GDDM-GKS
Programming Guide and Reference
## Contents

**Preface** .......................................................... xi
Latest GDDM information ........................................... xi
GDDM publications ................................................... xii
Books from related libraries ..................................... xiii
Book structure ......................................................... xiv

**Chapter 1. Introduction to GKS** ........................................... 1
The ROOM program ..................................................... 1

**Chapter 2. Using GKS** ....................................................... 5
Concepts of GKS ......................................................... 5
  Workstations .......................................................... 5
  Graphical output ..................................................... 6
  Coordinate systems .................................................. 6
  Segments ............................................................... 6
  Graphical input ...................................................... 6
  GKS metafiles ......................................................... 7
  GKS operating states ................................................. 7
  GKS data structures ................................................ 8
GKS functions ........................................................... 8
  GKS function types .................................................. 8
  Control functions .................................................... 9
  Output and attribute functions .................................... 9
  Transformation functions .......................................... 16
  Segment functions ................................................... 20
  Input functions ....................................................... 24
  Metafile functions ................................................. 25
  Utility functions ..................................................... 25
  Inquiry functions ..................................................... 26
  Error handling ....................................................... 26
Designing an interactive graphics program ......................... 27
  Argument conventions .............................................. 27
  The ROOM program .................................................. 27

**Chapter 3. Introduction to GDDM-GKS** .................................... 39
Relation to GDDM Base ............................................... 39
Relation to the GKS standard ......................................... 39
  Differences between GDDM-GKS and the ISO and ANS GKS standards 39
FORTRAN binding ....................................................... 40
Programming languages other than FORTRAN ....................... 40
Non-GKS external names ............................................. 41

**Chapter 4. Using GDDM-GKS** .............................................. 43
Invocation of GDDM-GKS functions ................................... 43
  The nonreentrant interface ....................................... 44
  The reentrant interface ............................................ 44
  The system programmer interface ................................ 45
FORTRAN language considerations .................................. 45
  VS FORTRAN ........................................................ 45
  FORTRAN IV .......................................................... 46
GQDCCH .......................................................... 157
GQDDDS .......................................................... 158
GQDLHC .......................................................... 160
GQDPK ............................................................ 161
GQDSGA .......................................................... 163
GQDSK ............................................................ 164
GQDSP ............................................................ 166
GQDST ............................................................ 167
GQDVSL .......................................................... 168
GQDWSK .......................................................... 170
GQECI .............................................................. 172
GQFSAI ............................................................ 173
GQEGDPM ......................................................... 174
GQENTN ........................................................... 175
GQEPAL ............................................................. 176
GQEPIL ............................................................. 177
GQEPML ............................................................ 178
GQETXI ............................................................. 179
GQEWK ............................................................. 181
