

str *(returned by GDDM) (character)*
The string array. In a VS FORTRAN program, the strings can be of variable length. For FORTRAN IV, you must define the strings as CHARACTER*80.

Operating states

GKCL, GKOP, WSOP, WSAC, SGOP

Related functions

Inquire locator device state (GQLCS), Inquire stroke device state (GQSKS), Inquire default locator device data (GQDLC), Inquire default stroke device data (GQDSK), Pack data record (GPRECS)

Description

You use this function to unpack input function data records returned by Inquire *** device state and Inquire default *** device data groups of functions. It cannot be used to unpack data records returned by the Read item from GKSM (GRDITM) function.

Principal errors

```
2001  Output parameter size insufficient
2003  Invalid data record
```

GURECS

Purpose

GURECS	(ldr, datrec, iil, irl, isl, mstr, errind, il, ia, rl, ra, sl, lstr, str)
APL code	1514
GKS RCP code	X'38006C00' (939551744)

Function: To unpack data record.

Utility function. Unpacks a data record. If your program is written in FORTRAN IV or VS FORTRAN, use the function Unpack data record (FORTRAN only) (GUREC) instead.

Parameters

ldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*.
datrec *(specified by user) (array of 80-byte character tokens)*
The data record array.
iil *(specified by user) (fullword integer)*
The dimension of integer array.
irl *(specified by user) (fullword integer)*
The dimension of floating-point array.
isl *(specified by user) (fullword integer)*
The number of character-string entries in the string array.
mstr *(specified by user) (fullword integer)*
The maximum length of an entry in the string array.
errind *(returned by GDDM) (fullword integer)*
The error indicator.
il *(returned by GDDM) (fullword integer)*
The number of integer entries returned in *ia*.
ia *(returned by GDDM) (an array of fullword integers)*
The integer array.
rl *(returned by GDDM) (fullword integer)*
The number of floating-point entries returned in *ra*.
ra *(returned by GDDM) (array of short floating-point numbers)*
The floating-point array.
sl *(returned by GDDM) (fullword integer)*
The number of character-string entries returned in *str*.
lstr *(returned by GDDM) (an array of fullword integers)*
The lengths of the strings returned in *str*.
str *(returned by GDDM) (character)*
The string array.

Operating states

GKCL, GKOP, WSOP, WSAC, SGOP

Related functions

Inquire locator device state (GQLCS), Inquire stroke device state (GQSKS), Inquire default locator device data (GQDLC), Inquire default stroke device data (GQDSK), Pack data record (GPRECS)

Description

You use this function to unpack input function data records returned by Inquire *** device state and Inquire default *** device data groups of functions. It cannot be used to unpack data records returned by the Read item from GKSM (GRDITM) function.

Principal errors

2001 Output parameter size insufficient
2003 Invalid data record

GUWK

Purpose

GUWK	(wkid, regfl)
APL code	1311
GKS RCP code	X'38000800' (939526144)

Function: To update workstation.

Control function. Updates a workstation according to the value of a control flag.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
regfl *(specified by user) (fullword integer)*
The update regeneration flag. This flag controls the way in which the workstation is updated. The possible values are:
0 (GPOSTP) Postpone
1 (GPERFO) Perform

Operating states

WSOP, WSAC, SGOP

Related functions

Set deferral state (GSDS), Inquire workstation deferral and update states (GQWKDU), Inquire dynamic modification of workstation attributes (GQDWKA), Inquire dynamic modification of segment attributes (GQDSGA), Redraw all segments on workstation (GRSGWK).

Description

This function causes all deferred actions for the workstation specified by *wkid* to be executed. Any blocked data for the workstation is transmitted. If the update regeneration flag *regfl* is set to *perform*, and the "new frame action necessary at update" entry in the workstation state list is *yes*, this function redraws all segments at the workstation in addition to performing the deferred actions.

If *regfl* is set to *perform*, and a new frame at update is necessary, this function regenerates the display in the same way as Redraw all segments on workstation (GRSGWK). The following actions are performed:

1. The display surface is cleared if not empty.
2. If the "workstation transformation update state" entry in the workstation state list is *pending*, the pending transformation is set (that is, the current workstation window and viewport are set to the values for the requested workstation window and viewport). The "workstation transformation update state" entry is set to *notpending*.
3. All visible segments at the workstation are redisplayed.
4. The "new frame action necessary at update" entry in the workstation state list is set to *no*.

A new frame at update would be necessary if implicit regeneration is suppressed and any action requiring implicit regeneration has been performed. The functions requiring implicit regeneration are those that modify workstation or segment attributes.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
20	Specified workstation identifier is invalid
25	Specified workstation is not open
33	Specified workstation is of category MI
35	Specified workstation is of category INPUT
36	Specified workstation is Workstation Independent Segment Storage
2000	Enumeration type out of range

GWAIT

Purpose

GWAIT	**(tout, wkid, icl, idnr)**
APL code	1391
GKS RCP code	X'38005D00' (939547904)

Function: To await event.

Input function. Returns information concerning the oldest entry in the input queue. (See the section "GDDM-GKS restrictions" below.)

Parameters

tout *(specified by user) (short floating point)*
The time-out value in seconds.
wkid *(returned by GDDM) (fullword integer)*
The workstation identifier.
icl *(returned by GDDM) (fullword integer)*
The input class. The possible values are:

0	(GNCLAS)	None
1	(GLOCAT)	Locator
2	(GSTROK)	Stroke
3	(GVALUA)	Valuator
4	(GCHOIC)	Choice
5	(GPICK)	Pick
6	(GSTRIN)	String

If the time-out interval has elapsed, and the input queue is empty, the value *none* is returned.
idnr *(returned by GDDM) (fullword integer)*
The logical input device number.

Operating states

WSOP, WSAC, SGOP

Related functions

Set locator mode (GSLCM), Set stroke mode (GSSKM), Set valuator mode (GSVLM), Set choice mode (GSCHM), Set pick mode (GSPKM), Set string mode (GSSTM), Flush device events (GFLUSH)

Description

(See the section “GDDM-GKS restrictions” below.)

When an input device is set in *event* mode, the input data is stored in an input queue. This queue stores all events for all input devices that are in event mode. The Await event (GWAIT) function checks the content of the input queue and makes available the oldest event that took place.

GDDM-GKS Restrictions

GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.

Principal errors

7	GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
143	EVENT and SAMPLE input mode are not available at this level of GKS
147	Input queue has overflowed
151	Timeout is invalid

GWITM

Purpose

GWITM	(wkid, type, idrl, ldr, datrec)
APL code	1407
GKS RCP code	X'38006500' (939549952)

Function: To write item to GKSM.

Metafile function. Writes a user item to a GKS metafile (GKSM) workstation.

Parameters

wkid *(specified by user) (fullword integer)*
The workstation identifier.
type *(specified by user) (fullword integer)*
The item type. For user items, the item type must be greater than 100.
idrl *(specified by user) (fullword integer)*
The number of significant characters in the data record.
ldr *(specified by user) (fullword integer)*
The dimension of the data record array *datrec*. This is the number of 80-character records in the array.
datrec *(specified by user) (array of 80-byte character tokens)*
The data record array.

Operating states

WSAC, SGOP

Related functions

Get item type from GKSM (GGTITM), Read item from GKSM (GRDITM)

Description

Output is generated on a GKSM output workstation if it is active.

This function can be used only to pass non-graphical data to the metafile. Graphical data is passed automatically, once a *metafile output (MO)* workstation has been activated. User items can be read from an input metafile using Read item from GKSM (GRDITM) but cannot be interpreted.

Principal errors

5	GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
20	Specified workstation identifier is invalid
30	Specified workstation is not active
32	Specified workstation is not of category MO
160	Item type is not allowed for user items
161	Item length is invalid

Appendix A. GKS data structures

This appendix lists the contents of the GKS data structures as maintained by GDDM-GKS. The information for each entry includes:

1. The name of the entry
2. The coordinate system

   The following abbreviations are used:

   WC - world coordinate system
   NDC - normalized device coordinate system
   DC - device coordinate system.
3. The permitted values
4. The data type

   The following abbreviations are used:

   I - integer
   R - real
   S - string
   P - point, in WC, NDC or DC
   N - name, used for identification of the error file
   E - enumeration type
   D - data record.
5. The initial value

   The following abbreviations are also used:

   undef - Undefined value
   wdt - Initial value taken from workstation description table.

An occurrence of n merely indicates a variable integer value and does not necessarily relate to other occurrences of n.

© Copyright IBM Corp. 1984, 1987

Operating state

Entry name	Coordinate system	Permitted values	Data type	Initial value
Operating state value	—	GKCL GKOP WSOP WSAC SGOP	E	GKCL

GKS description table

Entry name	Coordinate system	Permitted values	Data type	Initial value
Level of GKS	—	0a 0b 0c 1a 1b 1c 2a 2b 2c	E	2b
Number of available workstation types	—	1 ... n	I	13
List of available workstation types	—	—	(n) N	1 through 13
Maximum number of simultaneously open workstations	—	1 ... n	I	5
Maximum number of simultaneously active workstations	—	1 ... n	I	5
Maximum number of workstations associated with a segment	—	1 ... n	I	5
Maximum normalization transformation number	—	1 ... n	I	10

GKS state list

Entry name	Coordinate system	Permitted values	Data type	Initial value
Set of open workstations	—	—	(n) N	Empty
Set of active workstations	—	—	(n) N	Empty
Current polyline index	—	1 ... n	I	1
Current linetype	—	-n ... -1, 1 ... n	I	1
Current linewidth scale factor	—	>= 0	R	1.0
Current polyline color index	—	0 ... n	I	1
Current linetype ASF[1]	—	Bundled Individual	E	Individual
Current linewidth scale factor ASF[1]	—	Bundled Individual	E	Individual
Current polyline color index ASF[1]	—	Bundled Individual	E	Individual
Current polymarker index	—	1 ... n	I	1
Current marker type	—	-n ... -1, 1 ... n	I	3
Current marker size scale factor	—	>= 0	R	1.0
Current polymarker color index	—	0 ... n	I	1
Current marker type ASF[1]	—	Bundled Individual	E	Individual
Current marker size scale factor ASF[1]	—	Bundled Individual	E	Individual
Current polymarker color index ASF[1]	—	Bundled Individual	E	Individual
Current text index	—	1 ... n	I	1
Current text font and precision	—	-n ... -1, 1 ... n String Character Stroke	I E	1, String
Current character expansion factor	—	>0	R	0.0
Current character spacing	—	—	R	0.0
Current text color index	—	0 ... n	I	1
Current text font and precision ASF[1]	—	Bundled Individual	E	Individual
Current character expansion factor ASF[1]	—	Bundled Individual	E	Individual
Current character spacing ASF[1]	—	Bundled Individual	E	Individual
Current text color index ASF[1]	—	Bundled Individual	E	Individual
Current character height	WC	>0	R	0.01

Entry name	Coordinate system	Permitted values	Data type	Initial value
Current character up vector	WC	—	(2) R	0,1
Current character width	WC	>0	R	0.01
Current character base vector	WC	—	(2) R	0,1
Current text path	—	Right Left Up Down	E	Right
Current text alignment (horizontal and vertical)	—		(2) E	
Horizontal		Normal Left Center Right		Normal
Vertical		Normal Top Cap Half Base Bottom		Normal
Current fill area index	—	1 ... n	I	1
Current fill area interior style	—	Hollow Solid Pattern Hatch	E	Hollow
Current fill area style index	—	-n ... -1, 1 ... n	I	1
Current fill area color index	—	0 ... n	I	1
Current fill area interior style ASF[1]	—	Bundled Individual	E	Individual
Current fill area style index ASF[1]	—	Bundled Individual	E	Individual
Current fill area color index ASF[1]	—	Bundled Individual	E	Individual
Current pattern width vector	WC	—	(2) R	1,0
Current pattern height vector	WC	—	(2) R	0,1
Current pattern reference point	WC	—	P	(0,0)
Current pick identifier	—	—	N	0
Current normalization transformation number	—	0 ... n	I	0

Entry name	Coordinate system	Permitted values	Data type	Initial value
Normalization transformation number[2]	—	0 ... n	I	Entry number
Window[2]	WC	—	(4) R	0,1,0,1
Viewport[2]	NDC	—	(4) R	0,1,0,1
Clipping indicator	—	Clip No clip	E	Clip
Clipping rectangle	NDC	—	(4) R	0,1,0,1
Name of open segment	—	—	N	undef
Set of segment names in use	—	—	(n) N	Empty
Set of segment state lists (one state list for every segment)	—	—	—	Empty

Notes:

1. In GDDM-GKS, all the initial aspect source flag (ASF) values are *individual*.
2. List of normalization transformations ordered by viewport input priority (initially in numerical order with 0 highest). Each entry contains these items.
3. GDDM-GKS does not maintain an input queue or current event report because the GKS *sample* and *event* input facilities are not available.

Workstation state list

One workstation state list exists for every open workstation. For metafile output workstations, the values marked “wdt” below are actually implementation-dependent because the workstation description table does not contain the corresponding entries.

Relevant for all workstation categories

Entry name	Coordinate system	Permitted values	Data type	Initial value
Workstation identifier	—	—	N	—
Connection identifier	—	—	N	—
Workstation type	—	—	N	—

Note: The above three entries are initialized by the Open workstation (GOPWK) function.

Relevant for output, output/input, WISS, and metafile output workstations

Entry name	Coordinate system	Permitted values	Data type	Initial value
Workstation state	—	Active Inactive	E	Inactive
Set of stored segments for this workstation	—	—	(n) N	Empty
Set of stored groups for this workstation	—	—	(n) N	Empty

Relevant for output, output/input, and metafile output workstations

Entry name	Coordinate system	Permitted values	Data type	Initial value
Deferral mode[1]	—	ASAP BNIG BNIL ASTI	E	wdt
Implicit regeneration mode	—	Suppressed Allowed	E	wdt
Display surface empty	—	Empty Not empty	E	Empty
New frame action necessary at update	—	No Yes	E	No
Number of polyline bundle table entries	—	5 ... n	I	wdt
Polyline index[2]	—	1 ... n	I	wdt
Linetype[2]	—	-n ... -1, 1 ... n	I	wdt
Linewidth scale factor[2]	—	>=0	R	wdt
Polyline color index[2]	—	0 ... n	I	wdt

Entry name	Coordinate system	Permitted values	Data type	Initial value
Number of polymarker bundle table entries	—	5 ... n	I	wdt
Polymarker index[3]	—	1 ... n	I	wdt
Marker type[3]	—	-n ... -1, 1 ... n	I	wdt
Marker size scale factor[3]	—	>=0	R	wdt
Polymarker color index[3]	—	0 ... n	I	wdt
Number of text bundle table entries	—	2 ... n	I	wdt
Text index[4]	—	1 ... n	I	wdt
Text font and precision[4]	—	-n ... -1, 1 ... n String Character Stroke	I E	wdt
Character expansion factor[4]	—	>0	R	wdt
Character spacing[4]	—	—	R	wdt
Text color index[4]	—	0 ... n	I	wdt
Number of fill area bundle table entries	—	5 ... n	I	wdt
Fill area index[5]	—	1 ... n	I	wdt
Fill area interior style[5]	—	Hollow Solid Pattern Hatch	E	wdt
Fill area style index[5]	—	-n ... -1, 1 ... n	I	wdt
Fill area color index[5]	—	0 ... n	I	wdt
Number of pattern table entries	—	0 ... n	I	wdt
Pattern index[6]	—	1 ... n	I	wdt
Pattern array dimensions[6]	—	1 ... n	(2) I	wdt
Pattern array[6]	—	0 ... n	(n) I	wdt
Number of color table entries	—	2 ... n	I	wdt
Color index[7]	—	0 ... n	I	wdt
Color (red, green, blue intensities)[7]	—	[0,1]	(3) R	wdt
Workstation transformation update state	—	Not pending Pending	E	Not pending
Requested workstation window	NDC	—	(4) R	0,1,0,1
Current workstation window	NDC	—	(4) R	0,1,0,1
Requested workstation viewport	DC	—	(4) R	[8]

Entry name	Coordinate system	Permitted values	Data type	Initial value
Current workstation viewport	DC	—	(4) R	8

Notes:

1. The following abbreviations are used:

   ASAP - As soon as possible
   BNIG - Before the next interaction globally
   BNIL - Before the next interaction locally
   ASTI - At some time.

2. For every entry in the table of defined polyline bundles
3. For every entry in the table of defined polymarker bundles
4. For every entry in the table of defined text bundles
5. For every entry in the table of defined fill area bundles
6. For every entry in the table of pattern representations
7. For every entry in the table of color representations
8. Maximum display surface from the workstation description table.

Relevant for input and output/input workstations

Entry name	Coordinate system	Permitted values	Data type	Initial value
Locator device number[1]	—	1 ... n	I	wdt
Operating mode[1]	—	Request	E	Request
Echo switch[1]	—	Echo No echo	E	Echo
Initial normalization transformation number[1]	—	0 ... n	I	0
Initial locator position[1]	WC	—	P	wdt
Prompt and echo type[1]	—	-n ... -1, 1 ... n	I	1
Echo area[1]	DC	—	(4) R	wdt
Locator data record[1]	—	—	D	wdt
Stroke device number[2]	—	1 ... n	I	wdt
Operating mode[2]	—	Request	E	Request
Echo switch[2]	—	Echo No echo	E	Echo
Initial normalization transformation number[2]	—	0 ... n	I	undef
Initial number of points[2]	—	0 ... n	I	0
Initial points in stroke[2]	WC	—	(n) P	Empty
Prompt and echo type[2]	—	-n ... -1, 1 ... n	I	1
Echo area[2]	DC	—	(4) R	wdt
Stroke data record[2]	—	—	D	wdt
Input buffer size[2,3]	—	1 ... n	I	wdt
Valuator device number[4]	—	1 ... n	I	wdt
Operating mode[4]	—	Request	E	Request
Echo switch[4]	—	Echo No echo	E	Echo
Initial value[4]	—	—	R	wdt
Prompt and echo type[4]	—	-n ... -1, 1 ... n	I	1
Echo area[4]	DC	—	(4) R	wdt
Valuator data record[4]	—	—	D	wdt
Low value[4,5]	—	—	R	wdt
High value[4,5]	—	—	R	wdt
Choice device number[6]	—	1 ... n	I	wdt
Operating mode[6]	—	Request	E	Request
Echo switch[6]	—	Echo No echo	E	Echo
Initial choice number[6]	—	0 ... n	I	undef
Prompt and echo type[6]	—	-n ... -1, 1 ... n	I	1
Echo area[6]	DC	—	(4) R	wdt
Choice data record[6]	—	—	D	wdt

Entry name	Coordinate system	Permitted values	Data type	Initial value
Pick device number[7]	—	—	I	wdt
Operating mode[7]	—	Request	E	Request
Echo switch[7]	—	Echo No echo	E	Echo
Initial status[7]	—	Ok No pick	E	No pick
Initial segment[7]	—	—	N	undef
Initial pick identifier[7]	—	—	N	undef
Prompt and echo type[7]	—	-n ... -1, 1 ... n	I	1
Echo area[7]	DC	—	(4) R	wdt
Pick data record[7]	—	—	D	wdt
String device number[8]	—	1 ... n	I	wdt
Operating mode[8]	—	Request	E	Request
Echo switch[8]	—	Echo No echo	E	Echo
Initial string[8]	—	—	S	Blank ' '
Prompt and echo type[8]	—	-n ... -1, 1 ... n	I	1
Echo area[8]	DC	—	(4) R	wdt
String data record[8]	—	—	D	wdt
Input buffer size[8,9]	—	1 ... n	I	wdt
Initial cursor position[8,9]	—	1 ... n	I	wdt

Notes:

1. For every locator logical input device
2. For every stroke logical input device
3. Must be contained in the stroke data record
4. For every valuator logical input device
5. Must be contained in the valuator data record
6. For every choice logical input device
7. For every pick logical input device
8. For every string logical input device
9. Must be contained in the string data record.

Workstation description tables

In this section, initial values are given only if they apply to all the physical devices supported by GDDM-GKS.

Relevant for all workstation categories

Entry name	Coordinate system	Permitted values	Data type	Initial value
Workstation type	—	—	N	—
Workstation category	—	Output Input Output/input WISS[1] MO[1] MI[1]	E	—

Notes:

1. The following abbreviations are used:

   WISS - Workstation independent segment storage
   MO - Metafile output
   MI - Metafile input.