GQFACI ............................................................. 182
GQFAF .............................................................. 182
GQFAI .............................................................. 184
GQFAIS ............................................................ 185
GQFAR .............................................................. 186
GQFASI ............................................................. 187
GQGDPM ........................................................... 188
GQIQOV ............................................................. 189
GQLCS .............................................................. 190
GQLI ................................................................. 192
GQLN ................................................................. 193
GQLVKS ............................................................ 194
GQLWK .............................................................. 195
GQLWSC ........................................................... 196
GQMK ................................................................. 197
GQMKSC ............................................................ 198
GQMNTN ........................................................... 199
GQNT ................................................................. 200
GQOPS ............................................................... 201
GQOPSG ............................................................ 201
GQOPWK ............................................................ 202
GQPA ................................................................. 203
GQPAF ............................................................... 204
GQPAR ............................................................... 205
GQPARF ............................................................. 206
GQPCR ............................................................... 207
GQPFAR ............................................................. 208
GQPKID ............................................................. 209
GQPKS ............................................................... 210
GQPLCI .............................................................. 212
GQPLF ............................................................... 213
GQPLI ............................................................... 214
GQPLR ............................................................... 215
GQPMCI ............................................................. 216
GQPMF ............................................................... 217
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSELNT</td>
<td>287</td>
</tr>
<tr>
<td>GSFACI</td>
<td>288</td>
</tr>
<tr>
<td>GSFAI</td>
<td>289</td>
</tr>
<tr>
<td>GSFAIS</td>
<td>290</td>
</tr>
<tr>
<td>GSFAI</td>
<td>291</td>
</tr>
<tr>
<td>GSFAIS</td>
<td>292</td>
</tr>
<tr>
<td>GSFAR</td>
<td>293</td>
</tr>
<tr>
<td>GSFASI</td>
<td>294</td>
</tr>
<tr>
<td>GSFLIT</td>
<td>295</td>
</tr>
<tr>
<td>GSLCM</td>
<td>296</td>
</tr>
<tr>
<td>GSLN</td>
<td>297</td>
</tr>
<tr>
<td>GSLWSC</td>
<td>298</td>
</tr>
<tr>
<td>GSMCH</td>
<td>299</td>
</tr>
<tr>
<td>GSKM</td>
<td>300</td>
</tr>
<tr>
<td>GSKMSC</td>
<td>301</td>
</tr>
<tr>
<td>GSPMLC</td>
<td>302</td>
</tr>
<tr>
<td>GSPMLC</td>
<td>303</td>
</tr>
<tr>
<td>GSSGP</td>
<td>304</td>
</tr>
<tr>
<td>GSSGT</td>
<td>305</td>
</tr>
<tr>
<td>GSSKM</td>
<td>306</td>
</tr>
<tr>
<td>GSSKM</td>
<td>307</td>
</tr>
<tr>
<td>GSSTM</td>
<td>308</td>
</tr>
<tr>
<td>GSTXAL</td>
<td>309</td>
</tr>
<tr>
<td>GSTXCI</td>
<td>310</td>
</tr>
<tr>
<td>GSTXFP</td>
<td>311</td>
</tr>
<tr>
<td>GSTXI</td>
<td>312</td>
</tr>
<tr>
<td>GSTXP</td>
<td>313</td>
</tr>
<tr>
<td>GSTXR</td>
<td>314</td>
</tr>
<tr>
<td>GSVIS</td>
<td>315</td>
</tr>
<tr>
<td>GSVP</td>
<td>316</td>
</tr>
<tr>
<td>GSVPIP</td>
<td>317</td>
</tr>
<tr>
<td>GSWKVP</td>
<td>318</td>
</tr>
<tr>
<td>GSWKVN</td>
<td>319</td>
</tr>
<tr>
<td>GSWN</td>
<td>320</td>
</tr>
<tr>
<td>GTX</td>
<td>321</td>
</tr>
<tr>
<td>GTXS</td>
<td>322</td>
</tr>
<tr>
<td>GUREC</td>
<td>323</td>
</tr>
<tr>
<td>GURECS</td>
<td>324</td>
</tr>
<tr>
<td>GUWK</td>
<td>325</td>
</tr>
<tr>
<td>GWAIT</td>
<td>326</td>
</tr>
<tr>
<td>GWITM</td>
<td>327</td>
</tr>
</tbody>
</table>
Appendix A. GKS data structures ........................................... 359
  Operating state ......................................................... 360
  GKS description table .................................................. 360
  GKS state list .......................................................... 361
  Workstation state list .................................................. 364
  Workstation description tables ....................................... 368
  Segment state list ...................................................... 377
  GKS error state list .................................................... 377