Relevant for input, output, and output/input workstations

Entry name	Coordinate system	Permitted values	Data type	Initial value
Device coordinate units	—	Meters Other	E	Meters
Maximum display surface size[1]	DC Raster Units	>0 —	(2) R (2) I	— —

Notes:

1. The visible area of the display surface or available area on the tablet for input workstations.

Relevant for output and output/input workstations

Entry name	Coordinate system	Permitted values	Data type	Initial value
Raster or vector display	—	Vector Raster Other	E	—
Polyline bundle representation[1]	—	IRG IMM	E	IRG
Polymarker bundle representation[1]	—	IRG IMM	E	IRG
Text bundle representation[1]	—	IRG IMM	E	IRG
Fill area bundle representation[1]	—	IRG IMM	E	IRG
Pattern representation[1]	—	IRG IMM	E	IRG
Color representation[1]	—	IRG IMM	E	IRG

Entry name	Coordinate system	Permitted values	Data type	Initial value
Workstation transformation[1]	—	IRG IMM	E	IRG
Deferral mode[2]	—	ASAP BNIG BNIL ASTI	E	—
Implicit regeneration mode	—	Suppressed Allowed	E	Suppressed
Number of available linetypes	—	4 ... n	I	8
List of available linetypes	—	-n ... -1, 1 ... n	(n) I	-4 ... -1, 1 ... 4
Number of available linewidths[3]	—	0 ... n	I	2
Nominal linewidth	DC	>0	R	—
Minimum linewidth	DC	>0	R	—
Maximum linewidth	DC	>0	R	—
Number of predefined polyline indexes (bundles)	—	5 ... n	I	5
Linetype[4]	—	-n ... -1, 1 ... n	I	—
Linewidth scale factor[4]	—	—	R	—
Polyline color index (within the range of predefined color indexes)[4]	—	0 ... n	I	—
Number of available marker types	—	5 ... n	I	10
List of available marker types	—	-n ... -1, 1 ... n	(n) I	-5 ... -1, 1 ... 5
Number of available marker sizes[5]	—	0 ... n	I	0
Nominal marker size	DC	>0	R	—
Minimum marker size	DC	>0	R	—
Maximum marker size	DC	>0	R	—
Number of predefined polymarker indexes (bundles)	—	5 ... n	I	5
Marker type[6]	—	-n ... -1, 1 ... n	I	—
Marker size scale factor[6]	—	—	R	—
Polymarker color index (within the range of predefined color indexes)[6]	—	0 ... n	I	—
Number of text font and precision pairs	—	1 ... n	I	44

Entry name	Coordinate system	Permitted values	Data type	Initial value
List of text font and precision pairs	—	-n ... -1, 1 ... n String Character Stroke	(n) I (n) E	—
Number of available character expansion factors[7]	—	0 ... n	I	1
Minimum character expansion factor[8]	—	>0	R	1
Maximum character expansion factor[8]	—	>0	R	1
Number of available character heights[9]	—	0 ... n	I	1
Minimum character height[10]	DC	>0	R	—
Maximum character height[10]	DC	>0	R	—
Number of predefined text indexes (bundles)	—	2 ... n	I	5
Text font and precision[11]	—	-n ... -1, 1 ... n String Character Stroke	I E	—
Character expansion factor[11]	—	>0	R	—
Character spacing[11]	—	—	R	—
Text color index (within the range of predefined color indexes)[11]	—	0 ... n	I	—
Number of available fill area interior styles	—	1 ... 4	I	—
List of available fill area interior styles	—	Hollow Solid Pattern Hatch	(n) E	—
Number of available hatch styles	—	0 ... n	I	6
List of available hatch styles	—	-n ... -1, 1 ... n	(n) I	-6 ... -1
Number of predefined fill area indexes (bundles)	—	5 ... n	I	5
Fill area interior style[12]	—	Hollow Solid Pattern Hatch	E	—
Fill area style index[12,13]	—	-n ... -1, 1 ... n	I	—

Entry name	Coordinate system	Permitted values	Data type	Initial value
Fill area color index (within the range of predefined color indexes)[12]	—	0 ... n	I	—
Number of predefined pattern indexes (representations)	—	0 ... n	I	—
Pattern array dimensions[14]	—	1 ... n	(2) I	—
Pattern array[14]	—	0 ... n	(n) n x I	—
Number of available colors or intensities[15]	—	0,2 ... n	I	—
Color available	—	Color Monochrome	E	—
Number of predefined color indexes (representations)	—	2 ... n	I	—
Color (red, green, blue intensities)[16]	—	[0,1]	(3) R	—
Number of available generalized drawing primitives	—	0 ... n	I	0
Generalized drawing primitive identifier[17]	—	—	N	—
Number of sets of attributes used[17]	—	0 ... 4	I	—
List of sets of attributes used[17]	—	Polyline Polymarker Text Fill Area	(n) E	—
Maximum number of polyline bundle table entries	—	5 ... n	I	20
Maximum number of polymarker bundle table entries	—	5 ... n	I	20
Maximum number of text bundle table entries	—	2 ... n	I	10
Maximum number of fill area bundle table entries	—	5 ... n	I	20
Maximum number of pattern indexes	—	0 ... n	I	—
Maximum number of color indexes	—	2 ... n	I	20
Number of segment priorities supported[18]	—	0 ... n	I	0

Entry name	Coordinate system	Permitted values	Data type	Initial value
Segment transformation[1]	—	IRG IMM	E	IRG
Visibility (visible->invisible)[1]	—	IRG IMM	E	IRG
Visibility (invisible->visible)[1]	—	IRG IMM	E	IRG
Highlighting[1]	—	IRG IMM	E	IRG
Segment priority[1]	—	IRG IMM	E	IRG
Adding primitives to open segment overlapping segment of higher priority[1]	—	IRG IMM	E	IRG
Delete segment[1]	—	IRG IMM	E	IRG

Notes:

1. Dynamic modification accepted; the following abbreviations are used:

   IRG: Implicit regeneration necessary (may be deferred)
   IMM: Performed immediately.

2. The following abbreviations are used:

   ASAP - As soon as possible
   BNIG - Before next interaction globally
   BNIL - Before next interaction locally
   ASTI - At some time.

   The initial value is BNIG for OUTIN workstations, and ASTI for OUTPUT and MO workstations.

3. A value of 0 indicates that a continuous range of linewidths is supported.
4. For every entry in the predefined polyline bundles.
5. A value of 0 indicates that a continuous range of marker sizes is supported.
6. Contained in every entry of the table of predefined polymarker bundles.
7. Relevant only for string and character precision text. A value of 0 indicates that a continuous range of character expansion factors is supported.
8. If the available character expansion factors vary between fonts, these values are for font 1.
9. Relevant only for string and character precision text. A value of 0 indicates that a continuous range of character heights is supported.
10. If the available character heights vary between fonts, these values are for font 1.
11. These entries occur in every entry in the table of predefined text bundles.
12. These entries occur in every entry in the table of predefined fill area bundles.

13. For interior style pattern the index must be within the range of predefined pattern indexes. For interior style hatch the index must be within the range of available hatch styles.
14. For every entry in the table of predefined pattern representations.
15. A value of 0 indicates that a continuous range of colors is supported.
16. For every entry in the table of predefined color representations. The entries 0 and 1 are always included.
17. For every entry in the list of available generalized drawing primitives. In this implementation, the list is empty.
18. A value of 0 indicates that a continuous range of priorities is supported.

Relevant for input and output/input workstations

Entry name	Coordinate system	Permitted values	Data type	Initial value
Locator device number[1]	—	1 ... n	I	—
Default initial locator position[1]	WC	—	P	0.1, 0.1
Number of available prompt and echo types[1]	—	1 ... n	I	—
List of available prompt and echo types[1]	—	-n ... -1, 1 ... n	(n) I	—
Default echo area[1]	DC	—	(4) R	—
Default locator data record[1]	—	—	D	—
Stroke device number[2]	—	1 ... n	I	—
Maximum input buffer size[2]	—	64 ... n	I	—
Number of available prompt and echo types[2]	—	1 ... n	I	—
List of available prompt and echo types[2]	—	-n ... -1, 1 ... n	(n) I	—
Default echo area[2]	DC	—	(4) R	—
Default stroke data record[2,3]	—	—	D	—
Input buffer size[2,3]	—	1 ... n	I	64
Valuator device number[4]	—	1 ... n	I	—
Default initial value[4]	—	—	R	—
Number of available prompt and echo types[4]	—	1 ... n	I	1
List of available prompt and echo types[4]	—	-n ... -1, 1 ... n	(n) I	1
Default echo area[4]	DC	—	(4) R	—
Default valuator data record[4]	—	—	D	—
Low value[4,5]	—	—	R	-1E18
High value[4,5]	—	—	R	1E18
Choice device number[6]	—	1 ... n	I	—
Maximum number of choice alternatives[6]	—	1 ... n	I	—
Number of available prompt and echo types[6]	—	1 ... n	I	1
List of available prompt and echo types[6]	—	-n ... -1, 1 ... n	(n) I	1
Default echo area[6]	DC	—	(4) R	—
Default choice data record[6]	—	—	D	—
Pick device number[7]	—	—	I	—

Entry name	Coordinate system	Permitted values	Data type	Initial value
Number of available prompt and echo types[7]	—	1 ... n	I	1
List of available prompt and echo types[7]	—	-n ... -1, 1 ... n	(n) I	1
Default echo area[7]	DC	—	(4) R	—
Default pick data record[7]	—	—	D	—
String device number[8]	—	1 ... n	I	—
Maximum input buffer size[8]	—	72 ... n	I	72
Number of available prompt and echo types[8]	—	1 ... n	I	—
List of available prompt and echo types[8]	—	-n ... -1, 1 ... n	(n) I	—
Default echo area[8]	DC	—	(4) R	—
Default string data record[8]	—	—	D	—
Input buffer size[8]	—	1 ... n	I	72
Initial cursor position[8]	—	1 ... n	I	1

Notes:

1. These values apply to every locator class logical input device.
2. These values apply to every stroke class logical input device.
3. The default stroke data record must contain at a minimum the input buffer size.
4. These values apply to every valuator class logical input device.
5. The default valuator data record must contain these values at a minimum.
6. These values apply to every choice class logical input device.
7. These values apply to every pick class logical input device.
8. These values apply to every string class logical input device.

Segment state list

One segment state list exists for the open segment and for each stored segment.

Entry name	Coordinate system	Permitted values	Data type	Initial value
Segment name	—	—	N	—
Set of associated workstations	—	—	(n) N	[1]
Segment transformation matrix	—	—	(2 x 3) R	1,0,0 0,1,0[2]
Visibility	—	Visible Invisible	E	Visible
Highlighting	—	Normal Highlighted	E	Normal
Segment priority	—	[0,1]	R	0
Detectability	—	Undetectable Detectable	E	Undetectable

Notes:

1. Workstations that were active when the segment was created.
2. The elements $M_{1,3}$ and $M_{2,3}$ are in NDC coordinates. The other elements have no units.

GKS error state list

Entry name	Coordinate system	Permitted values	Data type	Initial value
Error state	—	Off On	E	Off
Error file	—	—	N	0

Appendix B. GDDM-GKS enumeration types

All the enumeration types of GKS are passed to or returned by GDDM-GKS as fullword integer parameters. The following table lists all of the enumeration types and values used for each of the possible states. The table contains the following information:

- The enumeration type.
- The possible states for the enumeration type.
- The names of the FORTRAN variables that correspond to the individual states. All of these variables are integers.
- The integer value assigned to each variable.

The following abbreviations are used in this table:

- BNIG - Before the next global interaction
- BNIL - Before the next local interaction
- GDP - Generalized drawing primitive
- NDC - Normalized device coordinates
- WISS - Workstation independent segment storage
- WSAC - Workstation active but no open segment
- WSOP - Workstation open but not active.

Enumeration type	States	Variable	Value
Aspect source	Bundled Individual	GBUNDL GINDIV	0 1
Attribute control flag	Current Specified	GCURNT GSPEC	0 1
Clear control flag	Conditionally Always	GCONDI GALWAY	0 1
Clipping indicator	No clip Clip	GNCLIP GCLIP	0 1
Color available	Monochrome Color	GMONOC GCOLOR	0 1
Coordinate switch	World coordinates NDC	GWC GNDC	0 1
Deferral mode	As soon as possible BNIG BNIL At some time	GASAP GBNIG GBNIL GASTI	0 1 2 3
Detectability	Undetectable Detectable	GUNDET GDETEC	0 1
Device coordinate units	Meters Other	GMETRE GOTHU	0 1
Display surface empty	Not Empty Empty	GNEMPT GEMPTY	0 1
Dynamic modification	Implicit regeneration Immediate	GIRG GIMM	0 1

© Copyright IBM Corp. 1984, 1987

Enumeration type	States	Variable	Value
Echo switch	No echo	GNECHO	0
	Echo	GECHO	1
Fill area interior style	Hollow	GHOLLO	0
	Solid	GSOLID	1
	Pattern	GPATTR	2
	Hatch	GHATCH	3
GDP attributes	Polyline bundle	GPLBND	0
	Polymarker bundle	GPMBND	1
	Text bundle	GTXBND	2
	Fill area bundle	GFABND	3
Highlighting	Normal	GNORML	0
	Highlighted	GHILIT	1
Initial choice prompt flag	Off	GPROFF	0
	On	GPRON	1
Input device status	None	GNONE	0
	Ok	GOK	1
	No pick	GNPICK	2
	No choice	GNCHOI	2
Input class	None	GNCLAS	0
	Locator	GLOCAT	1
	Stroke	GSTROK	2
	Valuator	GVALUA	3
	Choice	GCHOIC	4
	Pick	GPICK	5
	String	GSTRIN	6
Implicit regeneration mode	Suppressed	GSUPPD	0
	Allowed	GALLOW	1
Level of GKS	0a	GL0A	0
	0b	GL0B	1
	0c	GL0C	2
	1a	GL1A	3
	1b	GL1B	4
	1c	GL1C	5
	2a	GL2A	6
	2b	GL2B	7
	2c	GL2C	8
Line type	Solid	GLSOLI	1
	Dash	GLDASH	2
	Dot	GLDOT	3
	Dash-dot	GLDASD	4
Marker type	Period (.)	GPOINT	1
	Plus (+)	GPLUS	2
	Asterisk (*)	GAST	3
	Circle (o)	GOMARK	4
	Cross (x)	GXMARK	5
New frame action necessary	No	GNO	0
	Yes	GYES	1
Operating mode	Request	GREQU	0
	Sample	GSAMPL	1
	Event	GEVENT	2

Enumeration type	States	Variable	Value
Operating state value	GKS closed	GGKCL	0
	GKS open	GGKOP	1
	WSOP	GWSOP	2
	WSAC	GWSAC	3
	Segment open	GSGOP	4
Polyline/fill area control flag	Polyline	GPLINE	0
	Fill area	GFILLA	1
Presence of invalid values	Absent	GABSNT	0
	Present	GPRSNT	1
Regeneration flag	Postpone	GPOSTP	0
	Perform	GPERFO	1
Relative input priority	Higher	GHIGHR	0
	Lower	GLOWER	1
Simultaneous events flag	No more	GNMORE	0
	More	GMORE	1
Text alignment horizontal	Normal	GAHNOR	0
	Left	GALEFT	1
	Center	GACENT	2
	Right	GARITE	3
Text alignment vertical	Normal	GAVNOR	0
	Top	GATOP	1
	Cap	GACAP	2
	Half	GAHALF	3
	Base	GABASE	4
	Bottom	GABOTT	5
Text path	Right	GRIGHT	0
	Left	GLEFT	1
	Up	GUP	2
	Down	GDOWN	3
Text precision	String	GSTRP	0
	Character	GCHARP	1
	Stroke	GSTRKP	2
Type of returned value	Set	GSET	0
	Realized	GREALI	1
Update state	Pending	GNPEND	0
	Not pending	GPEND	1
Vector/raster other type	Vector	GVECTR	0
	Raster	GRASTR	1
	Other	GOTHWK	2
Visibility	Invisible	GINVIS	0
	Visible	GVISI	1
Workstation category	Output	GOUTPT	0
	Input	GINPUT	1
	Input/output	GOUTIN	2
	WISS	GWISS	3
	Metafile output	GMO	4
	Metafile input	GMI	5
Workstation state	Inactive	GINACT	0
	Active	GACTIV	1

Appendix C. Metafile structure

This appendix describes the items that can occur in a metafile created by a GDDM-GKS metafile output (MO) workstation.

A metafile consists of a sequence of items. Each item comprises a type, a data record length and a data record. The type is an identification number that indicates either that the item contains information that can be interpreted by GKS (and the function to be performed) or that it contains information that was written by an application program (using Write item to GKSM (GWITM)). The following functions make use of the item type numbers:

- Get item type from GKSM (GGTITM)

  Returns the type and data record length of a metafile item.

- Interpret item (GIITM)

  Performs the actions indicated by the type parameter using the contents of the item data record.

- Write item to GKSM (GWITM)

  An item with a type specified by the application (must be greater than 100) is written to an output metafile.

Metafile item types

Items are created in a metafile when the application invokes GKS functions that apply to an open or active MO workstation. The tables in the following sections list the GKS functions that can cause metafile items to be created.

Some functions create metafile items at an MO workstation only if the workstation identifier parameter refers to that workstation. These functions are indicated by an asterisk (⋆) in the following tables.

© Copyright IBM Corp. 1984, 1987

Control functions

Function	Item types created
Open workstation (GOPWK) ⋆	- (file header)
Close workstation (GCLWK) ⋆	0 (end item)
Activate workstation (GACWK) ⋆	61, 21-43
Deactivate workstation (GDAWK) ⋆	- (disable output)
Clear workstation (GCLRWK) ⋆	1
Redraw all segments on workstation (GRSGWK) ⋆	2
Update workstation (GUWK) ⋆	3
Set deferral state (GSDS) ⋆	4
Message (GMSGS) ⋆	5

Output primitives

Function	Item types created
Polyline (GPL)	11
Polymarker (GPM)	12
Text (GTXS)	13
Fill area (GFA)	14
Cell array (GCA)	15

Output attributes

Function	Item types created
Set polyline index (GSPLI)	21
Set linetype (GSLN)	22
Set linewidth scale factor (GSLWSC)	23
Set polyline color index (GSPLCI)	24
Set polymarker index (GSPMI)	25
Set marker type (GSMK)	26
Set marker size scale factor (GSMKSC)	27
Set polymarker color index (GSPMCI)	28
Set text index (GSTXI)	29
Set text font and precision (GSTXFP)	30
Set character expansion factor (GSCHXP)	31
Set character spacing (GSCHSP)	32
Set text color index (GSTXCI)	33
Set character height (GSCHH)	34
Set character up vector (GSCHUP)	34
Set text path (GSTXP)	35
Set text alignment (GSTXAL)	36
Set fill area index (GSFAI)	37
Set fill area interior style (GSFAIS)	38
Set fill area style index (GSFASI)	39
Set fill area color index (GSFACI)	40

Function	Item types created
Set pattern size (GSPA)	41
Set pattern reference point (GSPARF)	42
Set aspect source flags (GSASF)	43
Set pick identifier (GSPKID)	44

Workstation attributes

Function	Item types created
Set polyline representation (GSPLR) ⋆	51
Set polymarker representation (GSPMR) ⋆	52
Set text representation (GSTXR) ⋆	53
Set fill area representation (GSFAR) ⋆	54
Set pattern representation (GSPAR) ⋆	55
Set color representation (GSCR) ⋆	56

Transformation functions

Function	Item types created
Set window (GSWN)	34, 41, 42
Set viewport (GSVP)	61, 34, 41, 42
Select normalization transformation (GSELNT)	61, 34, 41, 42
Set clipping indicator (GSCLIP)	61
Set workstation window (GSWKWN) ⋆	71
Set workstation viewport (GSWKVP) ⋆	72

Segment functions

Function	Item types created
Create segment (GCRSG)	44, 81
Close segment (GCLSG)	82
Rename segment (GRENSG)	83
Delete segment (GDSG)	84
Delete segment from workstation (GDSGWK) ⋆	84
Associate segment with workstation (GASGWK) ⋆	81, (91-95), (21-44), (11-16), 61, 82
Copy segment to workstation (GCSGWK) ⋆	(21-44), (11-16), 61
Insert segment (GINSG)	(21-44), (11-16), 61

Segment attributes

Function	Item types created
Set segment transformation (GSSGT)	91
Set visibility (GSVIS)	92
Set highlighting (GSHLIT)	93
Set segment priority (GSSGP)	94
Set detectability (GSDTEC)	95

Metafile functions

Function	Item types created
Write item to GKSM (GWITM) ⋆	>100

Metafile item structure

This section describes the logical structure of each of the metafile items. Each item comprises:

- Type – a fullword integer value
- Data record length – a fullword integer value denoted below by L.
- Data record – the remainder of the item. The format of each data record depends on the item type, as shown below.

The type and data record length are returned to the application when the function Get item type from GKSM (GGTITM) is invoked. When the function Read item from GKSM (GRDITM) is invoked, the next metafile item data record is returned. Metafile items can be interpreted by invoking the function Interpret item (GIITM).