Appendix B. GDDM-GKS enumeration types ............................... 379

Appendix C. Metafile structure ........................................... 383
  Metafile item types .................................................... 383
  Metafile item structure ................................................ 386
    Control items ......................................................... 387
    Items for output primitives ....................................... 388
    Items for output primitive attributes ........................... 389
    Items for workstation attributes ................................ 392
    Items for transformations .......................................... 394
    Items for segment manipulation .................................... 394
    Items for segment attributes ..................................... 395
    User items ................................................................ 396

Appendix D. ROOM program source code ................................ 397

Appendix E. Example programs ............................................. 405

Appendix F. GDDM-GKS RCP codes ........................................ 419
  RCP codes ordered by function name ................................ 419
  RCP codes ordered by code value ..................................... 423

Appendix G. GDDM-GKS APL codes ........................................ 427
  APL codes ordered by function name ................................ 427
  APL codes ordered by code value ..................................... 431

Glossary ............................................................................ 435

Index .................................................................................. 439
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
<table>
<thead>
<tr>
<th>GDDM publications</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>GDDM Base</td>
<td></td>
</tr>
<tr>
<td>GDDM Base Application Programming Guide</td>
<td>SC33-0867</td>
</tr>
<tr>
<td>GDDM Base Application Programming Reference</td>
<td>SC33-0868</td>
</tr>
<tr>
<td>GDDM Diagnosis</td>
<td>SC33-0870</td>
</tr>
<tr>
<td>GDDM General Information</td>
<td>GC33-0866</td>
</tr>
<tr>
<td>GDDM/MVS Program Directory</td>
<td>GC33-1801</td>
</tr>
<tr>
<td>GDDM/VM Program Directory</td>
<td>GC33-1802</td>
</tr>
<tr>
<td>GDDM/VSE Program Directory</td>
<td>GC33-1803</td>
</tr>
<tr>
<td>GDDM Messages</td>
<td>SC33-0869</td>
</tr>
<tr>
<td>GDDM Series Licensed Program Specifications</td>
<td>GC33-0876</td>
</tr>
<tr>
<td>GDDM System Customization and Administration</td>
<td>SC33-0871</td>
</tr>
<tr>
<td>GDDM User's Guide</td>
<td>SC33-0875</td>
</tr>
<tr>
<td>GDDM Using the Image Symbol Editor</td>
<td>SC33-0920</td>
</tr>
<tr>
<td>GDDM-GKS</td>
<td></td>
</tr>
<tr>
<td>GDDM-GKS Programming Guide and Reference</td>
<td>SC33-0334</td>
</tr>
<tr>
<td>GDDM-IMD</td>
<td></td>
</tr>
<tr>
<td>GDDM Interactive Map Definition</td>
<td>SC33-0338</td>
</tr>
<tr>
<td>GDDM-IVU</td>
<td></td>
</tr>
<tr>
<td>GDDM Image View Utility</td>
<td>SC33-0479</td>
</tr>
<tr>
<td>GDDM-PGF</td>
<td></td>
</tr>
<tr>
<td>GDDM-PGF Application Programming Guide</td>
<td>SC33-0913</td>
</tr>
<tr>
<td>GDDM-PGF Programming Reference</td>
<td>SC33-0333</td>
</tr>
<tr>
<td>GDDM-PGF Interactive Chart Utility</td>
<td>SC33-0328</td>
</tr>
<tr>
<td>GDDM-PGF Vector Symbol Editor</td>
<td>SC33-0330</td>
</tr>
<tr>
<td>GDDM-PGF OPS User's Guide</td>
<td>SC33-1776</td>
</tr>
</tbody>
</table>

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:


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


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:


The following manuals may be useful:


Book structure

Introduction to GKS ... pages 1 through 3
introduces you to the Graphical Kernel System (GKS).

Using GKS ... pages 5 through 38
gives tutorial information on GKS concepts and functions, and shows you how
a program is built by calling GKS functions.

Introduction to GDDM-GKS ... pages 39 through 41
provides a general description of GDDM-GKS and its relation to the GDDM
family of products.

Using GDDM-GKS ... pages 43 through 62
describes how to invoke GDDM-GKS functions using various programming
languages and under various subsystems.

GDDM-GKS functions ... pages 63 through 358
lists the GDDM-GKS functions, with their syntax and parameters.

Appendixes ... pages 359 through 434
GKS data structures
GDDM-GKS enumeration types
Metafile structure
ROOM program source code
Example programs
GDDM-GKS RCP codes
GDDM-GKS APL codes

Glossary ... pages 435 through 438

Index ... pages 439 through end
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 ADMJRROOM
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.
Introduction to GKS
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.
Using GKS

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, metatile 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.

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

This sequence produces the following output:

[Diagram of fill area]

**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, 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:
Using GKS

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, =zerodot.=zerodot, 1=zerodot=zerodot.=zerodot, =zerodot.=zerodot, 1=zerodot=zerodot.=zerodot)
SET VIEWPORT (1, =zerodot.=zerodot, 1.=zerodot, =zerodot.=zerodot, 1.=zerodot)
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 $(0.0,1.0) \times (0.0,1.0)$. The default workstation viewport is the full display surface; only the part that will allow the aspect ratio to be preserved is used.
Using GKS

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 \((0.0,1.0)\times(0.0,1.0)\). 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, dceunits, xdevice, ydevice, xraster, yraster)
if xdevice > ydevice
    then scale = xdevice
    else scale = ydevice
xndc = xdevice/scale
ynhc = ydevice/scale
SET VIEWPORT (2, 0.0, xndc, 0.0, yndc)
SET WORKSTATION WINDOW (wkid, 0.0, xdevice, 0.0, ydevice)
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:

![Diagram showing critical clipping rectangle and subsequent changes in normalization transformation](image)

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.
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.

\[
\text{EVALUATE TRANSFORMATION MATRIX (5.0, 5.0, 15.0, 15.0, 0.0, 1.0, 1.0, 1, Output Matrix)}
\]

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.

\[
\text{ACCUMULATE TRANSFORMATION MATRIX (Input Matrix, 20.0, 20.0, 0.0, 0.0, 0.0, 2.0, 2.0, 1, Output Matrix)}
\]

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.

\[
\text{ACCUMULATE TRANSFORMATION MATRIX (Input Matrix, 20.0, 20.0, 0.0, 0.0, 0.0, 0.7853982, 1.0, 1.0, 1, Output Matrix)}
\]

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.
Chapter 2. Using GKS
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 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 (GINS).

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,
  dunits, 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**

```
Diagram of the program

Initialization

Choice input

Choice #1
Pick input
Create Segment

Choice #2
Locator input
Transform segment

Choice #3
Valuator input
Rotate segment

Choice #4
Pick input
Delete Segment

Choice #5

Termination

Exit
```

**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.
Using GKS

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.
Using GKS

```
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 (=zerodot.=zerodot, =zerodot.=zerodot,
                              xcor-furnwidth, ycor-furnheight, =zerodot.=zerodot, 1.=zerodot,
                              1.=zerodot, =zerodot, 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:
Using GKS

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:

![Screen capture showing valuator input](image)

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:

![Screen diagram]

**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.

```plaintext
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.

```plaintext
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.
Introduction to GDDM-GKS

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.
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 fffff (parameter, ... )
```

where `fffff` 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 fffff (AAB, parameter, ... )
```

where `fffff` 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:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Length</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>8</td>
<td>AAB</td>
<td>Application anchor block</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>AABFC</td>
<td>GDDM feedback code</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>AABSC</td>
<td>GDDM severity code</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>AABEC</td>
<td>GDDM error code</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>AABAP</td>
<td>GDDM anchor pointer</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>0</th>
<th>Normal</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Warning</td>
</tr>
<tr>
<td>8</td>
<td>Error</td>
</tr>
<tr>
<td>12</td>
<td>Severe error</td>
</tr>
<tr>
<td>16</td>
<td>Unrecoverable errors</td>
</tr>
</tbody>
</table>

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
Using GDDM-GKS

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
- **GTX** Text
- **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:

- **GMSG** 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
- **GQSTS** Inquire string device state
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:

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

or, when using the reentrant interface:

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

or, when using the system programmer interface:

```plaintext
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:

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

For example,

```plaintext
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 alphanumerical 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.
Using GDDM-GKS

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:

```plaintext
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:
  ```plaintext
  ADMMDFT GKSWS=(L5379CST)
  ```

- Locally-attached 3279 displays; include the following statement:
  ```plaintext
  ADMMDFT GKSWS=(L79A3)
  ```