Fields in the item data records can be of the following types:

Fullword integer denoted by i.

Short floating-point denoted by (r). A point represented by a pair of real numbers (2r) is denoted by (p).

Character string denoted by (c).

Control items

End item

0	L

This is the last item of every metafile. This item is used to set the error condition *No item is left in GKS Metafile input*.

Clear workstation

1	L	C

Requests the Clear workstation (GCLRWK) function for all active workstations.

- C(i) - Clearing control flag
  - 0 = Conditional
  - 1 = Always

Redraw all segments on workstation

2	L

Requests the Redraw all segments on workstation (GRSGWK) function for all active workstations.

Update workstation

3	L	R

Requests the Update workstation (GUWK) function for all active workstations.

- R(i) - Update regeneration flag
  - 0 = Perform
  - 1 = Postpone

Deferral state

4	L	D	R

Requests the Set deferral state (GSDS) function for all active workstations.

- D(i) - Deferral mode
  - 0 = As soon as possible (ASAP)
  - 1 = Before next interaction globally (BNIG)
  - 2 = Before next interaction locally (BNIL)
  - 3 = At some time (ASTI)
- R(i) - Implicit regeneration mode
  - 0 = Allowed
  - 1 = Suppressed

Message

5	L	N	T

Requests the Message (GMSGS) function for all active workstations.

- N(i) - The number of characters in the string.
- T(Nc) - The string with N characters.

Items for output primitives

Polyline

11	L	N	P

- N(i) - The number of points in the polyline.
- P(Np) - The list of points.

Polymarker

12	L	N	P

- N(i) - The number of points.
- P(Np) - The list of points.

Text

13	L	P	N	T

- P(p) - The starting point of the character string.
- N(i) - The number of characters in string T.
- T(Nc) - The string of N characters.

Fill area

14	L	N	P

- N(i) - The number of points.
- P(Np) - The list of points.

Cell array

15	L	P	Q	R	N	M	CT

- P(p), Q(p), R(p) - The coordinates of the corner points of the pixel array (P and Q are the images of the points P and Q specified in the function Cell array (GCA), and R is the point associated with the (DX,1) cell).
- N(i) - The number of columns in the array.
- M(i) - The number of rows in the array.
- CT(MNi) - The array of color indexes stored row by row.

Items for output primitive attributes

Polyline index

21	L	I

- I(i) - The polyline index.

Linetype

22	L	LT

- LT(i) - The line type.

Linewidth scale factor

23	L	LW

- LW(r) - The line-width scale factor.

Polyline color index

24	L	CI

- CI(i) - The polyline color index.

Polymarker index

25	L	I

- I(i) - The polymarker index.

Marker type

26	L	MT

- MT(i) - The marker type.

Marker size scale factor

27	L	MS

- MS(r) - The marker size scale factor.

Polymarker color index

28	L	CI

- CI(i) - The polymarker color index.

Text index

29	L	I

- I(i) - The text index.

Text font and precision

30	L	F	P

- F(i) - The text font.
- P(i) - The text precision.
  0 = String
  1 = Character
  2 = Stroke

Character expansion factor

31	L	CEF

- CEF(r) - The character expansion factor.

Character spacing

32	L	CS

- CS(r) - The character spacing.

Text color index

33	L	CI

- CI(i) - The text color index.

Character vectors

34	L	CH	CW

- CH(2r) - The character height vector.
- CW(2r) - The character width vector.

Text path

35	L	P

- P(i) - The text path.
  - 0 = Right
  - 1 = Left
  - 2 = Up
  - 3 = Down

Text alignment

36	L	H	V

- H(i) - The horizontal character alignment.
  - 0 = Normal
  - 1 = Left
  - 2 = Center
  - 3 = Right
- V(i) - The vertical character alignment.
  - 0 = Normal
  - 1 = Top
  - 2 = Cap
  - 3 = Half
  - 4 = Base
  - 5 = Bottom

Fill area index

37	L	I

- I(i) - The fill area index.

Fill area interior style

38	L	S

- S(i) - The fill area interior style.
  - 0 = Hollow
  - 1 = Solid
  - 2 = Pattern
  - 3 = Hatch

Fill area style index

39	L	SI

- SI(i) - The fill area style index.

Fill area color index

40	L	CI

- CI(i) - The fill area color index.

Pattern vectors

41	L	PW	PH

- PW(2r) - The pattern width vector.
- PH(2r) - The pattern height vector.

Pattern reference point

42	L	P

- P(p) - The reference point.

Aspect source flags

43	L	F

- F(13i) - The aspect source flags.
  0 = Bundled
  1 = Individual

Pick identifier

44	L	P

- P(i) - The pick identifier.

Items for workstation attributes

Polyline representation

51	L	I	LT	LW	CI

- I(i) - The polyline index.
- LT(i) - The line type.
- LW(r) - The line-width scale factor.
- CI(i) - The polyline color index.

Polymarker representation

52	L	I	MT	MS	CI

- I(i) - The polymarker index.
- MT(i) - The marker type.
- MS(r) - The marker size scale factor.
- CI(i) - The polymarker color index.

Text representation

53	L	I	F	P	CEF	CS	CI

- I(i) - The text index.
- F(i) - The text font.
- P(i) - The text precision.
  - 0 = String
  - 1 = Character
  - 2 = Stroke
- CEF(r) - The character expansion factor.
- CS(r) - The character spacing.
- CI(i) - The text color index.

Fill area representation

54	L	I	S	SI	CI

- I(i) - The fill area index.
- S(i) - The fill area interior style.
  - 0 = Hollow
  - 1 = Solid
  - 2 = Pattern
  - 3 = Hatch
- SI(i) - The fill area style index.
- CI(i) - The fill area color index.

Pattern representation

55	L	I	N	M	CT

- I(i) - The pattern index.
- N(i) - The number of columns in the array.
- M(i) - The number of rows in the array.
- CT(MNi) - The table of color indexes stored row by row.

Color representation

56	L	CI	RGB

- CI(i) - The color index.
- RGB(3r) - The red, green, and blue intensities.

Items for transformations

Clipping rectangle

61	L	C

- C(4r) - The limits of the clipping rectangle (XMIN, XMAX, YMIN, YMAX).

Workstation window

71	L	W

- W(4r) - The limits of the workstation window (XMIN, XMAX, YMIN, YMAX).

Workstation viewport

72	L	V

- V(4r) - The limits of the workstation viewport (XMIN, XMAX, YMIN, YMAX).

Items for segment manipulation

Create segment

81	L	S

- S(i) - The segment name.

Close segment

82	L

This indicates the end of a segment.

Rename segment

83	L	SO	SN

- SO(i) - The old segment name.
- SN(i) - The new segment name.

Delete segment

84	L	S

- S(i) - The segment name.

Items for segment attributes

Segment transformation

91	L	S	M

- S(i) - The segment name.
- M(6r) - The transformation matrix:
  M_{11}, M_{12}, M_{13}
  M_{21}, M_{22}, M_{23}

Set visibility

92	L	S	V

- S(i) - The segment name.
- V(i) - The visibility setting:
  0 = Visible
  1 = Invisible

Set highlighting

93	L	S	H

- S(i) - The segment name.
- H(i) - The highlighting setting:
  0 = Normal
  1 = Highlighted

Set segment priority

94	L	S	P

- S(i) - The segment name.
- P(r) - The segment priority.

Set detectability

95	L	S	D

- S(i) - The segment name.
- D(i) - The detectability setting:
  0 = Undetectable

1 = Detectable

User items

User item

xxx	L	D

The number xxx must be greater than 100.

- D - The user data, containing L bytes.

Appendix D. ROOM program source code

This section contains the complete FORTRAN source code for the ROOM program that was discussed in Chapter 2, “Using GKS.”

This program is written so that it can be compiled by the VS FORTRAN and FORTRAN IV compilers. It therefore does not use some techniques that would usually be found in programs written only for the VS FORTRAN compiler.

```
C*********************************************************************** ADM00010
C**                                                                   ** ADM00020
C**                5666-802                                           ** ADM00030
C**                (C) COPYRIGHT IBM CORP. 1987                       ** ADM00040
C**                LICENSED MATERIALS - PROPERTY OF IBM               ** ADM00050
C**                                                                   ** ADM00060
C**                    ADMJROOM                                       ** ADM00070
C**                                                                   ** ADM00080
C**  A SAMPLE FORTRAN PROGRAM TO DO A SIMPLE ROOM LAYOUT WITH         ** ADM00090
C**  TWO FURNITURE SELECTIONS FOR PLACEMENT WITHIN A ROOM.            ** ADM00100
C**                                                                   ** ADM00110
C**                                                                   ** ADM00120
C*********************************************************************** ADM00130
C                                                                       ADM00140
C*** Control variables                                                  ADM00150
C                                                                       ADM00160
C - Segment number, message flag                                        ADM00170
      INTEGER      SEGNUM,MSGFLG                                        ADM00180
C - Pick flag, position flag                                            ADM00190
      INTEGER      PCKFLG,POSFLG                                        ADM00200
C                                                                       ADM00210
C*** Program data                                                       ADM00220
C                                                                       ADM00230
C - Room Outline                                                        ADM00240
      REAL         XRRY1(7),YRRY1(7)                                    ADM00250
C - Initial desk outline                                                ADM00260
      REAL         XDRY(5),YDRY(5)                                      ADM00270
C - Desk                                                                ADM00280
      REAL         XDSK(5),YDSK(5)                                      ADM00290
C - Initial chair outline                                               ADM00300
      REAL         XCHRY1(9),YCHRY1(9),XCHRY2(2)                        ADM00310
      REAL         YCHRY2(2),XCHRY3(5),YCHRY3(5)                        ADM00320
C - Chair                                                               ADM00330
      REAL         XCHR1(9),YCHR1(9),XCHR2(2),YCHR2(2)                  ADM00340
      REAL         XCHR3(5),YCHR3(5)                                    ADM00350
C - Furniture width and height                                          ADM00360
      REAL         FURWDT,FURHIG                                        ADM00370
C                                                                       ADM00380
C*** GKS input parameters                                               ADM00390
C                                                                       ADM00400
C - Workstation identifier, workstation type                            ADM00410
      INTEGER      WKID1,WKTYPE                                         ADM00420
C - Normalization transformation viewport size                          ADM00430
      REAL         XNDC,YNDC                                            ADM00440
C - Echo area for valuator input (minimum x/y values)                   ADM00450
      REAL         MINXDC,MINYDC                                        ADM00460
C - X, Y shift for transformation matrix                                ADM00470
```