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

```plaintext
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 workstations 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=(=c513),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:
Using GDDM-GKS

- 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 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

• 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.
Using GDDM-GKS

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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Declaration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute codes</td>
<td>INTEGER*4</td>
</tr>
<tr>
<td>Control codes</td>
<td>FIXED BINARY(31)</td>
</tr>
<tr>
<td>Option codes</td>
<td>PICTURE S9(8) COMPUTATIONAL or equivalent</td>
</tr>
<tr>
<td>Identification numbers</td>
<td>F-constant</td>
</tr>
</tbody>
</table>

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 Viewport coordinates
Line end-points Window coordinates
Color table values

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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Function</th>
<th>Name</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accumulate transformation matrix</td>
<td>GACTM</td>
<td>106</td>
<td>78</td>
</tr>
<tr>
<td>Activate workstation</td>
<td>GACWK</td>
<td>4</td>
<td>80</td>
</tr>
<tr>
<td>Associate segment with workstation</td>
<td>GASGWK</td>
<td>61</td>
<td>81</td>
</tr>
<tr>
<td>Await event</td>
<td>GWAIT</td>
<td>93</td>
<td>356</td>
</tr>
<tr>
<td>Cell array</td>
<td>GCA</td>
<td>16</td>
<td>82</td>
</tr>
<tr>
<td>Clear workstation</td>
<td>GCLRK</td>
<td>6</td>
<td>85</td>
</tr>
<tr>
<td>Close GKS</td>
<td>GCLKS</td>
<td>1</td>
<td>85</td>
</tr>
<tr>
<td>Close segment</td>
<td>GCLSG</td>
<td>57</td>
<td>87</td>
</tr>
<tr>
<td>Close workstation</td>
<td>GCLWK</td>
<td>3</td>
<td>87</td>
</tr>
<tr>
<td>Copy segment to workstation</td>
<td>GCSDK</td>
<td>62</td>
<td>90</td>
</tr>
<tr>
<td>Create segment</td>
<td>GCRSG</td>
<td>56</td>
<td>89</td>
</tr>
<tr>
<td>Deactivate workstation</td>
<td>GDAWK</td>
<td>5</td>
<td>91</td>
</tr>
<tr>
<td>Delete segment</td>
<td>GDSG</td>
<td>59</td>
<td>92</td>
</tr>
<tr>
<td>Delete segment from workstation</td>
<td>GDSDK</td>
<td>60</td>
<td>93</td>
</tr>
<tr>
<td>Emergency close GKS</td>
<td>GECLKS</td>
<td></td>
<td>94</td>
</tr>
<tr>
<td>Error handling</td>
<td>GERHND</td>
<td></td>
<td>95</td>
</tr>
<tr>
<td>Error logging</td>
<td>GERLOG</td>
<td></td>
<td>96</td>
</tr>
<tr>
<td>Escape</td>
<td>GESC</td>
<td>11</td>
<td>97</td>
</tr>
<tr>
<td>Evaluate transformation matrix</td>
<td>GEVTM</td>
<td>105</td>
<td>98</td>
</tr>
<tr>
<td>Fill area</td>
<td>GFA</td>
<td>15</td>
<td>100</td>
</tr>
<tr>
<td>Flush device events</td>
<td>GFLUSH</td>
<td>94</td>
<td>102</td>
</tr>
<tr>
<td>Generalized drawing primitive</td>
<td>GGDP</td>
<td>17</td>
<td>103</td>
</tr>
<tr>
<td>Get choice</td>
<td>GTCH</td>
<td>98</td>
<td>105</td>
</tr>
<tr>
<td>Get item type from GKSM</td>
<td>GTITM</td>
<td>102</td>
<td>106</td>
</tr>
<tr>
<td>Get locator</td>
<td>GTLC</td>
<td>95</td>
<td>107</td>
</tr>
<tr>
<td>Get pick</td>
<td>GTPK</td>
<td>99</td>
<td>108</td>
</tr>
<tr>
<td>Get string (FORTRAN only)</td>
<td>GSTD</td>
<td>100</td>
<td>110</td>
</tr>
<tr>
<td>Get string</td>
<td>GTS</td>
<td>100</td>
<td>111</td>
</tr>
<tr>
<td>Get stroke</td>
<td>GTSK</td>
<td>96</td>
<td>109</td>
</tr>
<tr>
<td>Get valuator</td>
<td>GTVL</td>
<td>97</td>
<td>112</td>
</tr>
<tr>
<td>Initialize choice</td>
<td>GINCH</td>
<td>72</td>
<td>114</td>
</tr>
<tr>
<td>Initialize locator</td>
<td>GINLC</td>
<td>69</td>
<td>116</td>
</tr>
<tr>
<td>Initialize pick</td>
<td>GINPK</td>
<td>73</td>
<td>120</td>
</tr>
<tr>
<td>Initialize string (FORTRAN only)</td>
<td>GINST</td>
<td>74</td>
<td>127</td>
</tr>
<tr>
<td>Initialize string</td>
<td>GINSTS</td>
<td>74</td>
<td>129</td>
</tr>
<tr>
<td>Initialize stroke</td>
<td>GINSK</td>
<td>70</td>
<td>123</td>
</tr>
<tr>
<td>Initialize valuator</td>
<td>GINVL</td>
<td>71</td>
<td>132</td>
</tr>
<tr>
<td>Inquire aspect source flags</td>
<td>GQASF</td>
<td></td>
<td>144</td>
</tr>
<tr>
<td>Inquire character base vector</td>
<td>GQCHB</td>
<td></td>
<td>147</td>
</tr>
<tr>
<td>Inquire character expansion factor</td>
<td>GQCHXP</td>
<td></td>
<td>153</td>
</tr>
<tr>
<td>Inquire character height</td>
<td>GQCHH</td>
<td></td>
<td>148</td>
</tr>
<tr>
<td>Inquire character spacing</td>
<td>GQCHSP</td>
<td></td>
<td>150</td>
</tr>
<tr>
<td>Inquire character up vector</td>
<td>GQCHUP</td>
<td></td>
<td>151</td>
</tr>
<tr>
<td>Inquire character width</td>
<td>GQCHW</td>
<td></td>
<td>152</td>
</tr>
<tr>
<td>Function</td>
<td>Name</td>
<td>Number</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>----------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Inquire choice device state</td>
<td>GQCHS</td>
<td></td>
<td>149</td>
</tr>
<tr>
<td>Inquire clipping indicator</td>
<td>GQCLIP</td>
<td></td>
<td>154</td>
</tr>
<tr>
<td>Inquire color facilities</td>
<td>GQCF</td>
<td></td>
<td>146</td>
</tr>
<tr>
<td>Inquire color representation</td>
<td>GQCR</td>
<td></td>
<td>155</td>
</tr>
<tr>
<td>Inquire current normalization transformation number</td>
<td>GQCNTN</td>
<td></td>
<td>154</td>
</tr>
<tr>
<td>Inquire default choice device data</td>
<td>GQDCH</td>
<td></td>
<td>157</td>
</tr>
<tr>
<td>Inquire default deferral state values</td>
<td>GQDDS</td>
<td></td>
<td>158</td>
</tr>
<tr>
<td>Inquire default locator device data</td>
<td>GQDLC</td>
<td></td>
<td>160</td>
</tr>
<tr>
<td>Inquire default pick device data</td>
<td>GQDPK</td>
<td></td>
<td>161</td>
</tr>
<tr>
<td>Inquire default string device data</td>
<td>GQDST</td>
<td></td>
<td>167</td>
</tr>
<tr>
<td>Inquire default stroke device data</td>
<td>GQDSK</td>
<td></td>
<td>164</td>
</tr>
<tr>
<td>Inquire default valuator device data</td>
<td>GQDVL</td>
<td></td>
<td>168</td>
</tr>
<tr>
<td>Inquire display space size</td>
<td>GQDSP</td>
<td></td>
<td>166</td>
</tr>
<tr>
<td>Inquire dynamic modification of segment attributes</td>
<td>GQDSGA</td>
<td></td>
<td>163</td>
</tr>
<tr>
<td>Inquire dynamic modification of workstation attributes</td>
<td>GQDWKA</td>
<td></td>
<td>170</td>
</tr>
<tr>
<td>Inquire fill area color index</td>
<td>GQFAIC</td>
<td></td>
<td>182</td>
</tr>
<tr>
<td>Inquire fill area facilities</td>
<td>GQFAF</td>
<td></td>
<td>182</td>
</tr>
<tr>
<td>Inquire fill area index</td>
<td>GQFAI</td>
<td></td>
<td>184</td>
</tr>
<tr>
<td>Inquire fill area interior style</td>
<td>GQFAIS</td>
<td></td>
<td>185</td>
</tr>
<tr>
<td>Inquire fill area representation</td>
<td>GQFAR</td>
<td></td>
<td>186</td>
</tr>
<tr>
<td>Inquire fill area style index</td>
<td>GQFASI</td>
<td></td>
<td>187</td>
</tr>
<tr>
<td>Inquire generalized drawing primitive</td>
<td>GQGDP</td>
<td></td>
<td>188</td>
</tr>
<tr>
<td>Inquire input queue overflow</td>
<td>GQIQOV</td>
<td></td>
<td>189</td>
</tr>
<tr>
<td>Inquire level of GKS</td>
<td>GQLVKS</td>
<td></td>
<td>194</td>
</tr>
<tr>
<td>Inquire linetype</td>
<td>GQLN</td>
<td></td>
<td>193</td>
</tr>
<tr>
<td>Inquire linewidth scale factor</td>
<td>GQLWSC</td>
<td></td>
<td>196</td>
</tr>
<tr>
<td>Inquire list element of available generalized drawing primitives</td>
<td>GQEGDP</td>
<td></td>
<td>174</td>
</tr>
<tr>
<td>Inquire list element of available workstation types</td>
<td>GQEWK</td>
<td></td>
<td>181</td>