© Copyright IBM Corp. 1984, 1987

```
      REAL         SHFTX,SHFTY                                          ADM00480
C - Rotation angle                                                      ADM00490
      REAL ANG                                                          ADM00500
C - Data record                                                         ADM00510
      LOGICAL*1    DATREC(80)                                           ADM00520
C                                                                       ADM00530
C*** GKS output parameters                                              ADM00540
C                                                                       ADM00550
C - Error indicator, device coordinate units                            ADM00560
      INTEGER      ERRIND,DCUNIT                                        ADM00570
C - Maximum display surface size in DC and Raster units                 ADM00580
      REAL         XDC,YDC                                              ADM00590
      INTEGER      XRAS,YRAS                                            ADM00600
C - Number of segment names, nth set member of set of stored segments   ADM00610
      INTEGER      NUMSEG,SEGN                                          ADM00620
C - Input device status, choice number                                  ADM00630
      INTEGER      STAT,CHCNUM                                          ADM00640
C - Picked segment number, pick identifier                              ADM00650
      INTEGER      SEGNM,PICKID                                         ADM00660
C - Locator returned transformation number and position                 ADM00670
      INTEGER      TRANUM                                               ADM00680
      REAL         XPOP,YPOP                                            ADM00690
C - Transformation matrices                                             ADM00700
      REAL         MTX(6),MTX2(6)                                       ADM00710
C                                                                       ADM00720
C*** Work variables                                                     ADM00730
C                                                                       ADM00740
      REAL         RTEMP                                                ADM00750
      INTEGER      J                                                    ADM00760
C                                                                       ADM00770
C*** Initial values                                                     ADM00780
C                                                                       ADM00790
C - Workstation identifier, workstation type                            ADM00800
      DATA         WKTYPE  /1/                                          ADM00810
      DATA         WKID1   /1/                                          ADM00820
C - Room Outline                                                        ADM00830
      DATA XRRY1   /00.0,00.0,60.0,60.0,00.0,00.0,10.5/                 ADM00840
      DATA YRRY1   /55.0,99.8,99.8,34.0,34.0,36.0,49.4/                 ADM00850
C - Initial desk outline                                                ADM00860
      DATA XDRY    /75.0,100.0,100.0,75.0,75.0/                         ADM00870
      DATA YDRY    /87.0,87.0,100.0,100.0,87.0/                         ADM00880
C - Desk                                                                ADM00890
      DATA XDSK    /00.0,25.0,25.0,00.0,00.0/                           ADM00900
      DATA YDSK    /00.0,00.0,13.0,13.0,00.0/                           ADM00910
C - Initial chair outline                                               ADM00920
      DATA XCHRY1  /93.0,93.0,94.0,94.0,99.0,99.0,100.0,                ADM00930
     $             100.0,93.0/                                          ADM00940
      DATA YCHRY1  /73.0,80.0,80.0,73.0,73.0,80.0,80.0,                 ADM00950
     $              73.0,73.0/                                          ADM00960
      DATA XCHRY2  /94.0,99.0/                                          ADM00970
      DATA YCHRY2  /79.0,79.0/                                          ADM00980
      DATA XCHRY3  /93.0,93.0,100.0,100.0,93.0/                         ADM00990
      DATA YCHRY3  /73.0,80.0,80.0,73.0,73.0/                           ADM01000
C - Chair                                                               ADM01010
      DATA XCHR1   /00.0,00.0,01.0,01.0,06.0,06.0,07.0,07.0,00.0/       ADM01020
      DATA YCHR1   /00.0,07.0,07.0,00.0,00.0,07.0,07.0,00.0,00.0/       ADM01030
      DATA XCHR2   /01.0,06.0/                                          ADM01040
      DATA YCHR2   /06.0,06.0/                                          ADM01050
```

```
      DATA XCHR3   /00.0,00.0,07.0,07.0,00.0/                          ADM01060
      DATA YCHR3   /00.0,07.0,07.0,00.0,00.0/                          ADM01070
C                                                                      ADM01080
C*********************************************************************ADM01090
C*    INITIALISATION                                                 *ADM01100
C*********************************************************************ADM01110
C                                                                      ADM01120
C***  OPEN GKS                                                         ADM01130
      CALL GOPKS(1,0)                                                  ADM01140
C                                                                      ADM01150
C*********************************************************************ADM01160
C*    Open and activate all the workstations to be used              *ADM01170
C*********************************************************************ADM01180
C                                                                      ADM01190
      CALL GOPWK(WKID1,1,WKTYPE)                                       ADM01200
      CALL GACWK(WKID1)                                                ADM01210
C                                                                      ADM01220
C*********************************************************************ADM01230
C*    Set normalization and workstation transformation               *ADM01240
C*********************************************************************ADM01250
C                                                                      ADM01260
C  -  Set world window (user units)                                    ADM01270
      CALL GSWN(1,0.0,100.0,0.0,100.0)                                 ADM01280
C  -  Inquire maximum display surface size                             ADM01290
      CALL GQDSP(WKTYPE,ERRIND,DCUNIT,XDC,YDC,XRAS,YRAS)               ADM01300
C  -  Find the larger of the two dimensions                            ADM01310
      RTEMP = XDC                                                      ADM01320
      IF (RTEMP .LT. YDC) RTEMP = YDC                                  ADM01330
C  -  Calculate aspect ratio of the display surface                    ADM01340
      XNDC = XDC / RTEMP                                               ADM01350
      YNDC = YDC / RTEMP                                               ADM01360
C  -  Set viewport and WK window to same ratio                         ADM01370
      CALL GSVP(1,0.0,XNDC,0.0,YNDC)                                   ADM01380
      CALL GSWKWN(WKID1,0.0,XNDC,0.0,YNDC)                             ADM01390
C  -  Select transformation 1                                          ADM01400
      CALL GSELNT(1)                                                   ADM01410
C                                                                      ADM01420
C*********************************************************************ADM01430
C*    Output initial segments to display                             *ADM01440
C*********************************************************************ADM01450
C                                                                      ADM01460
C***  Create segment 100: 2 items                                      ADM01470
      CALL GCRSG(100)                                                  ADM01480
C                                                                      ADM01490
C   Set pick ID to 101 for the first item                              ADM01500
      CALL GSPKID(101)                                                 ADM01510
C  -  Set fill color to background color                               ADM01520
      CALL GSFACI(0)                                                   ADM01530
C  -  Set fill interior style to solid                                 ADM01540
      CALL GSFAIS(1)                                                   ADM01550
C  -  Output an invisible fill area                                    ADM01560
      CALL GFA(5,XDRY,YDRY)                                            ADM01570
C  -  Set line color to index 1                                        ADM01580
      CALL GSPLCI(1)                                                   ADM01590
C  -  Output a polyline to make a desk                                 ADM01600
      CALL GPL(5,XDRY,YDRY)                                            ADM01610
C  -  Set text color and output text identifier                        ADM01620
      CALL GSTXCI(3)                                                   ADM01630
```

```
      CALL GTXS(78.5,92.0,4,'desk')                                    ADM01640
C                                                                      ADM01650
C   Set pick ID to 102 for the second item                             ADM01660
      CALL GSPKID(102)                                                 ADM01670
C  -  Output an invisible fill area                                    ADM01680
      CALL GFA(5,XCHRY3,YCHRY3)                                        ADM01690
C  -  Output 2 polylines to make a chair                               ADM01700
      CALL GPL(9,XCHRY1,YCHRY1)                                        ADM01710
      CALL GPL(2,XCHRY2,YCHRY2)                                        ADM01720
C  -  Output text identifier                                           ADM01730
      CALL GTXS(86.0,73.5,5,'chair')                                   ADM01740
C                                                                      ADM01750
C***  Close segment 100                                                ADM01760
      CALL GCLSG                                                       ADM01770
C***  Make the segment detectable                                      ADM01780
      CALL GSDTEC(100,1)                                               ADM01790
C                                                                      ADM01800
C***  Create segment 200 : Room outline                                ADM01810
      CALL GCRSG(200)                                                  ADM01820
C  -  Set line color                                                   ADM01830
      CALL GSPLCI(2)                                                   ADM01840
C  -  Output lines for room outline                                    ADM01850
      CALL GPL(7,XRRY1,YRRY1)                                          ADM01860
C                                                                      ADM01870
C***  Close segment 200                                                ADM01880
      CALL GCLSG                                                       ADM01890
C                                                                      ADM01900
C***  Create segment 300: Choice menu                                  ADM01910
      CALL GCRSG(300)                                                  ADM01920
      CALL GSTXCI(6)                                                   ADM01930
      CALL GTXS(02.0,26.0,14,'PF1-pick furn.')                         ADM01940
      CALL GTXS(02.0,22.0,15,'PF2-place furn.')                        ADM01950
      CALL GTXS(02.0,18.0,16,'PF3-rotate furn.')                       ADM01960
      CALL GTXS(53.0,26.0,16,'PF4-remove furn.')                       ADM01970
      CALL GTXS(53.0,22.0, 8,'PF5-exit')                               ADM01980
C                                                                      ADM01990
C***  Close segment 300                                                ADM02000
      CALL GCLSG                                                       ADM02010
C                                                                      ADM02020
C*********************************************************************ADM02030
C*    Initialize input functions                                   *   ADM02040
C*********************************************************************ADM02050
C                                                                      ADM02060
C***  Initialize valuator                                              ADM02070
      MINXDC = XDC/8.0                                                 ADM02080
      MINYDC = YDC/10.0                                                ADM02090
      CALL GINVL(WKID1,1,0.0,1,MINXDC,XDC,MINYDC,YDC,                  ADM02100
     $     -360.0,+360.0,0,DATREC)                                     ADM02110
C                                                                      ADM02120
C***  Initialize locator with initial cursor position                  ADM02130
      CALL GINLC(WKID1,1,1,29.93,69.51,1,0.0,XDC,0.0,YDC,0,            ADM02140
     $     DATREC)                                                     ADM02150
C                                                                      ADM02160
C***  Set viewport input priority                                      ADM02170
      CALL GSVPIP(1,0,0)                                               ADM02180
C                                                                      ADM02190
C***  Set attributes and initial flags                                 ADM02200
```

```
C                                                                       ADM02210
C     Set all the following output text color to 2                      ADM02220
      CALL GSTXCI(2)                                                    ADM02230
C                                                                       ADM02240
C     Set the type of all following lines to solid                      ADM02250
      CALL GSLN(1)                                                      ADM02260
C                                                                       ADM02270
C     Initialise segment number, message flag                           ADM02280
      SEGNUM = 0                                                        ADM02290
      MSGFLG = 0                                                        ADM02300
C                                                                       ADM02310
C*********************************************************************  ADM02320
C*   BEGIN USER INPUT - REPEAT UNTIL OPTION 5(EXIT) IS PICKED       *   ADM02330
C*********************************************************************  ADM02340
C                                                                       ADM02350
C    Inquire number of segments on workstation.  If there are no        ADM02360
C    segments other than the 3 segments in the initial screen, set      ADM02370
C    pick flag and position flag to be off                              ADM02380
C                                                                       ADM02390
1000  CALL GQSGWK(WKID1,1,ERRIND,NUMSEG,SEGN)                           ADM02400
      IF (NUMSEG .GT. 3) GOTO 1100                                      ADM02410
          PCKFLG = 0                                                    ADM02420
          POSFLG = 0                                                    ADM02430
1100  CONTINUE                                                          ADM02440
C                                                                       ADM02450
C***  If a message already exists do not output another                 ADM02460
C                                                                       ADM02470
      IF (MSGFLG .NE. 0) GOTO 1200                                      ADM02480
C         Call REDRAW ALL SEGMENTS to clear msg area                    ADM02490
          CALL GRSGWK(WKID1)                                            ADM02500
          CALL GTXS(1.0,1.0,30,'Enter option 1-5 using PF key ')        ADM02510
1200  MSGFLG = 0                                                        ADM02520
C                                                                       ADM02530
C***  Request choice until status is OK and choice is 1 to 5            ADM02540
C                                                                       ADM02550
1300  CALL GRQCH(WKID1,2,STAT,CHCNUM)                                   ADM02560
      IF ((STAT .NE. 1) .OR. (CHCNUM .GT. 5) .OR. (CHCNUM .LT. 1))      ADM02570
     $   GOTO 1300                                                      ADM02580
C                                                                       ADM02590
C   *****************************************************************   ADM02600
C   ***IF PICK FURNITURE IS CHOSEN                                 *   ADM02610
C   *****************************************************************   ADM02620
C                                                                       ADM02630
      IF (CHCNUM .NE. 1) GOTO 2400                                      ADM02640
C                                                                       ADM02650
C         Call REDRAW ALL SEGMENTS to clear msg area                    ADM02660
C         Display msg asking user to pick furniture                     ADM02670
C                                                                       ADM02680
          CALL GRSGWK(WKID1)                                            ADM02690
          CALL GTXS(1.0,1.0,15,'Pick furniture ')                       ADM02700
2000      CALL GRQPK (WKID1, 1, STAT, SEGNM, PICKID)                    ADM02710
          IF (STAT .EQ. 0) GOTO 2000                                    ADM02720
C                                                                       ADM02730
C ***     Test for invalid segment and status equal 2 'no pick'         ADM02740
          IF ((SEGNM .EQ. 100) .AND. (STAT .NE. 2)) GOTO 2100           ADM02750
C             call REDRAW ALL SEGMENTS to clear msg area                ADM02760
              CALL GRSGWK(WKID1)                                        ADM02770
              CALL GSTXI(2)                                             ADM02780
```

```
              CALL GTXS(1.0,1.0,35,                                      ADM02790
     $             'Selection is invalid - pick again  ')               ADM02800
              CALL GSTXI(1)                                              ADM02810
              GOTO 2000                                                  ADM02820
2100      CONTINUE                                                       ADM02830
C                                                                        ADM02840
          PCKFLG = 1                                                     ADM02850
          SEGNUM = SEGNUM + 1                                            ADM02860
C                                                                        ADM02870
C***      Switch according to the furniture picked                       ADM02880
          IF (PICKID .NE. 101) GOTO 2200                                 ADM02890
C         Create a desk segment                                          ADM02900
              CALL GCRSG(SEGNUM)                                         ADM02910
              CALL GSPKID(SEGNUM)                                        ADM02920
C          -  Set the segment invisible before it is placed              ADM02930
              CALL GSVIS(SEGNUM,0)                                       ADM02940
C          -  Output a fill area with background color                   ADM02950
              CALL GFA(5,XDSK,YDSK)                                      ADM02960
              CALL GPL(5,XDSK,YDSK)                                      ADM02970
              CALL GCLSG                                                 ADM02980
              FURWDT=12.5                                                ADM02990
              FURHIG=6.5                                                 ADM03000
2200      CONTINUE                                                       ADM03010
C                                                                        ADM03020
          IF (PICKID .NE. 102) GOTO 2300                                 ADM03030
C         Create a chair segment                                         ADM03040
              CALL GCRSG(SEGNUM)                                         ADM03050
              CALL GSPKID(SEGNUM)                                        ADM03060
C          -  Set the segment invisible before it is placed              ADM03070
              CALL GSVIS(SEGNUM,0)                                       ADM03080
C          -  Output a fill with background color                        ADM03090
              CALL GFA(5,XCHR3,YCHR3)                                    ADM03100
              CALL GPL(9,XCHR1,YCHR1)                                    ADM03110
              CALL GPL(2,XCHR2,YCHR2)                                    ADM03120
              CALL GCLSG                                                 ADM03130
              FURWDT=3.5                                                 ADM03140
              FURHIG=3.5                                                 ADM03150
2300      CONTINUE                                                       ADM03160
2400  CONTINUE                                                           ADM03170
C                                                                        ADM03180
C  ****************************************************************      ADM03190
C  ***IF PLACE FURNITURE IS CHOSEN                               *      ADM03200
C  ****************************************************************      ADM03210
C                                                                        ADM03220
      IF (CHCNUM .NE. 2) GOTO 3100                                       ADM03230
C***      If no furniture is picked yet, display error message and exit ADM03240
          IF (PCKFLG .NE. 0) GOTO 2900                                   ADM03250
C             Call REDRAW ALL SEGMENTS to clear msg area                 ADM03260
              CALL GRSGWK(WKID1)                                         ADM03270
              CALL GTXS(1.0,1.0,29,'You must pick furniture first')      ADM03280
              MSGFLG = 1                                                 ADM03290
              GOTO 1000                                                  ADM03300
2900      CONTINUE                                                       ADM03310
C                                                                        ADM03320
C***      Display msg asking user to place the furniture                 ADM03330
C         Call REDRAW ALL SEGMENTS to clear msg area                     ADM03340
          CALL GRSGWK(WKID1)                                             ADM03350
```

```
          CALL GTXS(1.0,1.0,33,'Place furniture in the room      ')     ADM03360
C                                                                       ADM03370
C***      Request locator until the furniture is inside the room        ADM03380
3000      CALL GRQLC(WKID1, 1, STAT, TRANUM, XPOP, YPOP)                ADM03390
          IF ((STAT .EQ. 0) .OR. (XPOP .GT. 60.0) .OR. (YPOP .LT. 40.0))ADM03400
     $        GOTO 3000                                                 ADM03410
          POSFLG = 1                                                    ADM03420
C                                                                       ADM03430
C***      Transform the furniture and make the center point of the      ADM03440
C         furniture locate at the point just entered                    ADM03450
C                                                                       ADM03460
          SHFTX = XPOP-FURWDT                                           ADM03470
          SHFTY = YPOP-FURHIG                                           ADM03480
          CALL GEVTM(0.0,0.0,SHFTX,SHFTY,0.0,1.0,1.0,0,MTX)             ADM03490
          CALL GSSGT(SEGNUM,MTX)                                        ADM03500
C                                                                       ADM03510
C***      Make the segment visible                                      ADM03520
          CALL GSVIS(SEGNUM,1)                                          ADM03530
C***      and detectable                                                ADM03540
          CALL GSDTEC(SEGNUM,1)                                         ADM03550
3100  CONTINUE                                                          ADM03560
C                                                                       ADM03570
C  ******************************************************************   ADM03580
C  ***IF ROTATE FURNITURE IS CHOSEN                                 *   ADM03590
C  ******************************************************************   ADM03600
C                                                                       ADM03610
      IF (CHCNUM .NE. 3) GOTO 4300                                      ADM03620
C                                                                       ADM03630
C***      If this furniture is not positioned yet, display error        ADM03640
C         and exit                                                      ADM03650
          IF (POSFLG .NE. 0) GOTO 3900                                  ADM03660
C             Call REDRAW ALL SEGMENTS  to clear msg area               ADM03670
              CALL GRSGWK(WKID1)                                        ADM03680
              CALL GSTXI(2)                                             ADM03690
              CALL GTXS(1.0,1.0,38,                                     ADM03700
     $            'You must position furniture first     ')            ADM03710
              CALL GSTXI(1)                                             ADM03720
              MSGFLG = 1                                                ADM03730
              GOTO 1000                                                 ADM03740
3900      CONTINUE                                                      ADM03750
C                                                                       ADM03760
C***      Display msg asking user to enter rotation angle               ADM03770
C         Call REDRAW ALL SEGMENTS to clear msg area                    ADM03780
          CALL GRSGWK(WKID1)                                            ADM03790
          CALL GTXS(1.0,1.0,33,'Enter rotation angle in degrees  ')     ADM03800
          CALL GTXS(1.0,10.0, 6,'angle:')                               ADM03810
4000      CALL GRQVL(WKID1,1,STAT,ANG)                                  ADM03820
          IF (STAT .NE. 1 ) GOTO 4000                                   ADM03830
C                                                                       ADM03840
C         Convert degrees to radians                                    ADM03850
4200      ANG = ANG/180*3.1416                                          ADM03860
C                                                                       ADM03870
C***      Accumulate the transformation matrix and set transformation   ADM03880
          CALL GACTM(MTX,XPOP,YPOP,0.0,0.0,ANG,1.0,1.0,0,MTX2)          ADM03890
          CALL GSSGT(SEGNUM,MTX2)                                       ADM03900
C                                                                       ADM03910
4300  CONTINUE                                                          ADM03920
C                                                                       ADM03930
```

```
C  ****************************************************************   ADM03940
C  ***IF REMOVE FURNITURE IS CHOSEN                                 *   ADM03950
C  ****************************************************************   ADM03960
C                                                                      ADM03970
      IF (CHCNUM .NE. 4) GOTO 5100                                     ADM03980
C                                                                      ADM03990
C***      If no furniture is in the room, display error message       ADM04000
          IF (NUMSEG .GT. 3) GOTO 4900                                 ADM04010
C                                                                      ADM04020
C             Call REDRAW ALL SEGMENTS to clear msg area               ADM04030
              CALL GRSGWK(WKID1)                                       ADM04040
              CALL GSTXI(2)                                            ADM04050
              CALL GTXS(1.0,1.0,33,'No furniture is in the room      ') ADM04060
              CALL GSTXI(1)                                            ADM04070
              MSGFLG = 1                                               ADM04080
              GOTO 1000                                                ADM04090
4900      CONTINUE                                                     ADM04100
C                                                                      ADM04110
C***      Display msg asking user to pick the furniture to be deleted  ADM04120
C         Call REDRAW ALL SEGMENTS to clear msg area                   ADM04130
          CALL GRSGWK(WKID1)                                           ADM04140
          CALL GTXS(1.0,1.0,33,'Pick the furniture to be removed ')    ADM04150
5000      CALL GRQPK(WKID1, 1, STAT, SEGNM, PICKID)                    ADM04160
          IF ((STAT .NE. 1) .OR. (PICKID .GE. 100)) GOTO 5000          ADM04170
C                                                                      ADM04180
C***      Delete the furniture segment                                 ADM04190
          CALL GDSG(SEGNM)                                             ADM04200
C***      Show segment does not exist any more                         ADM04210
          PCKFLG = 0                                                   ADM04220
          POSFLG = 0                                                   ADM04230
C                                                                      ADM04240
5100  CONTINUE                                                         ADM04250
C                                                                      ADM04260
C  ****************************************************************   ADM04270
C  ***IF EXIT IS CHOSEN                                             *   ADM04280
C  ****************************************************************   ADM04290
      IF (CHCNUM .EQ. 5) GOTO 9999                                     ADM04300
C                                                                      ADM04310
C     Go back to do another request choice                             ADM04320
      GOTO 1000                                                        ADM04330
C                                                                      ADM04340
C                                                                      ADM04350
C*****************************************************************   ADM04360
C   TERMINATION                                                     *   ADM04370
C*****************************************************************   ADM04380
C                                                                      ADM04390
C  -  Deactivate and close the workstation                             ADM04400
9999  CALL GDAWK (WKID1)                                               ADM04410
      CALL GCLWK (WKID1)                                               ADM04420
C                                                                      ADM04430
C***  CLOSE GKS                                                        ADM04440
      CALL GCLKS                                                       ADM04450
      STOP                                                             ADM04460
      END                                                              ADM04470
```

Appendix E. Example programs

This appendix contains two example programs:

METOUT displays a GDDM-GKS metafile, or saves the picture as a GDF file.

METCNV converts data in a GDDM-GKS metafile to its character equivalent (or the reverse operation).

Both programs are written for the VS FORTRAN compiler.

```
C************************************************************************ 01200000
C*                5666-802                                              * 01800000
C*                (C) COPYRIGHT IBM CORP. 1979,1987                     * 02400000
C*                LICENSED MATERIALS - PROPERTY OF IBM                  * 03000000
C*                                                                      * 03600000
C*                       METOUT                                         * 04200000
C*                                                                      * 04800000
C*   Program to display a metafile created by GDDM-GKS                  * 05400000
C*   and/or save the picture as a GDF file.                             * 06000000
C*                                                                      * 06600000
C*   The program requests the number of the metafile                    * 07200000
C*   to be input and then whether the picture is to be displayed        * 07800000
C*   or saved as GDF                                                    * 08400000
C*   If the picture is to be saved the GDF file number is               * 09000000
C*   requested                                                          * 09600000
C*                                                                      * 10200000
C*   If the file cannot be found, or another error                      * 10800000
C*   is detected, the error message is displayed.                       * 11400000
C*                                                                      * 12000000
C*   The program terminates when the picture has been                   * 12600000
C*   displayed and the ENTER key has been pressed, or when              * 13200000
C*   the picture has been saved.                                        * 13800000
C*                                                                      * 14400000
C*                                                                      * 15000000
C************************************************************************ 15600000
C                                                                         16200000
C                                                                         16800000
C   Connection identifier for input metafile                              17400000
      INTEGER METAN                                                       18000000
C   Returned choice data                                                  18600000
      INTEGER STAT,CHNR                                                   19200000
C   Metafile item data                                                    19800000
C   - This program assumes that any item data record will fit in to       20400000
C     a data record of 100 * 80 characters.                               21000000
      INTEGER MITYP,MILEN                                                 21600000
      PARAMETER (MAXREC=10)                                               21900000
      PARAMETER (MAXLEN=MAXREC*80)                                        22200000
      CHARACTER*80 DATREC(MAXREC)                                         22500000
C                                                                         22800000
C   Response to prompt for Display or Save action                         23400000
      CHARACTER ACTION /'D'/                                              24000000
C                                                                         24600000
C   Workstation connection identifier and type                            25200000
      INTEGER CONNID /1/, WKTYPE /1/                                      25800000
C                                                                         26400000
C   GKS Operating state                                                   27000000
```