</tr>
<tr>
<td>Inquire list element of color indexes</td>
<td>GQECI</td>
<td></td>
<td>172</td>
</tr>
<tr>
<td>Inquire list element of fill area indexes</td>
<td>GQEFAl</td>
<td></td>
<td>173</td>
</tr>
<tr>
<td>Inquire list element of normalization transformation numbers</td>
<td>GQENTN</td>
<td></td>
<td>175</td>
</tr>
<tr>
<td>Inquire list element of pattern indexes</td>
<td>GQEPAI</td>
<td></td>
<td>176</td>
</tr>
<tr>
<td>Inquire list element of polyline indexes</td>
<td>GQEPLI</td>
<td></td>
<td>177</td>
</tr>
<tr>
<td>Inquire list element of polymarker indexes</td>
<td>GQEPMI</td>
<td></td>
<td>178</td>
</tr>
<tr>
<td>Inquire list element of text indexes</td>
<td>GQETXI</td>
<td></td>
<td>179</td>
</tr>
<tr>
<td>Inquire locator device state</td>
<td>GQLCS</td>
<td></td>
<td>190</td>
</tr>
<tr>
<td>Inquire marker size scale factor</td>
<td>GQMKSC</td>
<td></td>
<td>198</td>
</tr>
<tr>
<td>Inquire marker type</td>
<td>GQMK</td>
<td></td>
<td>197</td>
</tr>
<tr>
<td>Inquire maximum length of workstation state tables</td>
<td>GQLWKM</td>
<td></td>
<td>195</td>
</tr>
<tr>
<td>Inquire maximum normalization transformation number</td>
<td>GQMNNTN</td>
<td></td>
<td>199</td>
</tr>
<tr>
<td>Inquire more simultaneous events</td>
<td>GQSIM</td>
<td></td>
<td>234</td>
</tr>
<tr>
<td>Inquire name of open segment</td>
<td>GQOPSG</td>
<td></td>
<td>201</td>
</tr>
<tr>
<td>Inquire normalization transformation</td>
<td>GQNT</td>
<td></td>
<td>200</td>
</tr>
<tr>
<td>Inquire number of available logical input devices</td>
<td>GQLI</td>
<td></td>
<td>192</td>
</tr>
<tr>
<td>Inquire number of segment priorities supported</td>
<td>GQSGP</td>
<td></td>
<td>230</td>
</tr>
<tr>
<td>Inquire operating state value</td>
<td>GQOPS</td>
<td></td>
<td>201</td>
</tr>
<tr>
<td>Inquire pattern facilities</td>
<td>GQPAF</td>
<td></td>
<td>204</td>
</tr>
<tr>
<td>Inquire pattern reference point</td>
<td>GQPARF</td>
<td></td>
<td>206</td>
</tr>
<tr>
<td>Inquire pattern representation</td>
<td>GQPAR</td>
<td></td>
<td>205</td>
</tr>
<tr>
<td>Inquire pattern size</td>
<td>GQPA</td>
<td></td>
<td>203</td>
</tr>
<tr>
<td>Inquire pick device state</td>
<td>GQPKS</td>
<td></td>
<td>210</td>
</tr>
<tr>
<td>Inquire pick identifier</td>
<td>GQPKID</td>
<td></td>
<td>209</td>
</tr>
<tr>
<td>Inquire pixel</td>
<td>GQPX</td>
<td></td>
<td>225</td>
</tr>
<tr>
<td>Inquire pixel array</td>
<td>GQPXA</td>
<td></td>
<td>227</td>
</tr>
<tr>
<td>Function</td>
<td>Name</td>
<td>Number</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Inquire pixel array dimensions</td>
<td>GQPXAD</td>
<td>228</td>
<td></td>
</tr>
<tr>
<td>Inquire polyline color index</td>
<td>GQPLCI</td>
<td>212</td>
<td></td>
</tr>
<tr>
<td>Inquire polyline facilities</td>
<td>GQPLF</td>
<td>213</td>
<td></td>
</tr>
<tr>
<td>Inquire polyline index</td>
<td>GQPLI</td>
<td>214</td>
<td></td>
</tr>
<tr>
<td>Inquire polyline representation</td>
<td>GQPLR</td>
<td>215</td>
<td></td>
</tr>
<tr>
<td>Inquire polymarker color index</td>
<td>GQPMCI</td>
<td>216</td>
<td></td>
</tr>
<tr>
<td>Inquire polymarker facilities</td>
<td>GQPMF</td>
<td>217</td>
<td></td>
</tr>
<tr>
<td>Inquire polymarker index</td>
<td>GQPMI</td>
<td>219</td>
<td></td>
</tr>
<tr>
<td>Inquire polymarker representation</td>
<td>GQPMR</td>
<td>219</td>
<td></td>
</tr>
<tr>
<td>Inquire predefined color representation</td>
<td>GQPCR</td>
<td>207</td>
<td></td>
</tr>
<tr>
<td>Inquire predefined fill area representation</td>
<td>GQPFAR</td>
<td>208</td>
<td></td>
</tr>
<tr>
<td>Inquire predefined pattern representation</td>
<td>GQPPAR</td>
<td>221</td>
<td></td>
</tr>
<tr>
<td>Inquire predefined polyline representation</td>
<td>GQPPPLR</td>
<td>222</td>
<td></td>