© Copyright IBM Corp. 1984, 1987

```
      INTEGER OPSTAT /0/                                                27600000
C                                                                       28200000
C   ERROR HANDLING                                                      28800000
C                                                                       29400000
      INTEGER*4 ERRIND /0/                                              30000000
C                                                                       30600000
C********************************************************************** 31200000
C**                                                                  ** 31800000
C**       MAIN PROGRAM                                               ** 32400000
C**                                                                  ** 33000000
C********************************************************************** 33600000
C*  Output header                                                       34200000
C                                                                       34800000
      WRITE (*,*) ' '                                                   35400000
      WRITE (*,*) 'METOUT sample program  '                             36000000
      WRITE (*,*) ' '                                                   36600000
C                                                                       37200000
C Initialize the input metafile number                                  37800000
C the output workstation connection identifier is 1 by default          38400000
C                                                                       39000000
      METAN = -1                                                        39600000
C                                                                       40200000
C Request the metafile number from the operator                         40800000
C                                                                       41400000
      WRITE (*,*) 'type the input metafile identifier (0 to 9999)'      42000000
      READ  (*,*,END=50) METAN                                          42600000
50    CONTINUE                                                          43200000
C                                                                       43800000
C Check the returned metafile number                                    44400000
C                                                                       45000000
      IF ((METAN.LT.0).OR.(METAN.GT.9999)) THEN                         45600000
        WRITE(*,*) 'Metafile identifier is invalid'                     46200000
        GOTO 9999                                                       46800000
      ENDIF                                                             47400000
C                                                                       48000000
C Request the action to be performed                                    48600000
C                                                                       49200000
      WRITE (*,*) 'Type D to display or S to save the picture as GDF'   49800000
      ACTION='D'                                                        50000000
      READ(*,'(A)',END=100) ACTION                                      51000000
100   CONTINUE                                                          51600000
C                                                                       52200000
C Request the GDF file connection id if Save was requested              52800000
C                                                                       53400000
      IF (ACTION.EQ.'S') THEN                                           54000000
        WRITE (*,*) 'Type output GDF file identifier (0 to 9999)'       54600000
        READ  (*,*,END=150) CONNID                                      55200000
150     CONTINUE                                                        55800000
C                                                                       56400000
C Check the returned GDF file connection id                             57000000
C                                                                       57600000
        IF ((CONNID.LT.0).OR.(CONNID.GT.9999)) THEN                     58200000
          WRITE(*,*) 'GDF file identifier is invalid'                   58800000
          GOTO 9999                                                     59400000
        ENDIF                                                           60000000
C And set the rquested workstation type: GDF file output                60600000
      WKTYPE = 5                                                        61200000
```

```
      ENDIF                                                              61800000
C******************************************************************** 62400000
C**                                                                  ** 63000000
C**         OPEN GKS AND THE REQUIRED WORKSTATIONS                   ** 63600000
C******************************************************************** 64200000
C** This program uses error file ADMJ0000                             64800000
      CALL GOPKS(0,0)                                                  65400000
C** Open the workstation required for output                          66000000
      CALL GOPWK(1,CONNID,WKTYPE)                                      66600000
C** and activate it                                                   67200000
      CALL GACWK(1)                                                    67800000
C** Open the metafile input workstation                              68500000
      CALL GOPWK(2,METAN,4)                                            69200000
C                                                                      69900000
C   Now perform a loop retrieving the item type, item length and the   70600000
C   data record for each item and then interpreting the item           71300000
C                                                                      72000000
5000  CONTINUE                                                         72700000
C- Get the item type and length                                        73400000
      CALL GGTITM(2,MITYP,MILEN)                                       74100000
C- If the item is the END item we must leave the loop                  74800000
      IF (MITYP.EQ.0) GOTO 5999                                        75500000
C- else - skip the item if the data record will not fit in datrec      75700000
C         or it is a user item                                         76000000
        IF ((MILEN.GT.MAXLEN).OR.(MITYP.GT.100)) THEN                  76200000
          CALL GRDITM(2,0,MAXREC,DATREC)                               76600000
        ELSE                                                           76900000
C-      - read and the item data record                                77200000
          CALL GRDITM(2,MILEN,MAXREC,DATREC)                           77500000
C-        and interpret the item                                       77600000
          CALL GIITM(MITYP,MILEN,MAXREC,DATREC)                        78100000
        ENDIF                                                          78300000
C-        continue with the next item                                  78700000
        GOTO 5000                                                      79000000
5999  CONTINUE                                                         79700000
C-  update the workstation to ensure display is correct               80400000
      CALL GUWK(1,1)                                                   81100000
C-  If the output workstation is the console request enter key from    81800000
C   the operator                                                       82500000
      IF (WKTYPE.EQ.1) CALL GRQCH(1,1,STAT,CHNR)                       83200000
C-  Deactivate the output workstation                                  83900000
      CALL GDAWK(1)                                                    84600000
C-  and close it                                                       85300000
      CALL GCLWK(1)                                                    86000000
C-  close the metafile input workstation                              86700000
      CALL GCLWK(2)                                                    87400000
C-  close GKS                                                         88100000
      CALL GCLKS                                                       88800000
9999  CONTINUE                                                         89500000
C**** Termination                                                      90200000
      STOP                                                             90900000
      END                                                              91600000
C                                                                      92300000
C                                                                      93000000
      SUBROUTINE GERHND (ERRNR,FCTID,ERRFIL)                           93700000
C******************************************************************** 94400000
C GERHND - This subroutine replaces GKS default error handling       * 95100000
C******************************************************************** 95800000
```

```
      INTEGER ERRNR,FCTID,ERRFIL                                    96500000
C- Call GERLOG to perform error logging                             97200000
      CALL GERLOG (ERRNR,FCTID,ERRFIL)                              97900000
      RETURN                                                        98600000
      END                                                           99300000
```

```
C     ****************************************************************00100000
C     **  P R O G R A M   M E T C N V                                **00200000
C     ****************************************************************00300000
C     **                                                            **00400000
C     **          5666-802                                          **00600000
C     **          (C) COPYRIGHT IBM CORP. 1987                      **00800000
C     **          LICENSED MATERIALS - PROPERTY OF IBM              **01000000
C     **                                                            **01200000
C     **  This program converts data between a GKS metafile and a   **01400000
C     **  character equivalent of that data.  The conversion can be **01600000
C     **  done in either direction (metafile to character or char-  **01800000
C     **  acter to metafile).                                       **02000000
C     **                                                            **02200000
C     **  Three I/O units (datasets) are used:                      **02400000
C     **  - input options are read from unit "IIN"                  **02600000
C     **  - output messages are written to unit "IOUT"              **02800000
C     **  - the character form of the metafile is read from or      **03000000
C     **    written to unit "ICHR".                                 **03200000
C     **                                                            **03400000
C     ****************************************************************03600000
C                                                                     03800000
      PROGRAM METCNV                                                 04000000
C                                                                     04200000
C**** DATA RECORD ARRAY                                             04400000
      PARAMETER (MAXREC=10)                                          04600000
      PARAMETER (MAXLEN=MAXREC*80)                                   04800000
      CHARACTER*80 DATREC(MAXREC)                                    05000000
C                                                                     05200000
C**** CHAR/INTEGER/REAL EQUIVALENTS OF "DATREC"                     05400000
      PARAMETER (MXCDAT=MAXREC*80)                                   05600000
      PARAMETER (MXIDAT=MXCDAT/4)                                    05800000
      PARAMETER (MXRDAT=MXCDAT/4)                                    06000000
      CHARACTER*1  CDAT(MXCDAT)                                      06200000
      DIMENSION    IDAT(MXIDAT)                                      06400000
      DIMENSION    RDAT(MXRDAT)                                      06600000
      EQUIVALENCE (DATREC(1),CDAT(1),IDAT(1),RDAT(1))                06800000
C                                                                     07000000
C**** CONVERSION TYPE                                               07200000
      PARAMETER (ICNVMC=1)                                           07400000
      PARAMETER (ICNVCM=2)                                           07600000
C**** MAXIMUM (NON-USER) METAFILE ITEM TYPE                         07800000
      PARAMETER (MAXTYP=100)                                         08000000
C                                                                     08200000
C                                                                     08400000
C     ****************************************************************08600000
C     **  I N I T I A L I S A T I O N                               **08800000
C     ****************************************************************09000000
C                                                                     09200000
C**** INITIALISE INPUT/OUTPUT UNIT NUMBERS                          09400000
      IIN=5                                                          09600000
      IOUT=6                                                         09800000
      ICHR=7                                                         10000000
C**** INITIALISE GKS WORKSTATION IDENTIFIER                         10200000
      IWKID=1                                                        10400000
C                                                                     10600000
C**** GET TYPE OF CONVERSION REQUIRED                               10800000
      WRITE(IOUT,100)ICNVMC,ICNVCM                                   11000000
  100 FORMAT(/1X,'ENTER TYPE OF CONVERSION REQUIRED:'/               11200000
```

```
     &  3X,I1,' = METAFILE TO CHARACTER'/                                11400000
     &  3X,I1,' = CHARACTER TO METAFILE')                                11600000
      READ(IIN,110)ICNV                                                  11800000
  110 FORMAT(I1)                                                         12000000
C**** CHECK VALUE ENTERED                                                12200000
      IF (ICNV.EQ.ICNVCM.OR.ICNV.EQ.ICNVMC) THEN                         12400000
        WRITE(IOUT,120)ICNV                                              12600000
  120   FORMAT(/1X,'CONVERSION TYPE SPECIFIED = ',I1)                    12800000
      ELSE                                                               13000000
        WRITE(IOUT,130)ICNV                                              13200000
  130   FORMAT(/1X,'CONVERSION TYPE ',I1,' IS NOT VALID.'/               13400000
     &    3X,'EXECUTION IS TERMINATED.')                                 13600000
        GOTO 3300                                                        13800000
      ENDIF                                                              14000000
C                                                                        14200000
C**** GET NUMBER OF METAFILE TO BE USED                                  14400000
      WRITE(IOUT,150)                                                    14600000
  150 FORMAT(/1X,'ENTER NUMBER OF METAFILE TO BE USED (1-4 DIGITS):')    14800000
      READ(IIN,160)IFILE                                                 15000000
  160 FORMAT(BN,I4)                                                      15200000
C**** CHECK VALUE ENTERED                                                15400000
      IF (IFILE.GE.0) THEN                                               15600000
        WRITE(IOUT,170)IFILE                                             15800000
  170   FORMAT(/1X,'METAFILE NUMBER SPECIFIED = ',I4)                    16000000
      ELSE                                                               16200000
        WRITE(IOUT,180)IFILE                                             16400000
  180   FORMAT(/BN,1X,'METAFILE NUMBER ',I4,' IS NOT VALID.'/            16600000
     &    3X,'EXECUTION IS TERMINATED.')                                 16800000
        GOTO 3300                                                        17000000
      ENDIF                                                              17200000
C                                                                        17400000
C**** OPEN GKS                                                           17600000
      CALL GOPKS (0,0)                                                   17800000
C**** OPEN APPROPRIATE METAFILE WORKSTATION (INPUT OR OUTPUT)            18000000
      IF (ICNV.EQ.ICNVCM) THEN                                           18200000
        CALL GOPWK (IWKID,IFILE,3)                                       18400000
        CALL GACWK (IWKID)                                               18600000
      ELSE                                                               18800000
        CALL GOPWK (IWKID,IFILE,4)                                       19000000
      ENDIF                                                              19200000
C**** INITIALIZE METAFILE ITEM COUNTER                                   19400000
      INUM=0                                                             19600000
C                                                                        19800000
C                                                                        20000000
C     ********************************************************************20200000
C     **  M E T A F I L E   I T E M   L O O P                          **20400000
C     ********************************************************************20600000
C                                                                        20800000
C**** PROCESS METAFILE ITEMS                                             21000000
  200 INUM=INUM+1                                                        21200000
C                                                                        21400000
C**** READ/GET ITEM TYPE AND LENGTH                                      21600000
      IF (ICNV.EQ.ICNVCM) THEN                                           21800000
        READ (ICHR,900)ITYP,ILEN                                         22000000
      ELSE                                                               22200000
        CALL GGTITM (IWKID,ITYP,ILEN)                                    22400000
```

```
      ENDIF                                                              22600000
C                                                                        22800000
C**** CHECK THAT "DATREC" CAN CONTAIN ALL THE DATA                       23000000
      IF (ILEN.GT.MAXLEN) THEN                                           23200000
        WRITE(IOUT,220)INUM,ITYP,ILEN,MAXLEN                             23400000
  220   FORMAT(/1X,'METAFILE ITEM ',I6,' WITH TYPE = ',I6,               23600000
     &    ' AND LENGTH = ',I6,' IS TOO LONG.'/                           23800000
     &    3X,'EXECUTION IS TERMINATED. (MAX LENGTH SUPPORTED = ',I6,')')24000000
        GOTO 3200                                                        24200000
      ENDIF                                                              24400000
C                                                                        24600000
C**** GET ITEM DATA, AND WRITE ITEM TYPE AND LENGTH                      24800000
      IF (ICNV.EQ.ICNVMC) THEN                                           25000000
        CALL GRDITM (IWKID,MAXLEN,MAXREC,DATREC)                         25200000
        WRITE(ICHR,900)ITYP,ILEN                                         25400000
      ENDIF                                                              25600000
C                                                                        25800000
C**** READ/WRITE ITEM DATA ACCORDING TO ITEM TYPE                        26000000
      IF (ITYP.EQ. 0) GOTO 1000                                          26200000
      IF (ITYP.EQ. 1) GOTO 1001                                          26400000
      IF (ITYP.EQ. 2) GOTO 1002                                          26600000
      IF (ITYP.EQ. 3) GOTO 1003                                          26800000
      IF (ITYP.EQ. 4) GOTO 1004                                          27000000
      IF (ITYP.EQ. 5) GOTO 1005                                          27200000
      IF (ITYP.EQ.11) GOTO 1011                                          27400000
      IF (ITYP.EQ.12) GOTO 1012                                          27600000
      IF (ITYP.EQ.13) GOTO 1013                                          27800000
      IF (ITYP.EQ.14) GOTO 1014                                          28000000
      IF (ITYP.EQ.15) GOTO 1015                                          28200000
      IF (ITYP.EQ.21) GOTO 1021                                          28400000
      IF (ITYP.EQ.22) GOTO 1022                                          28600000
      IF (ITYP.EQ.23) GOTO 1023                                          28800000
      IF (ITYP.EQ.24) GOTO 1024                                          29000000
      IF (ITYP.EQ.25) GOTO 1025                                          29200000
      IF (ITYP.EQ.26) GOTO 1026                                          29400000
      IF (ITYP.EQ.27) GOTO 1027                                          29600000
      IF (ITYP.EQ.28) GOTO 1028                                          29800000
      IF (ITYP.EQ.29) GOTO 1029                                          30000000
      IF (ITYP.EQ.30) GOTO 1030                                          30200000
      IF (ITYP.EQ.31) GOTO 1031                                          30400000
      IF (ITYP.EQ.32) GOTO 1032                                          30600000
      IF (ITYP.EQ.33) GOTO 1033                                          30800000
      IF (ITYP.EQ.34) GOTO 1034                                          31000000
      IF (ITYP.EQ.35) GOTO 1035                                          31200000
      IF (ITYP.EQ.36) GOTO 1036                                          31400000
      IF (ITYP.EQ.37) GOTO 1037                                          31600000
      IF (ITYP.EQ.38) GOTO 1038                                          31800000
      IF (ITYP.EQ.39) GOTO 1039                                          32000000
      IF (ITYP.EQ.40) GOTO 1040                                          32200000
      IF (ITYP.EQ.41) GOTO 1041                                          32400000
      IF (ITYP.EQ.42) GOTO 1042                                          32600000
      IF (ITYP.EQ.43) GOTO 1043                                          32800000
      IF (ITYP.EQ.44) GOTO 1044                                          33000000
      IF (ITYP.EQ.51) GOTO 1051                                          33200000
      IF (ITYP.EQ.52) GOTO 1052                                          33400000
      IF (ITYP.EQ.53) GOTO 1053                                          33600000
      IF (ITYP.EQ.54) GOTO 1054                                          33800000
      IF (ITYP.EQ.55) GOTO 1055                                          34000000
```

```
      IF (ITYP.EQ.56) GOTO 1056                                         34200000
      IF (ITYP.EQ.61) GOTO 1061                                         34400000
      IF (ITYP.EQ.71) GOTO 1071                                         34600000
      IF (ITYP.EQ.72) GOTO 1072                                         34800000
      IF (ITYP.EQ.81) GOTO 1081                                         35000000
      IF (ITYP.EQ.82) GOTO 1082                                         35200000
      IF (ITYP.EQ.83) GOTO 1083                                         35400000
      IF (ITYP.EQ.84) GOTO 1084                                         35600000
      IF (ITYP.EQ.91) GOTO 1091                                         35800000
      IF (ITYP.EQ.92) GOTO 1092                                         36000000
      IF (ITYP.EQ.93) GOTO 1093                                         36200000
      IF (ITYP.EQ.94) GOTO 1094                                         36400000
      IF (ITYP.EQ.95) GOTO 1095                                         36600000
      IF (ITYP.GT.100) GOTO 1100                                        36800000
C                                                                       37000000
C**** UNSUPPORTED ITEM TYPE                                             37200000
      WRITE(IOUT,250)INUM,ITYP                                          37400000
  250 FORMAT(/1X,'METAFILE ITEM ',I6,' WITH TYPE = ',I6,                37600000
     &  ' IS NOT SUPPORTED.'/3X,'EXECUTION IS TERMINATED.')             37800000
      GOTO 3200                                                         38000000
C                                                                       38200000
C**** METAFILE-ITEM HEADER                                              38400000
  900 FORMAT(2I15)                                                      38600000
C**** METAFILE-ITEM DATA (CHARACTER)                                    38800000
  901 FORMAT(80A1)                                                      39000000
C**** METAFILE-ITEM DATA (INTEGER)                                      39200000
  902 FORMAT(5I15)                                                      39400000
C**** METAFILE-ITEM DATA (REAL)                                         39600000
  903 FORMAT(1P5E15.6)                                                  39800000
C                                                                       40000000
C**** END ITEM                                                          40200000
 1000 CONTINUE                                                          40400000
C**** REDRAW ALL SEGMENTS                                               40600000
 1002 CONTINUE                                                          40800000
C**** CLOSE SEGMENT                                                     41000000
 1082 CONTINUE                                                          41200000
C**** READ/WRITE ITEM DATA                                              41400000
      GOTO 2000                                                         41600000
C                                                                       41800000
C**** CLEAR WORKSTATION                                                 42000000
 1001 CONTINUE                                                          42200000
C**** UPDATE WORKSTATION                                                42400000
 1003 CONTINUE                                                          42600000
C**** POLYLINE INDEX                                                    42800000
 1021 CONTINUE                                                          43000000
C**** LINETYPE                                                          43200000
 1022 CONTINUE                                                          43400000
C**** POLYLINE COLOR INDEX                                              43600000
 1024 CONTINUE                                                          43800000
C**** POLYMARKER INDEX                                                  44000000
 1025 CONTINUE                                                          44200000
C**** MARKER TYPE                                                       44400000
 1026 CONTINUE                                                          44600000
C**** POLYMARKER COLOR INDEX                                            44800000
 1028 CONTINUE                                                          45000000
C**** TEXT INDEX                                                        45200000
 1029 CONTINUE                                                          45400000
C**** TEXT COLOR INDEX                                                  45600000
```

```
 1033 CONTINUE                                                          45800000
C**** TEXT PATH                                                         46000000
 1035 CONTINUE                                                          46200000
C**** FILL AREA INDEX                                                   46400000
 1037 CONTINUE                                                          46600000
C**** FILL AREA INTERIOR STYLE                                          46800000
 1038 CONTINUE                                                          47000000
C**** FILL AREA STYLE INDEX                                             47200000
 1039 CONTINUE                                                          47400000
C**** FILL AREA COLOR INDEX                                             47600000
 1040 CONTINUE                                                          47800000
C**** PICK IDENTIFIER                                                   48000000
 1044 CONTINUE                                                          48200000
C**** CREATE SEGMENT                                                    48400000
 1081 CONTINUE                                                          48600000
C**** DELETE SEGMENT                                                    48800000
 1084 CONTINUE                                                          49000000
C**** READ/WRITE ITEM DATA                                              49200000
      IF (ICNV.EQ.ICNVCM) THEN                                          49400000
        READ (ICHR,902)IDAT(1)                                          49600000
      ELSE                                                              49800000
        WRITE(ICHR,902)IDAT(1)                                          50000000
      ENDIF                                                             50200000
      GOTO 2000                                                         50400000
C                                                                       50600000
C**** DEFERRAL STATE                                                    50800000
 1004 CONTINUE                                                          51000000
C**** TEXT FONT AND PRECISION                                           51200000
 1030 CONTINUE                                                          51400000
C**** TEXT ALIGNMENT                                                    51600000
 1036 CONTINUE                                                          51800000
C**** RENAME SEGMENT                                                    52000000
 1083 CONTINUE                                                          52200000
C**** SET VISIBILITY                                                    52400000
 1092 CONTINUE                                                          52600000
C**** SET HIGHLIGHTING                                                  52800000
 1093 CONTINUE                                                          53000000
C**** SET DETECTABILITY                                                 53200000
 1095 CONTINUE                                                          53400000
C**** READ/WRITE ITEM DATA                                              53600000
      IF (ICNV.EQ.ICNVCM) THEN                                          53800000
        READ (ICHR,902)IDAT(1),IDAT(2)                                  54000000
      ELSE                                                              54200000
        WRITE(ICHR,902)IDAT(1),IDAT(2)                                  54400000
      ENDIF                                                             54600000
      GOTO 2000                                                         54800000
C                                                                       55000000
C**** MESSAGE                                                           55200000
 1005 IF (ICNV.EQ.ICNVCM) THEN                                          55400000
        READ (ICHR,902)IDAT(1)                                          55600000
        IMAX=4+IDAT(1)                                                  55800000
        READ (ICHR,901)(CDAT(I),I=5,IMAX)                               56000000
      ELSE                                                              56200000
        WRITE(ICHR,902)IDAT(1)                                          56400000
        IMAX=4+IDAT(1)                                                  56600000
        WRITE(ICHR,901)(CDAT(I),I=5,IMAX)                               56800000
      ENDIF                                                             57000000
```

```
      GOTO 2000                                                    57200000
C                                                                  57400000
C**** POLYLINE                                                     57600000
 1011 CONTINUE                                                     57800000
C**** POLYMARKER                                                   58000000
 1012 CONTINUE                                                     58200000
C**** FILL AREA                                                    58400000
 1014 CONTINUE                                                     58600000
C**** READ/WRITE ITEM DATA                                         58800000
      IF (ICNV.EQ.ICNVCM) THEN                                     59000000
        READ (ICHR,902)IDAT(1)                                     59200000
        IMAX=1+IDAT(1)*2                                           59400000
        READ (ICHR,903)(RDAT(I),I=2,IMAX)                          59600000
      ELSE                                                         59800000
        WRITE(ICHR,902)IDAT(1)                                     60000000
        IMAX=1+IDAT(1)*2                                           60200000
        WRITE(ICHR,903)(RDAT(I),I=2,IMAX)                          60400000
      ENDIF                                                        60600000
      GOTO 2000                                                    60800000
C                                                                  61000000
C**** TEXT                                                         61200000
 1013 IF (ICNV.EQ.ICNVCM) THEN                                     61400000
        READ (ICHR,903)RDAT(1),RDAT(2)                             61600000
        READ (ICHR,902)IDAT(3)                                     61800000
        IMAX=12+IDAT(3)                                            62000000
        READ (ICHR,901)(CDAT(I),I=13,IMAX)                         62200000
      ELSE                                                         62400000
        WRITE(ICHR,903)RDAT(1),RDAT(2)                             62600000
        WRITE(ICHR,902)IDAT(3)                                     62800000
        IMAX=12+IDAT(3)                                            63000000
        WRITE(ICHR,901)(CDAT(I),I=13,IMAX)                         63200000
      ENDIF                                                        63400000
      GOTO 2000                                                    63600000
C                                                                  63800000
C**** CELL ARRAY                                                   64000000
 1015 IF (ICNV.EQ.ICNVCM) THEN                                     64200000
        READ (ICHR,903)(RDAT(I),I=1,6)                             64400000
        READ (ICHR,902)IDAT(7),IDAT(8)                             64600000
        IMAX=8+IDAT(7)*IDAT(8)                                     64800000
        READ (ICHR,902)(IDAT(I),I=9,IMAX)                          65000000
      ELSE                                                         65200000
        WRITE(ICHR,903)(RDAT(I),I=1,6)                             65400000
        WRITE(ICHR,902)IDAT(7),IDAT(8)                             65600000
        IMAX=8+IDAT(7)*IDAT(8)                                     65800000
        WRITE(ICHR,902)(IDAT(I),I=9,IMAX)                          66000000
      ENDIF                                                        66200000
      GOTO 2000                                                    66400000
C                                                                  66600000
C**** LINEWIDTH SCALE FACTOR                                       66800000
 1023 CONTINUE                                                     67000000
C**** MARKER SIZE SCALE FACTOR                                     67200000
 1027 CONTINUE                                                     67400000
C**** CHARACTER EXPANSION FACTOR                                   67600000
 1031 CONTINUE                                                     67800000
C**** CHARACTER SPACING                                            68000000
 1032 CONTINUE                                                     68200000
C**** READ/WRITE ITEM DATA                                         68400000
      IF (ICNV.EQ.ICNVCM) THEN                                     68600000
```

```
        READ (ICHR,903)RDAT(1)                                          68800000
      ELSE                                                              69000000
        WRITE(ICHR,903)RDAT(1)                                          69200000
      ENDIF                                                             69400000
      GOTO 2000                                                         69600000
C                                                                       69800000
C**** CHARACTER VECTORS                                                 70000000
 1034 CONTINUE                                                          70200000
C**** PATTERN VECTORS                                                   70400000
 1041 CONTINUE                                                          70600000
C**** CLIPPING RECTANGLE                                                70800000
 1061 CONTINUE                                                          71000000
C**** WORKSTATION WINDOW                                                71200000
 1071 CONTINUE                                                          71400000
C**** WORKSTATION VIEWPORT                                              71600000
 1072 CONTINUE                                                          71800000
C**** READ/WRITE ITEM DATA                                              72000000
      IF (ICNV.EQ.ICNVCM) THEN                                          72200000
        READ (ICHR,903)RDAT(1),RDAT(2),RDAT(3),RDAT(4)                  72400000
      ELSE                                                              72600000
        WRITE(ICHR,903)RDAT(1),RDAT(2),RDAT(3),RDAT(4)                  72800000
      ENDIF                                                             73000000
      GOTO 2000                                                         73200000
C                                                                       73400000
C**** PATTERN REFERENCE POINT                                           73600000
 1042 IF (ICNV.EQ.ICNVCM) THEN                                          73800000
        READ (ICHR,903)RDAT(1),RDAT(2)                                  74000000
      ELSE                                                              74200000
        WRITE(ICHR,903)RDAT(1),RDAT(2)                                  74400000
      ENDIF                                                             74600000
      GOTO 2000                                                         74800000
C                                                                       75000000
C**** ASPECT SOURCE FLAGS                                               75200000
 1043 IF (ICNV.EQ.ICNVCM) THEN                                          75400000
        READ (ICHR,902)(IDAT(I),I=1,13)                                 75600000
      ELSE                                                              75800000
        WRITE(ICHR,902)(IDAT(I),I=1,13)                                 76000000
      ENDIF                                                             76200000
      GOTO 2000                                                         76400000
C                                                                       76600000
C**** POLYLINE REPRESENTATION                                           76800000
 1051 CONTINUE                                                          77000000
C**** POLYMARKER REPRESENTATION                                         77200000
 1052 CONTINUE                                                          77400000
C**** READ/WRITE ITEM DATA                                              77600000
      IF (ICNV.EQ.ICNVCM) THEN                                          77800000
        READ (ICHR,902)IDAT(1),IDAT(2)                                  78000000
        READ (ICHR,903)RDAT(3)                                          78200000
        READ (ICHR,902)IDAT(4)                                          78400000
      ELSE                                                              78600000
        WRITE(ICHR,902)IDAT(1),IDAT(2)                                  78800000
        WRITE(ICHR,903)RDAT(3)                                          79000000
        WRITE(ICHR,902)IDAT(4)                                          79200000
      ENDIF                                                             79400000
      GOTO 2000                                                         79600000
C                                                                       79800000
C**** TEXT REPRESENTATION                                               80000000
 1053 IF (ICNV.EQ.ICNVCM) THEN                                          80200000
```

```
        READ (ICHR,902)IDAT(1),IDAT(2),IDAT(3)                          80400000
        READ (ICHR,903)RDAT(4),RDAT(5)                                  80600000
        READ (ICHR,902)IDAT(6)                                          80800000
      ELSE                                                              81000000
        WRITE(ICHR,902)IDAT(1),IDAT(2),IDAT(3)                          81200000
        WRITE(ICHR,903)RDAT(4),RDAT(5)                                  81400000
        WRITE(ICHR,902)IDAT(6)                                          81600000
      ENDIF                                                             81800000
      GOTO 2000                                                         82000000
C                                                                       82200000
C**** FILL AREA REPRESENTATION                                          82400000
 1054 IF (ICNV.EQ.ICNVCM) THEN                                          82600000
        READ (ICHR,902)IDAT(1),IDAT(2),IDAT(3),IDAT(4)                  82800000
      ELSE                                                              83000000
        WRITE(ICHR,902)IDAT(1),IDAT(2),IDAT(3),IDAT(4)                  83200000
      ENDIF                                                             83400000
      GOTO 2000                                                         83600000
C                                                                       83800000
C**** FILL AREA REPRESENTATION                                          84000000
 1055 IF (ICNV.EQ.ICNVCM) THEN                                          84200000
        READ (ICHR,902)IDAT(1),IDAT(2),IDAT(3)                          84400000
        IMAX=3+IDAT(2)*IDAT(3)                                          84600000
        READ (ICHR,902)(IDAT(I),I=4,IMAX)                               84800000
      ELSE                                                              85000000
        WRITE(ICHR,902)IDAT(1),IDAT(2),IDAT(3)                          85200000
        IMAX=3+IDAT(2)*IDAT(3)                                          85400000
        WRITE(ICHR,902)(IDAT(I),I=4,IMAX)                               85600000
      ENDIF                                                             85800000
      GOTO 2000                                                         86000000
C                                                                       86200000
C**** COLOR REPRESENTATION                                              86400000
 1056 IF (ICNV.EQ.ICNVCM) THEN                                          86600000
        READ (ICHR,902)IDAT(1)                                          86800000
        READ (ICHR,903)RDAT(2),RDAT(3),RDAT(4)                          87000000
      ELSE                                                              87200000
        WRITE(ICHR,902)IDAT(1)                                          87400000
        WRITE(ICHR,903)RDAT(2),RDAT(3),RDAT(4)                          87600000
      ENDIF                                                             87800000
      GOTO 2000                                                         88000000
C                                                                       88200000
C**** SEGMENT TRANSFORMATION                                            88400000
 1091 IF (ICNV.EQ.ICNVCM) THEN                                          88600000
        READ (ICHR,902)IDAT(1)                                          88800000
        READ (ICHR,903)(RDAT(I),I=2,7)                                  89000000
      ELSE                                                              89200000
        WRITE(ICHR,902)IDAT(1)                                          89400000
        WRITE(ICHR,903)(RDAT(I),I=2,7)                                  89600000
      ENDIF                                                             89800000
      GOTO 2000                                                         90000000
C                                                                       90200000
C**** SET SEGMENT PRIORITY                                              90400000
 1094 IF (ICNV.EQ.ICNVCM) THEN                                          90600000
        READ (ICHR,902)IDAT(1)                                          90800000
        READ (ICHR,903)RDAT(2)                                          91000000
      ELSE                                                              91200000
        WRITE(ICHR,902)IDAT(1)                                          91400000
        WRITE(ICHR,903)RDAT(2)                                          91600000
      ENDIF                                                             91800000
```

```
      GOTO 2000                                                          92000000
C                                                                        92200000
C**** USER ITEMS                                                         92400000
 1100 IF (ICNV.EQ.ICNVCM) THEN                                           92600000
        READ (ICHR,901)(CDAT(I),I=1,ILEN)                                92800000
      ELSE                                                               93000000
        WRITE(ICHR,901)(CDAT(I),I=1,ILEN)                                93200000
      ENDIF                                                              93400000
      GOTO 2000                                                          93600000
C                                                                        93800000
C                                                                        94000000
C**** WRITE ITEM DATA TO METAFILE (IF APPROPRIATE)                       94200000
 2000 IF (ICNV.EQ.ICNVCM) THEN                                           94400000
        IF (ITYP.LE.MAXTYP) THEN                                         94600000
          CALL GIITM (ITYP,ILEN,MAXREC,DATREC)                           94800000
        ELSE                                                             95000000
          CALL GWITM (IWKID,ITYP,ILEN,MAXREC,DATREC)                     95200000
        ENDIF                                                            95400000
      ENDIF                                                              95600000
C                                                                        95800000
C**** CHECK FOR LAST METAFILE ITEM                                       96000000
      IF (ITYP.NE.0) GOTO 200                                            96200000
C                                                                        96400000
C                                                                        96600000
C     ************************************************************************96800000
C     **  T E R M I N A T I O N                                           **97000000
C     ************************************************************************97200000
C                                                                        97400000
C**** SUCCESSFUL COMPLETION                                              97600000
      WRITE(6,3100)INUM                                                  97800000
 3100 FORMAT(1X,'EXECUTION COMPLETED SUCCESSFULLY, ',I6,                 98000000
     &  ' METAFILE ITEMS PROCESSED.')                                    98200000
C                                                                        98400000
C**** TERMINATE GKS                                                      98600000
 3200 IF (ICNV.EQ.ICNVCM) CALL GDAWK (IWKID)                             98800000
      CALL GCLWK (IWKID)                                                 99000000
      CALL GCLKS                                                         99200000
C                                                                        99400000
 3300 STOP                                                               99600000
      END                                                                99800000
```

Appendix F. GDDM-GKS RCP codes

This appendix contains two lists of the GDDM-GKS RCP codes. The first list is ordered by the name of the function; the second list is in the order of the RCP code.

RCP codes ordered by function name

Call Name	RCP Code (Hex.)	RCP Code (Decimal)	Function
GACTM	38006A00	939551232	Accumulate transformation matrix
GACWK	38000400	939525120	Activate workstation
GASGWK	38003D00	939539712	Associate segment with workstation
GCA	38001000	939528192	Cell array
GCLKS	38000100	939524352	Close GKS
GCLRWK	38000600	939525632	Clear workstation
GCLSG	38003900	939538688	Close segment
GCLWK	38000300	939524864	Close workstation
GCRSG	38003800	939538432	Create segment
GCSGWK	38003E00	939539968	Copy segment to workstation
GDAWK	38000500	939525376	Deactivate workstation
GDSG	38003B00	939539200	Delete segment
GDSGWK	38003C00	939539456	Delete segment from workstation
GECLKS	3800F000	939585536	Emergency close GKS
GERLOG	3800F100	939585792	Error logging
GESC	38000B00	939526912	Escape
GEVTM	38006900	939550976	Evaluate transformation matrix
GFA	38000F00	939527936	Fill area
GFLUSH	38005E00	939548160	Flush device events
GGDP	38001100	939528448	Generalized drawing primitive
GGTCH	38006200	939549184	Get choice
GGTITM	38006600	939550208	Get item type from GKSM
GGTLC	38005F00	939548416	Get locator
GGTPK	38006300	939549440	Get pick
GGTSK	38006000	939548672	Get stroke
GGTSTS	38006400	939549696	Get string
GGTVL	38006100	939548928	Get valuator
GIITM	38006800	939550720	Interpret item
GINCH	38004800	939542528	Initialize choice
GINLC	38004500	939541760	Initialize locator
GINPK	38004900	939542784	Initialize pick
GINSG	38003F00	939540224	Insert segment
GINSK	38004600	939542016	Initialize stroke
GINSTS	38004A00	939543040	Initialize string
GINVL	38004700	939542272	Initialize valuator
GMSGS	38000A00	939526656	Message
GOPKS	38000000	939524096	Open GKS
GOPWK	38000200	939524608	Open workstation
GPL	38000C00	939527168	Polyline
GPM	38000D00	939527424	Polymarker
GPRECS	38006B00	939551488	Pack data record
GQACWK	38008600	939558400	Inquire set member of active workstations
GQASF	3800A100	939565312	Inquire aspect source flags
GQASWK	3800DE00	939580928	Inquire set member of associated workstations

© Copyright IBM Corp. 1984, 1987

Call Name	RCP Code (Hex.)	RCP Code (Decimal)	Function
GQCF	3800D000	939577344	Inquire color facilities
GQCHB	38008D00	939560192	Inquire character base vector
GQCHH	38008A00	939559424	Inquire character height
GQCHS	3800BE00	939572736	Inquire choice device state
GQCHSP	38009C00	939564032	Inquire character spacing
GQCHUP	38008B00	939559680	Inquire character up vector
GQCHW	38008C00	939559936	Inquire character width
GQCHXP	38009B00	939563776	Inquire character expansion factor
GQCLIP	3800A500	939566336	Inquire clipping indicator
GQCNTN	3800A200	939565568	Inquire current normalization transformation number
GQCR	3800B800	939571200	Inquire color representation
GQDCH	3800DB00	939580160	Inquire default choice device data
GQDDS	3800C500	939574528	Inquire default deferral state values
GQDLC	3800D800	939579392	Inquire default locator device data
GQDPK	3800DC00	939580416	Inquire default pick device data
GQDSGA	3800D600	939578880	Inquire dynamic modification of segment attributes
GQDSK	3800D900	939579648	Inquire default stroke device data
GQDSP	3800C300	939574016	Inquire display space size
GQDST	3800DD00	939580672	Inquire default string device data
GQDVL	3800DA00	939579904	Inquire default valuator device data
GQDWKA	3800C400	939574272	Inquire dynamic modification of workstation attributes
GQECI	3800B700	939570944	Inquire list element of color indexes
GQEFAI	3800B300	939569920	Inquire list element of fill area indexes
GQEGDP	3800D200	939577856	Inquire list element of available generalized drawing primitives
GQENTN	3800A300	939565824	Inquire list element of normalization transformation numbers
GQEPAI	3800B500	939570432	Inquire list element of pattern indexes
GQEPLI	3800AC00	939568128	Inquire list element of polyline indexes
GQEPMI	3800AE00	939568640	Inquire list element of polymarker indexes
GQETXI	3800B000	939569152	Inquire list element of text indexes
GQEWK	38008200	939557376	Inquire list element of available workstation types
GQFACI	3800A000	939565056	Inquire fill area color index
GQFAF	3800CC00	939576320	Inquire fill area facilities
GQFAI	38009000	939560960	Inquire fill area index
GQFAIS	38009E00	939564544	Inquire fill area interior style
GQFAR	3800B400	939570176	Inquire fill area representation
GQFASI	38009F00	939564800	Inquire fill area style index
GQGDP	3800D300	939578112	Inquire generalized drawing primitive
GQIQOV	3800E300	939582208	Inquire input queue overflow
GQLCS	3800BB00	939571968	Inquire locator device state
GQLI	3800D700	939579136	Inquire number of available logical input devices
GQLN	38009400	939561984	Inquire linetype
GQLVKS	38008100	939557120	Inquire level of GKS
GQLWK	3800D400	939578368	Inquire maximum length of workstation state tables
GQLWSC	38009500	939562240	Inquire linewidth scale factor
GQMK	38009700	939562752	Inquire marker type
GQMKSC	38009800	939563008	Inquire marker size scale factor
GQMNTN	38008400	939557888	Inquire maximum normalization transformation number
GQNT	3800A400	939566080	Inquire normalization transformation
GQOPS	38008000	939556864	Inquire operating state value
GQOPSG	3800A600	939566592	Inquire name of open segment
GQOPWK	38008500	939558144	Inquire set member of open workstations
GQPA	38009100	939561216	Inquire pattern size
GQPAF	3800CE00	939576832	Inquire pattern facilities
GQPAR	3800B600	939570688	Inquire pattern representation
GQPARF	38009200	939561472	Inquire pattern reference point
GQPCR	3800D100	939577600	Inquire predefined color representation

Call Name	RCP Code (Hex.)	RCP Code (Decimal)	Function
GQPFAR	3800CD00	939576576	Inquire predefined fill area representation
GQPKID	38009300	939561728	Inquire pick identifier
GQPKS	3800BF00	939572992	Inquire pick device state
GQPLCI	38009600	939562496	Inquire polyline color index
GQPLF	3800C600	939574784	Inquire polyline facilities
GQPLI	38008700	939558656	Inquire polyline index
GQPLR	3800AD00	939568384	Inquire polyline representation
GQPMCI	38009900	939563264	Inquire polymarker color index
GQPMF	3800C800	939575296	Inquire polymarker facilities
GQPMI	38008800	939558912	Inquire polymarker index
GQPMR	3800AF00	939568896	Inquire polymarker representation
GQPPAR	3800CF00	939577088	Inquire predefined pattern representation
GQPPLR	3800C700	939575040	Inquire predefined polyline representation
GQPPMR	3800C900	939575552	Inquire predefined polymarker representation
GQPTXR	3800CB00	939576064	Inquire predefined text representation
GQPX	3800E200	939581952	Inquire pixel
GQPXA	3800E100	939581696	Inquire pixel array
GQPXAD	3800E000	939581440	Inquire pixel array dimensions
GQSGA	3800DF00	939581184	Inquire segment attributes
GQSGP	3800D500	939578624	Inquire number of segment priorities supported
GQSGUS	3800A700	939566848	Inquire set member of segment names in use
GQSGWK	3800BA00	939571712	Inquire set member of segment names on workstation
GQSIM	3800A800	939567104	Inquire more simultaneous events
GQSKS	3800BC00	939572224	Inquire stroke device state
GQSTSS	3800C000	939573248	Inquire string device state
GQTXAL	38008F00	939560704	Inquire text alignment
GQTXCI	38009D00	939564288	Inquire text color index
GQTXF	3800CA00	939575808	Inquire text facilities
GQTXFP	38009A00	939563520	Inquire text font and precision
GQTXI	38008900	939559168	Inquire text index
GQTXP	38008E00	939560448	Inquire text path
GQTXR	3800B100	939569408	Inquire text representation
GQTXXS	3800B200	939569664	Inquire text extent
GQVLS	3800BD00	939572480	Inquire valuator device state
GQWKC	3800A900	939567360	Inquire workstation connection and type
GQWKCA	3800C100	939573504	Inquire workstation category
GQWKCL	3800C200	939573760	Inquire workstation classification
GQWKDU	3800AB00	939567872	Inquire workstation deferral and update states
GQWKM	38008300	939557632	Inquire workstation maximum numbers
GQWKS	3800AA00	939567616	Inquire workstation state
GQWKT	3800B900	939571456	Inquire workstation transformation
GRDITM	38006700	939550464	Read item from GKSM
GRENSG	38003A00	939538944	Rename segment
GRQCH	38005400	939545600	Request choice
GRQLC	38005100	939544832	Request locator
GRQPK	38005500	939545856	Request pick
GRQSK	38005200	939545088	Request stroke
GRQSTS	38005600	939546112	Request string
GRQVL	38005300	939545344	Request valuator
GRSGWK	38000700	939525888	Redraw all segments on workstation
GSASF	38002900	939534592	Set aspect source flags
GSCHH	38001F00	939532032	Set character height
GSCHM	38004E00	939544064	Set choice mode
GSCHSP	38001D00	939531520	Set character spacing
GSCHUP	38002000	939532288	Set character up vector
GSCHXP	38001C00	939531264	Set character expansion factor

Call Name	RCP Code (Hex.)	RCP Code (Decimal)	Function
GSCLIP	38003500	939537664	Set clipping indicator
GSCR	38003000	939536384	Set color representation
GSDS	38000900	939526400	Set deferral state
GSDTEC	38004400	939541504	Set detectability
GSELNT	38003400	939537408	Select normalization transformation
GSFACI	38002600	939533824	Set fill area color index
GSFAI	38002300	939533056	Set fill area index
GSFAIS	38002400	939533312	Set fill area interior style
GSFAR	38002E00	939535872	Set fill area representation
GSFASI	38002500	939533568	Set fill area style index
GSHLIT	38004200	939540992	Set highlighting
GSLCM	38004B00	939543296	Set locator mode
GSLN	38001300	939528960	Set linetype
GSLWSC	38001400	939529216	Set linewidth scale factor
GSMCH	38005A00	939547136	Sample choice
GSMK	38001700	939529984	Set marker type
GSMKSC	38001800	939530240	Set marker size scale factor
GSMLC	38005700	939546368	Sample locator
GSMPK	38005B00	939547392	Sample pick
GSMSK	38005800	939546624	Sample stroke
GSMSTS	38005C00	939547648	Sample string
GSMVL	38005900	939546880	Sample valuator
GSPA	38002700	939534080	Set pattern size
GSPAR	38002F00	939536128	Set pattern representation
GSPARF	38002800	939534336	Set pattern reference point
GSPKID	38002A00	939534848	Set pick identifier
GSPKM	38004F00	939544320	Set pick mode
GSPLCI	38001500	939529472	Set polyline color index
GSPLI	38001200	939528704	Set polyline index
GSPLR	38002B00	939535104	Set polyline representation
GSPMCI	38001900	939530496	Set polymarker color index
GSPMI	38001600	939529728	Set polymarker index
GSPMR	38002C00	939535360	Set polymarker representation
GSSGP	38004300	939541248	Set segment priority
GSSGT	38004000	939540480	Set segment transformation
GSSKM	38004C00	939543552	Set stroke mode
GSSTM	38005000	939544576	Set string mode
GSTXAL	38002200	939532800	Set text alignment
GSTXCI	38001E00	939531776	Set text color index
GSTXFP	38001B00	939531008	Set text font and precision
GSTXI	38001A00	939530752	Set text index
GSTXP	38002100	939532544	Set text path
GSTXR	38002D00	939535616	Set text representation
GSVIS	38004100	939540736	Set visibility
GSVLM	38004D00	939543808	Set valuator mode
GSVP	38003200	939536896	Set viewport
GSVPIP	38003300	939537152	Set viewport input priority
GSWKVP	38003700	939538176	Set workstation viewport
GSWKWN	38003600	939537920	Set workstation window
GSWN	38003100	939536640	Set window
GTXS	38000E00	939527680	Text
GURECS	38006C00	939551744	Unpack data record
GUWK	38000800	939526144	Update workstation
GWAIT	38005D00	939547904	Await event
GWITM	38006500	939549952	Write item to GKSM

RCP codes ordered by code value

RCP Code (Hex.)	RCP Code (Decimal)	Call Name	Function
38000000	939524096	GOPKS	Open GKS
38000100	939524352	GCLKS	Close GKS
38000200	939524608	GOPWK	Open workstation
38000300	939524864	GCLWK	Close workstation
38000400	939525120	GACWK	Activate workstation
38000500	939525376	GDAWK	Deactivate workstation
38000600	939525632	GCLRWK	Clear workstation
38000700	939525888	GRSGWK	Redraw all segments on workstation
38000800	939526144	GUWK	Update workstation
38000900	939526400	GSDS	Set deferral state
38000A00	939526656	GMSGS	Message
38000B00	939526912	GESC	Escape
38000C00	939527168	GPL	Polyline
38000D00	939527424	GPM	Polymarker
38000E00	939527680	GTXS	Text
38000F00	939527936	GFA	Fill area
38001000	939528192	GCA	Cell array
38001100	939528448	GGDP	Generalized drawing primitive
38001200	939528704	GSPLI	Set polyline index
38001300	939528960	GSLN	Set linetype
38001400	939529216	GSLWSC	Set linewidth scale factor
38001500	939529472	GSPLCI	Set polyline color index
38001600	939529728	GSPMI	Set polymarker index
38001700	939529984	GSMK	Set marker type
38001800	939530240	GSMKSC	Set marker size scale factor
38001900	939530496	GSPMCI	Set polymarker color index
38001A00	939530752	GSTXI	Set text index
38001B00	939531008	GSTXFP	Set text font and precision
38001C00	939531264	GSCHXP	Set character expansion factor
38001D00	939531520	GSCHSP	Set character spacing
38001E00	939531776	GSTXCI	Set text color index
38001F00	939532032	GSCHH	Set character height
38002000	939532288	GSCHUP	Set character up vector
38002100	939532544	GSTXP	Set text path
38002200	939532800	GSTXAL	Set text alignment
38002300	939533056	GSFAI	Set fill area index
38002400	939533312	GSFAIS	Set fill area interior style
38002500	939533568	GSFASI	Set fill area style index
38002600	939533824	GSFACI	Set fill area color index
38002700	939534080	GSPA	Set pattern size
38002800	939534336	GSPARF	Set pattern reference point
38002900	939534592	GSASF	Set aspect source flags
38002A00	939534848	GSPKID	Set pick identifier
38002B00	939535104	GSPLR	Set polyline representation
38002C00	939535360	GSPMR	Set polymarker representation
38002D00	939535616	GSTXR	Set text representation
38002E00	939535872	GSFAR	Set fill area representation
38002F00	939536128	GSPAR	Set pattern representation
38003000	939536384	GSCR	Set color representation
38003100	939536640	GSWN	Set window
38003200	939536896	GSVP	Set viewport
38003300	939537152	GSVPIP	Set viewport input priority
38003400	939537408	GSELNT	Select normalization transformation

RCP Code (Hex.)	RCP Code (Decimal)	Call Name	Function
38003500	939537664	GSCLIP	Set clipping indicator
38003600	939537920	GSWKWN	Set workstation window
38003700	939538176	GSWKVP	Set workstation viewport
38003800	939538432	GCRSG	Create segment
38003900	939538688	GCLSG	Close segment
38003A00	939538944	GRENSG	Rename segment
38003B00	939539200	GDSG	Delete segment
38003C00	939539456	GDSGWK	Delete segment from workstation
38003D00	939539712	GASGWK	Associate segment with workstation
38003E00	939539968	GCSGWK	Copy segment to workstation
38003F00	939540224	GINSG	Insert segment
38004000	939540480	GSSGT	Set segment transformation
38004100	939540736	GSVIS	Set visibility
38004200	939540992	GSHLIT	Set highlighting
38004300	939541248	GSSGP	Set segment priority
38004400	939541504	GSDTEC	Set detectability
38004500	939541760	GINLC	Initialize locator
38004600	939542016	GINSK	Initialize stroke
38004700	939542272	GINVL	Initialize valuator
38004800	939542528	GINCH	Initialize choice
38004900	939542784	GINPK	Initialize pick
38004A00	939543040	GINSTS	Initialize string
38004B00	939543296	GSLCM	Set locator mode
38004C00	939543552	GSSKM	Set stroke mode
38004D00	939543808	GSVLM	Set valuator mode
38004E00	939544064	GSCHM	Set choice mode
38004F00	939544320	GSPKM	Set pick mode
38005000	939544576	GSSTM	Set string mode
38005100	939544832	GRQLC	Request locator
38005200	939545088	GRQSK	Request stroke
38005300	939545344	GRQVL	Request valuator
38005400	939545600	GRQCH	Request choice
38005500	939545856	GRQPK	Request pick
38005600	939546112	GRQSTS	Request string
38005700	939546368	GSMLC	Sample locator
38005800	939546624	GSMSK	Sample stroke
38005900	939546880	GSMVL	Sample valuator
38005A00	939547136	GSMCH	Sample choice
38005B00	939547392	GSMPK	Sample pick
38005C00	939547648	GSMSTS	Sample string
38005D00	939547904	GWAIT	Await event
38005E00	939548160	GFLUSH	Flush device events
38005F00	939548416	GGTLC	Get locator
38006000	939548672	GGTSK	Get stroke
38006100	939548928	GGTVL	Get valuator
38006200	939549184	GGTCH	Get choice
38006300	939549440	GGTPK	Get pick
38006400	939549696	GGTSTS	Get string
38006500	939549952	GWITM	Write item to GKSM
38006600	939550208	GGTITM	Get item type from GKSM
38006700	939550464	GRDITM	Read item from GKSM
38006800	939550720	GIITM	Interpret item
38006900	939550976	GEVTM	Evaluate transformation matrix
38006A00	939551232	GACTM	Accumulate transformation matrix
38006B00	939551488	GPRECS	Pack data record
38006C00	939551744	GURECS	Unpack data record

RCP Code (Hex.)	RCP Code (Decimal)	Call Name	Function
38008000	939556864	GQOPS	Inquire operating state value
38008100	939557120	GQLVKS	Inquire level of GKS
38008200	939557376	GQEWK	Inquire list element of available workstation types
38008300	939557632	GQWKM	Inquire workstation maximum numbers
38008400	939557888	GQMNTN	Inquire maximum normalization transformation number
38008500	939558144	GQOPWK	Inquire set member of open workstations
38008600	939558400	GQACWK	Inquire set member of active workstations
38008700	939558656	GQPLI	Inquire polyline index
38008800	939558912	GQPMI	Inquire polymarker index
38008900	939559168	GQTXI	Inquire text index
38008A00	939559424	GQCHH	Inquire character height
38008B00	939559680	GQCHUP	Inquire character up vector
38008C00	939559936	GQCHW	Inquire character width
38008D00	939560192	GQCHB	Inquire character base vector
38008E00	939560448	GQTXP	Inquire text path
38008F00	939560704	GQTXAL	Inquire text alignment
38009000	939560960	GQFAI	Inquire fill area index
38009100	939561216	GQPA	Inquire pattern size
38009200	939561472	GQPARF	Inquire pattern reference point
38009300	939561728	GQPKID	Inquire pick identifier
38009400	939561984	GQLN	Inquire linetype
38009500	939562240	GQLWSC	Inquire linewidth scale factor
38009600	939562496	GQPLCI	Inquire polyline color index
38009700	939562752	GQMK	Inquire marker type
38009800	939563008	GQMKSC	Inquire marker size scale factor
38009900	939563264	GQPMCI	Inquire polymarker color index
38009A00	939563520	GQTXFP	Inquire text font and precision
38009B00	939563776	GQCHXP	Inquire character expansion factor
38009C00	939564032	GQCHSP	Inquire character spacing
38009D00	939564288	GQTXCI	Inquire text color index
38009E00	939564544	GQFAIS	Inquire fill area interior style
38009F00	939564800	GQFASI	Inquire fill area style index
3800A000	939565056	GQFACI	Inquire fill area color index
3800A100	939565312	GQASF	Inquire aspect source flags
3800A200	939565568	GQCNTN	Inquire current normalization transformation number
3800A300	939565824	GQENTN	Inquire list element of normalization transformation numbers
3800A400	939566080	GQNT	Inquire normalization transformation
3800A500	939566336	GQCLIP	Inquire clipping indicator
3800A600	939566592	GQOPSG	Inquire name of open segment
3800A700	939566848	GQSGUS	Inquire set member of segment names in use
3800A800	939567104	GQSIM	Inquire more simultaneous events
3800A900	939567360	GQWKC	Inquire workstation connection and type
3800AA00	939567616	GQWKS	Inquire workstation state
3800AB00	939567872	GQWKDU	Inquire workstation deferral and update states
3800AC00	939568128	GQEPLI	Inquire list element of polyline indexes
3800AD00	939568384	GQPLR	Inquire polyline representation
3800AE00	939568640	GQEPMI	Inquire list element of polymarker indexes
3800AF00	939568896	GQPMR	Inquire polymarker representation
3800B000	939569152	GQETXI	Inquire list element of text indexes
3800B100	939569408	GQTXR	Inquire text representation
3800B200	939569664	GQTXXS	Inquire text extent
3800B300	939569920	GQEFAI	Inquire list element of fill area indexes
3800B400	939570176	GQFAR	Inquire fill area representation
3800B500	939570432	GQEPAI	Inquire list element of pattern indexes
3800B600	939570688	GQPAR	Inquire pattern representation
3800B700	939570944	GQECI	Inquire list element of color indexes

RCP Code (Hex.)	RCP Code (Decimal)	Call Name	Function
3800B800	939571200	GQCR	Inquire color representation
3800B900	939571456	GQWKT	Inquire workstation transformation
3800BA00	939571712	GQSGWK	Inquire set member of segment names on workstation
3800BB00	939571968	GQLCS	Inquire locator device state
3800BC00	939572224	GQSKS	Inquire stroke device state
3800BD00	939572480	GQVLS	Inquire valuator device state
3800BE00	939572736	GQCHS	Inquire choice device state
3800BF00	939572992	GQPKS	Inquire pick device state
3800C000	939573248	GQSTSS	Inquire string device state
3800C100	939573504	GQWKCA	Inquire workstation category
3800C200	939573760	GQWKCL	Inquire workstation classification
3800C300	939574016	GQDSP	Inquire display space size
3800C400	939574272	GQDWKA	Inquire dynamic modification of workstation attributes
3800C500	939574528	GQDDS	Inquire default deferral state values
3800C600	939574784	GQPLF	Inquire polyline facilities
3800C700	939575040	GQPPLR	Inquire predefined polyline representation
3800C800	939575296	GQPMF	Inquire polymarker facilities
3800C900	939575552	GQPPMR	Inquire predefined polymarker representation
3800CA00	939575808	GQTXF	Inquire text facilities
3800CB00	939576064	GQPTXR	Inquire predefined text representation
3800CC00	939576320	GQFAF	Inquire fill area facilities
3800CD00	939576576	GQPFAR	Inquire predefined fill area representation
3800CE00	939576832	GQPAF	Inquire pattern facilities
3800CF00	939577088	GQPPAR	Inquire predefined pattern representation
3800D000	939577344	GQCF	Inquire color facilities
3800D100	939577600	GQPCR	Inquire predefined color representation
3800D200	939577856	GQEGDP	Inquire list element of available generalized drawing primitives
3800D300	939578112	GQGDP	Inquire generalized drawing primitive
3800D400	939578368	GQLWK	Inquire maximum length of workstation state tables
3800D500	939578624	GQSGP	Inquire number of segment priorities supported
3800D600	939578880	GQDSGA	Inquire dynamic modification of segment attributes
3800D700	939579136	GQLI	Inquire number of available logical input devices
3800D800	939579392	GQDLC	Inquire default locator device data
3800D900	939579648	GQDSK	Inquire default stroke device data
3800DA00	939579904	GQDVL	Inquire default valuator device data
3800DB00	939580160	GQDCH	Inquire default choice device data
3800DC00	939580416	GQDPK	Inquire default pick device data
3800DD00	939580672	GQDST	Inquire default string device data
3800DE00	939580928	GQASWK	Inquire set member of associated workstations
3800DF00	939581184	GQSGA	Inquire segment attributes
3800E000	939581440	GQPXAD	Inquire pixel array dimensions
3800E100	939581696	GQPXA	Inquire pixel array
3800E200	939581952	GQPX	Inquire pixel
3800E300	939582208	GQIQOV	Inquire input queue overflow
3800F000	939585536	GECLKS	Emergency close GKS
3800F100	939585792	GERLOG	Error logging

Appendix G. GDDM-GKS APL codes

This appendix contains two lists of the GDDM-GKS APL codes. The first list is ordered by the function name; the second list is in the order of the APL codes.

APL codes ordered by function name

Call Name	APL Code	Description
GACTM	1418	Accumulate transformation matrix
GACWK	1307	Activate workstation
GASGWK	1362	Associate segment with workstation
GCA	1316	Cell array
GCLKS	1301	Close GKS
GCLRWK	1309	Clear workstation
GCLSG	1358	Close segment
GCLWK	1303	Close workstation
GCRSG	1357	Create segment
GCSGWK	1363	Copy segment to workstation
GDAWK	1308	Deactivate workstation
GDSG	1360	Delete segment
GDSGWK	1361	Delete segment from workstation
GECLKS	1304	Emergency close GKS
GERLOG	1306	Error logging
GESC	1491	Escape
GEVTM	1416	Evaluate transformation matrix
GFA	1315	Fill area
GFLUSH	1392	Flush device events
GGDP	1493	Generalized drawing primitive
GGTCH	1398	Get choice
GGTITM	1410	Get item type from GKSM
GGTLC	1393	Get locator
GGTPK	1399	Get pick
GGTSK	1395	Get stroke
GGTSTS	1401	Get string
GGTVL	1396	Get valuator
GIITM	1412	Interpret item
GINCH	1496	Initialize choice
GINLC	1370	Initialize locator
GINPK	1497	Initialize pick
GINSG	1364	Insert segment
GINSK	1494	Initialize stroke
GINSTS	1498	Initialize string
GINVL	1495	Initialize valuator
GMSGS	1490	Message
GOPKS	1300	Open GKS
GOPWK	1302	Open workstation
GPL	1313	Polyline
GPM	1314	Polymarker
GPRECS	1513	Pack data record
GQACWK	1422	Inquire set member of active workstations
GQASF	1460	Inquire aspect source flags
GQASWK	1429	Inquire set member of associated workstations
GQCF	1415	Inquire color facilities

© Copyright IBM Corp. 1984, 1987

Call Name	APL Code	Description
GQCHB	1439	Inquire character base vector
GQCHH	1431	Inquire character height
GQCHS	1487	Inquire choice device state
GQCHSP	1455	Inquire character spacing
GQCHUP	1433	Inquire character up vector
GQCHW	1438	Inquire character width
GQCHXP	1454	Inquire character expansion factor
GQCLIP	1464	Inquire clipping indicator
GQCNTN	1461	Inquire current normalization transformation number
GQCR	1482	Inquire color representation
GQDCH	1503	Inquire default choice device data
GQDDS	1394	Inquire default deferral state values
GQDLC	1512	Inquire default locator device data
GQDPK	1505	Inquire default pick device data
GQDSGA	1425	Inquire dynamic modification of segment attributes
GQDSK	1501	Inquire default stroke device data
GQDSP	1388	Inquire display space size
GQDST	1504	Inquire default string device data
GQDVL	1502	Inquire default valuator device data
GQDWKA	1390	Inquire dynamic modification of workstation attributes
GQECI	1481	Inquire list element of color indexes
GQEFAI	1477	Inquire list element of fill area indexes
GQEGDP	1420	Inquire list element of available generalized drawing primitives
GQENTN	1462	Inquire list element of normalization transformation numbers
GQEPAI	1479	Inquire list element of pattern indexes
GQEPLI	1470	Inquire list element of polyline indexes
GQEPMI	1472	Inquire list element of polymarker indexes
GQETXI	1474	Inquire list element of text indexes
GQEWK	1508	Inquire list element of available workstation types
GQFACI	1459	Inquire fill area color index
GQFAF	1408	Inquire fill area facilities
GQFAI	1443	Inquire fill area index
GQFAIS	1457	Inquire fill area interior style
GQFAR	1478	Inquire fill area representation
GQFASI	1458	Inquire fill area style index
GQGDP	1421	Inquire generalized drawing primitive
GQIQOV	1437	Inquire input queue overflow
GQLCS	1500	Inquire locator device state
GQLI	1426	Inquire number of available logical input devices
GQLN	1447	Inquire linetype
GQLVKS	1499	Inquire level of GKS
GQLWK	1423	Inquire maximum length of workstation state tables
GQLWSC	1448	Inquire linewidth scale factor
GQMK	1450	Inquire marker type
GQMKSC	1451	Inquire marker size scale factor
GQMNTN	1506	Inquire maximum normalization transformation number
GQNT	1463	Inquire normalization transformation
GQOPS	1305	Inquire operating state value
GQOPSG	1465	Inquire name of open segment
GQOPWK	1419	Inquire set member of open workstations
GQPA	1444	Inquire pattern size
GQPAF	1413	Inquire pattern facilities
GQPAR	1480	Inquire pattern representation
GQPARF	1445	Inquire pattern reference point
GQPCR	1417	Inquire predefined color representation
GQPFAR	1409	Inquire predefined fill area representation

Call Name	APL Code	Description
GQPKID	1446	Inquire pick identifier
GQPKS	1488	Inquire pick device state
GQPLCI	1449	Inquire polyline color index
GQPLF	1397	Inquire polyline facilities
GQPLI	1427	Inquire polyline index
GQPLR	1471	Inquire polyline representation
GQPMCI	1452	Inquire polymarker color index
GQPMF	1402	Inquire polymarker facilities
GQPMI	1428	Inquire polymarker index
GQPMR	1473	Inquire polymarker representation
GQPPAR	1414	Inquire predefined pattern representation
GQPPLR	1400	Inquire predefined polyline representation
GQPPMR	1404	Inquire predefined polymarker representation
GQPTXR	1406	Inquire predefined text representation
GQPX	1436	Inquire pixel
GQPXA	1435	Inquire pixel array
GQPXAD	1434	Inquire pixel array dimensions
GQSGA	1432	Inquire segment attributes
GQSGP	1424	Inquire number of segment priorities supported
GQSGUS	1509	Inquire set member of segment names in use
GQSGWK	1484	Inquire set member of segment names on workstation
GQSIM	1466	Inquire more simultaneous events
GQSKS	1485	Inquire stroke device state
GQSTSS	1489	Inquire string device state
GQTXAL	1442	Inquire text alignment
GQTXCI	1456	Inquire text color index
GQTXF	1405	Inquire text facilities
GQTXFP	1453	Inquire text font and precision
GQTXI	1430	Inquire text index
GQTXP	1511	Inquire text path
GQTXR	1475	Inquire text representation
GQTXXS	1476	Inquire text extent
GQVLS	1486	Inquire valuator device state
GQWKC	1467	Inquire workstation connection and type
GQWKCA	1386	Inquire workstation category
GQWKCL	1387	Inquire workstation classification
GQWKDU	1469	Inquire workstation deferral and update states
GQWKM	1507	Inquire workstation maximum numbers
GQWKS	1468	Inquire workstation state
GQWKT	1483	Inquire workstation transformation
GRDITM	1411	Read item from GKSM
GRENSG	1359	Rename segment
GRQCH	1380	Request choice
GRQLC	1377	Request locator
GRQPK	1381	Request pick
GRQSK	1378	Request stroke
GRQSTS	1382	Request string
GRQVL	1379	Request valuator
GRSGWK	1310	Redraw all segments on workstation
GSASF	1341	Set aspect source flags
GSCHH	1331	Set character height
GSCHM	1374	Set choice mode
GSCHSP	1329	Set character spacing
GSCHUP	1332	Set character up vector
GSCHXP	1328	Set character expansion factor
GSCLIP	1354	Set clipping indicator

Call Name	APL Code	Description
GSCR	1349	Set color representation
GSDS	1312	Set deferral state
GSDTEC	1369	Set detectability
GSELNT	1353	Select normalization transformation
GSFACI	1338	Set fill area color index
GSFAI	1335	Set fill area index
GSFAIS	1336	Set fill area interior style
GSFAR	1347	Set fill area representation
GSFASI	1337	Set fill area style index
GSHLIT	1367	Set highlighting
GSLCM	1371	Set locator mode
GSLN	1318	Set linetype
GSLWSC	1319	Set linewidth scale factor
GSMCH	1385	Sample choice
GSMK	1322	Set marker type
GSMKSC	1323	Set marker size scale factor
GSMLC	1510	Sample locator
GSMPK	1403	Sample pick
GSMSK	1383	Sample stroke
GSMSTS	1389	Sample string
GSMVL	1384	Sample valuator
GSPA	1339	Set pattern size
GSPAR	1348	Set pattern representation
GSPARF	1340	Set pattern reference point
GSPKID	1342	Set pick identifier
GSPKM	1375	Set pick mode
GSPLCI	1320	Set polyline color index
GSPLI	1317	Set polyline index
GSPLR	1344	Set polyline representation
GSPMCI	1324	Set polymarker color index
GSPMI	1321	Set polymarker index
GSPMR	1345	Set polymarker representation
GSSGP	1368	Set segment priority
GSSGT	1365	Set segment transformation
GSSKM	1372	Set stroke mode
GSSTM	1376	Set string mode
GSTXAL	1334	Set text alignment
GSTXCI	1330	Set text color index
GSTXFP	1326	Set text font and precision
GSTXI	1325	Set text index
GSTXP	1333	Set text path
GSTXR	1346	Set text representation
GSVIS	1366	Set visibility
GSVLM	1373	Set valuator mode
GSVP	1351	Set viewport
GSVPIP	1352	Set viewport input priority
GSWKVP	1356	Set workstation viewport
GSWKWN	1355	Set workstation window
GSWN	1350	Set window
GTXS	1492	Text
GURECS	1514	Unpack data record
GUWK	1311	Update workstation
GWAIT	1391	Await event
GWITM	1407	Write item to GKSM

APL codes ordered by code value

APL Code	Call Name	Description
1300	GOPKS	Open GKS
1301	GCLKS	Close GKS
1302	GOPWK	Open workstation
1303	GCLWK	Close workstation
1304	GECLKS	Emergency close GKS
1305	GQOPS	Inquire operating state value
1306	GERLOG	Error logging
1307	GACWK	Activate workstation
1308	GDAWK	Deactivate workstation
1309	GCLRWK	Clear workstation
1310	GRSGWK	Redraw all segments on workstation
1311	GUWK	Update workstation
1312	GSDS	Set deferral state
1313	GPL	Polyline
1314	GPM	Polymarker
1315	GFA	Fill area
1316	GCA	Cell array
1317	GSPLI	Set polyline index
1318	GSLN	Set linetype
1319	GSLWSC	Set linewidth scale factor
1320	GSPLCI	Set polyline color index
1321	GSPMI	Set polymarker index
1322	GSMK	Set marker type
1323	GSMKSC	Set marker size scale factor
1324	GSPMCI	Set polymarker color index
1325	GSTXI	Set text index
1326	GSTXFP	Set text font and precision
1328	GSCHXP	Set character expansion factor
1329	GSCHSP	Set character spacing
1330	GSTXCI	Set text color index
1331	GSCHH	Set character height
1332	GSCHUP	Set character up vector
1333	GSTXP	Set text path
1334	GSTXAL	Set text alignment
1335	GSFAI	Set fill area index
1336	GSFAIS	Set fill area interior style
1337	GSFASI	Set fill area style index
1338	GSFACI	Set fill area color index
1339	GSPA	Set pattern size
1340	GSPARF	Set pattern reference point
1341	GSASF	Set aspect source flags
1342	GSPKID	Set pick identifier
1344	GSPLR	Set polyline representation
1345	GSPMR	Set polymarker representation
1346	GSTXR	Set text representation
1347	GSFAR	Set fill area representation
1348	GSPAR	Set pattern representation
1349	GSCR	Set color representation
1350	GSWN	Set window
1351	GSVP	Set viewport
1352	GSVPIP	Set viewport input priority
1353	GSELNT	Select normalization transformation
1354	GSCLIP	Set clipping indicator

APL Code	Call Name	Description
1355	GSWKWN	Set workstation window
1356	GSWKVP	Set workstation viewport
1357	GCRSG	Create segment
1358	GCLSG	Close segment
1359	GRENSG	Rename segment
1360	GDSG	Delete segment
1361	GDSGWK	Delete segment from workstation
1362	GASGWK	Associate segment with workstation
1363	GCSGWK	Copy segment to workstation
1364	GINSG	Insert segment
1365	GSSGT	Set segment transformation
1366	GSVIS	Set visibility
1367	GSHLIT	Set highlighting
1368	GSSGP	Set segment priority
1369	GSDTEC	Set detectability
1370	GINLC	Initialize locator
1371	GSLCM	Set locator mode
1372	GSSKM	Set stroke mode
1373	GSVLM	Set valuator mode
1374	GSCHM	Set choice mode
1375	GSPKM	Set pick mode
1376	GSSTM	Set string mode
1377	GRQLC	Request locator
1378	GRQSK	Request stroke
1379	GRQVL	Request valuator
1380	GRQCH	Request choice
1381	GRQPK	Request pick
1382	GRQSTS	Request string
1383	GSMSK	Sample stroke
1384	GSMVL	Sample valuator
1385	GSMCH	Sample choice
1386	GQWKCA	Inquire workstation category
1387	GQWKCL	Inquire workstation classification
1388	GQDSP	Inquire display space size
1389	GSMSTS	Sample string
1390	GQDWKA	Inquire dynamic modification of workstation attributes
1391	GWAIT	Await event
1392	GFLUSH	Flush device events
1393	GGTLC	Get locator
1394	GQDDS	Inquire default deferral state values
1395	GGTSK	Get stroke
1396	GGTVL	Get valuator
1397	GQPLF	Inquire polyline facilities
1398	GGTCH	Get choice
1399	GGTPK	Get pick
1400	GQPPLR	Inquire predefined polyline representation
1401	GGTSTS	Get string
1402	GQPMF	Inquire polymarker facilities
1403	GSMPK	Sample pick
1404	GQPPMR	Inquire predefined polymarker representation
1405	GQTXF	Inquire text facilities
1406	GQPTXR	Inquire predefined text representation
1407	GWITM	Write item to GKSM
1408	GQFAF	Inquire fill area facilities
1409	GQPFAR	Inquire predefined fill area representation
1410	GGTITM	Get item type from GKSM

APL Code	Call Name	Description
1411	GRDITM	Read item from GKSM
1412	GIITM	Interpret item
1413	GQPAF	Inquire pattern facilities
1414	GQPPAR	Inquire predefined pattern representation
1415	GQCF	Inquire color facilities
1416	GEVTM	Evaluate transformation matrix
1417	GQPCR	Inquire predefined color representation
1418	GACTM	Accumulate transformation matrix
1419	GQOPWK	Inquire set member of open workstations
1420	GQEGDP	Inquire list element of available generalized drawing primitives
1421	GQGDP	Inquire generalized drawing primitive
1422	GQACWK	Inquire set member of active workstations
1423	GQLWK	Inquire maximum length of workstation state tables
1424	GQSGP	Inquire number of segment priorities supported
1425	GQDSGA	Inquire dynamic modification of segment attributes
1426	GQLI	Inquire number of available logical input devices
1427	GQPLI	Inquire polyline index
1428	GQPMI	Inquire polymarker index
1429	GQASWK	Inquire set member of associated workstations
1430	GQTXI	Inquire text index
1431	GQCHH	Inquire character height
1432	GQSGA	Inquire segment attributes
1433	GQCHUP	Inquire character up vector
1434	GQPXAD	Inquire pixel array dimensions
1435	GQPXA	Inquire pixel array
1436	GQPX	Inquire pixel
1437	GQIQOV	Inquire input queue overflow
1438	GQCHW	Inquire character width
1439	GQCHB	Inquire character base vector
1442	GQTXAL	Inquire text alignment
1443	GQFAI	Inquire fill area index
1444	GQPA	Inquire pattern size
1445	GQPARF	Inquire pattern reference point
1446	GQPKID	Inquire pick identifier
1447	GQLN	Inquire linetype
1448	GQLWSC	Inquire linewidth scale factor
1449	GQPLCI	Inquire polyline color index
1450	GQMK	Inquire marker type
1451	GQMKSC	Inquire marker size scale factor
1452	GQPMCI	Inquire polymarker color index
1453	GQTXFP	Inquire text font and precision
1454	GQCHXP	Inquire character expansion factor
1455	GQCHSP	Inquire character spacing
1456	GQTXCI	Inquire text color index
1457	GQFAIS	Inquire fill area interior style
1458	GQFASI	Inquire fill area style index
1459	GQFACI	Inquire fill area color index
1460	GQASF	Inquire aspect source flags
1461	GQCNTN	Inquire current normalization transformation number
1462	GQENTN	Inquire list element of normalization transformation numbers
1463	GQNT	Inquire normalization transformation
1464	GQCLIP	Inquire clipping indicator
1465	GQOPSG	Inquire name of open segment
1466	GQSIM	Inquire more simultaneous events
1467	GQWKC	Inquire workstation connection and type
1468	GQWKS	Inquire workstation state

APL Code	Call Name	Description
1469	GQWKDU	Inquire workstation deferral and update states
1470	GQEPLI	Inquire list element of polyline indexes
1471	GQPLR	Inquire polyline representation
1472	GQEPMI	Inquire list element of polymarker indexes
1473	GQPMR	Inquire polymarker representation
1474	GQETXI	Inquire list element of text indexes
1475	GQTXR	Inquire text representation
1476	GQTXXS	Inquire text extent
1477	GQEFAI	Inquire list element of fill area indexes
1478	GQFAR	Inquire fill area representation
1479	GQEPAI	Inquire list element of pattern indexes
1480	GQPAR	Inquire pattern representation
1481	GQECI	Inquire list element of color indexes
1482	GQCR	Inquire color representation
1483	GQWKT	Inquire workstation transformation
1484	GQSGWK	Inquire set member of segment names on workstation
1485	GQSKS	Inquire stroke device state
1486	GQVLS	Inquire valuator device state
1487	GQCHS	Inquire choice device state
1488	GQPKS	Inquire pick device state
1489	GQSTSS	Inquire string device state
1490	GMSGS	Message
1491	GESC	Escape
1492	GTXS	Text
1493	GGDP	Generalized drawing primitive
1494	GINSK	Initialize stroke
1495	GINVL	Initialize valuator
1496	GINCH	Initialize choice
1497	GINPK	Initialize pick
1498	GINSTS	Initialize string
1499	GQLVKS	Inquire level of GKS
1500	GQLCS	Inquire locator device state
1501	GQDSK	Inquire default stroke device data
1502	GQDVL	Inquire default valuator device data
1503	GQDCH	Inquire default choice device data
1504	GQDST	Inquire default string device data
1505	GQDPK	Inquire default pick device data
1506	GQMNTN	Inquire maximum normalization transformation number
1507	GQWKM	Inquire workstation maximum numbers
1508	GQEWK	Inquire list element of available workstation types
1509	GQSGUS	Inquire set member of segment names in use
1510	GSMLC	Sample locator
1511	GQTXP	Inquire text path
1512	GQDLC	Inquire default locator device data
1513	GPRECS	Pack data record
1514	GURECS	Unpack data record

Glossary

This glossary defines technical terms used in GDDM documentation. If you do not find the term you are looking for, refer to the index of the appropriate GDDM manual or view the *IBM Dictionary of Computing*, located on the Internet at:

`http:\\www.networking.ibm.com/nsg/nsgmain.htm`

A

API. Application program interface.

APL. One of the programming languages supported by GDDM.

application program interface (API). The formally defined programming-language interface between an IBM system control program or licensed program and its user.

aspect ratio. The width-to-height ratio of an area, symbol, or shape.

assembler. One of the programming languages supported by GDDM.

attributes. Characteristics or properties that can be controlled, usually to obtain a required appearance; for example, the color of a line. See also **graphics attributes**.

B

background color. Black on a display, white on a printer. The initial color of the display medium. Contrast with **neutral color**.

BASIC. One of the programming languages supported by GDDM.

C

character. A letter, digit, or other symbol.

character box. In GDDM, the rectangle or (for sheared characters) the parallelogram boundaries that govern the size, orientation, spacing, and italicizing of individual symbols or characters to be shown on a display screen or printer page.

The box width, height, and if required, shear, are specified in world coordinates and may be program-controlled.

choice device. A logical input device that enables the application program to identify keys pressed by the terminal operator.

clipping. In computer graphics, removing parts of a display image that lie outside a viewport.

CMS. Conversational Monitor System. A time-sharing subsystem that runs under VM/SP.

COBOL. One of the programming languages supported by GDDM.

current position. In GDDM, the end of the previously drawn primitive. Unless a “move” is performed, this position will also be the start of the next primitive.

cursor. A physical indicator that may be moved around a display screen.

D

data set. The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

DBCS. Double-byte character set.

default value. A value chosen by GDDM when no value is explicitly specified by the user. For example, the default line type is a solid line.

device echo. A visual identification of the position of the graphics cursor. The form of the device echo is defined by the application program.

device family. In GDDM, a device classification that governs the general way in which I/O will be processed. For example:

- Family 1: 3270 display or printer
- Family 2: queued printer
- Family 3: system printer (alphanumerics only)
- Family 4: high-resolution printer.

device token. In GDDM, an 8-byte code giving entry to a table of pre-established device hardware characteristics that are required when the device is opened (initialized).

display device. Any output unit that gives a visual representation of data. For example, a screen or printer. More commonly, the term is used to mean a screen as opposed to a printer.

© Copyright IBM Corp. 1984, 1987

double-byte character set (DBCS). A set of characters in which each character occupies two byte positions in internal storage and in display buffers. Used for oriental languages.

dual characters. In GDDM, characters that each occupy two bytes in internal storage and in display buffers. They are used to display Kanji or Hangeul symbols.

E

echo. In interactive graphics, the visible form of the locator or other logical input device.

edit. To enter, modify, or delete data.

extended data stream. For 3179, 3278, 3279, and 3287 devices, input/output data formatted and encoded in support of color, programmed symbols, and extended highlighting. These features extend the 3270 data-stream architecture.

extended highlighting. The emphasizing of a displayed character's appearance by blinking, underscore, or reverse video.

external defaults. GDDM-supplied values that users can change to suit their own needs.

F

field. An area on the screen or the printed or plotted page.

font. A particular style of typeface (for example, Gothic English). In GDDM, a font may exist as a programmed symbol set.

FORTRAN. One of the programming languages supported by GDDM.

four-button cursor. A hand-held device, with cross-hair sight, for indicating positions on the surface of a tablet. Synonymous with **puck**.

G

GDDM. Graphical Data Display Manager.

GKS. Graphical Kernel System.

GKSM. GKS metafile. See "metafile."

graphics. A picture defined in terms of graphics primitives and graphics attributes.

graphics attributes. In GDDM, comprise color selection, color mix, line type, line width, graphics text attributes, marker symbol, and shading pattern definition.

graphics cursor. A physical indicator that can be moved (often with a joystick, mouse, or stylus) to any position on the screen.

graphics data format (GDF). A picture definition in an encoded order format used internally by GDDM and, optionally, providing the user with a lower-level programming interface than the GDDM API.

graphics primitive. A single item of drawn graphics, such as a line, arc, or graphics text string. See also **graphics segment**.

graphics segment. A group of graphics primitives (lines, arcs, and text) that have a common window and a common viewport and associated attributes. Graphics segments allow a group of primitives to be subject to various operations. See also **graphics primitive**.

H

Hangeul. A character set of symbols used in Korean ideographic alphabets.

I

integer. A whole number (for example, -2, 3, 457).

interactive graphics. In GDDM, those graphics that can be moved or manipulated by a user at a terminal.

J

JCL. Job Control Language.

K

Kanji. A character set of symbols used in Japanese ideographic alphabets.

L

line attributes. In GDDM, color, line type, and line width.

link edit. To create a loadable computer program by means of a linkage editor.

load module. A program unit that is suitable for loading into main storage for execution; it is usually the output of a linkage editor.

locator. A logical input device used to indicate a position on the screen. Its physical form may be the

alphanumeric cursor or a graphics cursor moved by a joystick.

logical input device. A concept that allows application programs to be written in a device independent manner. The logical input devices to which the program refers may be subsequently associated with different physical parts of a terminal, depending on which device is used at run-time.

M

marker. In GDDM, a symbol centered on a point. Line graphs and polar charts may use markers to indicate the plotted points.

menu. A displayed list of logically grouped functions from which the operator may make a selection.

metafile. GKS metafile (GKSM). A mechanism for retaining and transporting graphic data and control information. This information contains a device-independent description of a picture.

mixed character string. A string containing, for example, a mixture of Latin (one-byte) and Kanji (two-byte) characters.

mouse. A hand-held device (the IBM 5277 Mouse) that is moved around a locator pad to position the graphics cursor on the screen.

N

national language (NL) feature. The translations of the GDDM messages into a variety of languages other than English.

neutral color. White on a display, black on a printer. Contrast with **background color**.

nickname. In GDDM, a quick and easy means of referring to a device, the characteristics and identity of which have been predefined.

O

object code. Output from a compiler or assembler that is in itself executable machine code or is suitable for processing to produce executable machine code.

P

PDS. In OS/TSO, a partitioned data set.

pel. Synonym for **pixel**.

pick. The action of the operator selecting part of a graphics display by placing the graphics cursor over it.

pick device. A logical input device that allows the application to determine which part of the picture was selected (or picked) by the operator.

picture element. Synonym for **pixel**.

picture interchange format (PIF) file. In graphics systems, the type of file, containing picture data, that can be transferred between GDDM and a 3270-PC/G or 3270-PC/GX work station.

PIF. Picture interchange format (PIF) file.

pixel. The smallest area of a display screen capable of being addressed and switched between visible and invisible states. Synonymous with **pel**, and **picture element**.

PL/I. One of the programming languages supported by GDDM.

plotter. An output device that uses pens to draw its output on paper or transparency foils.

polyline. A sequence of adjoining lines.

primitive. See **graphics primitive**.

primitive attribute. A specifiable characteristic of a graphics primitive. See **graphics attributes**.

program library. (1) A collection of available computer programs and routines. (2) An organized collection of computer programs. (3) Synonym for **partitioned data set**.

programmed symbols (PS). Dot patterns loaded by GDDM into the PS stores of an output device.

PS. Programmed symbols.

puck. Synonym for **four-button cursor**.

R

raster device. A device with a display area consisting of dots.

reentrant. The attribute of a program or routine that allows the same copy of the program or routine to be used concurrently by two or more tasks.

resolution. In graphics and image processing, the number of pixels per unit of measure (inch or meter).

reverse video. A form of alphanumeric highlighting for a character, field, or cursor, in which its color is exchanged with that of its background. For example, changing a red character on a black background to a black character on a red background.

S

scrolling. In computer graphics, moving a display image vertically or horizontally in a manner such that new data appears at one edge as existing data disappears at the opposite edge.

segment. See **graphics segment**.

segment attributes. Attributes that apply to the segment as an entity, rather than to the individual primitives within the segment. For example, the visibility, transformability, or detectability of a segment.

segment priority. The order in which segments will be drawn, also the order in which they will be detected.

segment transform. The means to rotate, scale, and reposition segments without re-creating them.

shear. The action of tilting graphics text so that each character leans to the left or right while retaining a horizontal baseline.

SPI. System programmer interface.

string device. A logical input device that enables an application program to process character data entered by the terminal operator.

stroke device. A logical input device that enables an application program to process a sequence of x,y coordinate data entered by the terminal operator.

stylus. A pen-like pointer for indicating positions on the surface of a tablet.

T

tablet. (1) A locator device with a flat surface and a mechanism that converts indicated positions on the surface into coordinate data. (2) The IBM 5083 Tablet Model 2, which, with a four-button cursor or stylus, allows positions on the screen to be addressed and the graphics cursor to be moved without use of the keyboard.

tag. In interactive graphics, an identifier associated with one or more primitives that is returned to the program if such primitives are subsequently picked.

terminal. A device, usually equipped with a keyboard and a display unit, capable of sending and receiving information over a link. See also **display terminal**.

text. Characters or symbols sent to the device. GDDM provides alphanumeric text and graphics text.

transform. (1) The action of modifying a picture for display; for example, by scaling, rotating, shearing, or displacing. (2) The object that performs or defines such a modification; also referred to as a **transformation**.

TSO. Time sharing option. A subsystem of OS/VS under which GDDM can be used.

V

viewport. A subdivision of the picture space, most often used when two separate pictures are to be displayed together.

VM/SP CMS. IBM Virtual Machine/System Product Conversational Monitor System. A system under which GDDM can be used.

W

window. (1) In GDDM, a defined section of world coordinates. The window can be regarded as a set of coordinates that are overlaid on the viewport. (2) In GDDM, the “graphics window” is the set of coordinates used for defining the primitives that make up a graphics display. By default, both x and y coordinates run from 0 through 100. (3) In GDDM, an “operator window” is an independent rectangular subdivision of the screen. Several can exist at the same time, and each can receive output from, and send input to, either a separate GDDM program or a separate function of a single GDDM program. (4) In GDDM, the “page window” defines which part of a deep or wide page should currently be displayed.

workstation. (1) A display screen together with attachments such as a local copy device or a tablet. (2) In GDDM-GKS, an abstract device providing an interface to physical devices and **metafiles**.

world coordinates. The user application-oriented coordinates used for drawing graphics. See also **window**.

Index

Numerics

A

B

C

© Copyright IBM Corp. 1984, 1987

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC33-0334-00