</tr>
<tr>
<td>Inquire predefined polymarker representation</td>
<td>GQPPMR</td>
<td>233</td>
<td></td>
</tr>
<tr>
<td>Inquire predefined text representation</td>
<td>GQPTXR</td>
<td>224</td>
<td></td>
</tr>
<tr>
<td>Inquire segment attributes</td>
<td>GQSGA</td>
<td>229</td>
<td></td>
</tr>
<tr>
<td>Inquire set member of active workstations</td>
<td>GQACWK</td>
<td>143</td>
<td></td>
</tr>
<tr>
<td>Inquire set member of associated workstations</td>
<td>GQASHK</td>
<td>145</td>
<td></td>
</tr>
<tr>
<td>Inquire set member of open workstations</td>
<td>GQOPWK</td>
<td>202</td>
<td></td>
</tr>
<tr>
<td>Inquire set member of segment names in use</td>
<td>GQSGUS</td>
<td>232</td>
<td></td>
</tr>
<tr>
<td>Inquire set member of segment names on workstation</td>
<td>GQSGWK</td>
<td>233</td>
<td></td>
</tr>
<tr>
<td>Inquire string device state (FORTRAN only)</td>
<td>GQSTS</td>
<td>236</td>
<td></td>
</tr>
<tr>
<td>Inquire string device state</td>
<td>GQSTSS</td>
<td>238</td>
<td></td>
</tr>
<tr>
<td>Inquire stroke device state</td>
<td>GQSKS</td>
<td>235</td>
<td></td>
</tr>
<tr>
<td>Inquire text alignment</td>
<td>GQTXAL</td>
<td>240</td>
<td></td>
</tr>
<tr>
<td>Inquire text color index</td>
<td>GQTXCI</td>
<td>241</td>
<td></td>
</tr>
<tr>
<td>Inquire text extent (VS FORTRAN only)</td>
<td>GQTXE</td>
<td>247</td>
<td></td>
</tr>
<tr>
<td>Inquire text extent</td>
<td>GQTXS</td>
<td>250</td>
<td></td>
</tr>
<tr>
<td>Inquire text facilities</td>
<td>GQTXF</td>
<td>241</td>
<td></td>
</tr>
<tr>
<td>Inquire text font and precision</td>
<td>GQTXFP</td>
<td>243</td>
<td></td>
</tr>
<tr>
<td>Inquire text index</td>
<td>GQTXI</td>
<td>244</td>
<td></td>
</tr>
<tr>
<td>Inquire text path</td>
<td>GQTXP</td>
<td>245</td>
<td></td>
</tr>
<tr>
<td>Inquire text representation</td>
<td>GQTXR</td>
<td>246</td>
<td></td>
</tr>
<tr>
<td>Inquire valuator device state</td>
<td>GQVLS</td>
<td>252</td>
<td></td>
</tr>
<tr>
<td>Inquire workstation category</td>
<td>GQWKCA</td>
<td>255</td>
<td></td>
</tr>
<tr>
<td>Inquire workstation classification</td>
<td>GQWKCL</td>
<td>256</td>
<td></td>
</tr>
<tr>
<td>Inquire workstation connection and type</td>
<td>GQWKC</td>
<td>254</td>
<td></td>
</tr>
<tr>
<td>Inquire workstation deferral and update states</td>
<td>GQWKDU</td>
<td>257</td>
<td></td>
</tr>
<tr>
<td>Inquire workstation maximum numbers</td>
<td>GQWKM</td>
<td>258</td>
<td></td>
</tr>
<tr>
<td>Inquire workstation state</td>
<td>GQWKS</td>
<td>259</td>
<td></td>
</tr>
<tr>
<td>Inquire workstation transformation</td>
<td>GQWKT</td>
<td>260</td>
<td></td>
</tr>
<tr>
<td>Insert segment</td>
<td>GINSN</td>
<td>63</td>
<td>122</td>
</tr>
<tr>
<td>Interpret item</td>
<td>GIITM</td>
<td>104</td>
<td>112</td>
</tr>
<tr>
<td>Message (VS FORTRAN only)</td>
<td>GMSG</td>
<td>10</td>
<td>133</td>
</tr>
<tr>
<td>Message</td>
<td>GMSGS</td>
<td>10</td>
<td>134</td>
</tr>
<tr>
<td>Open GKS</td>
<td>GOPKS</td>
<td>0</td>
<td>135</td>
</tr>
<tr>
<td>Open workstation</td>
<td>GOPWK</td>
<td>2</td>
<td>136</td>
</tr>
<tr>
<td>Pack data record (FORTRAN only)</td>
<td>GPREC</td>
<td>107</td>
<td>140</td>
</tr>
<tr>
<td>Pack data record</td>
<td>GPRECS</td>
<td>107</td>
<td>141</td>
</tr>
<tr>
<td>Polyl ine</td>
<td>GPL</td>
<td>12</td>
<td>138</td>
</tr>
<tr>
<td>Polymarker</td>
<td>GPM</td>
<td>13</td>
<td>139</td>
</tr>
<tr>
<td>Read item from GKSM</td>
<td>GRDITM</td>
<td>103</td>
<td>261</td>
</tr>
<tr>
<td>Redraw all segments on workstation</td>
<td>GRSNWK</td>
<td>7</td>
<td>274</td>
</tr>
<tr>
<td>Rename segment</td>
<td>GREN1</td>
<td>58</td>
<td>262</td>
</tr>
<tr>
<td>Request choice</td>
<td>GRQCH</td>
<td>84</td>
<td>263</td>
</tr>
<tr>
<td>Request locator</td>
<td>GRQLC</td>
<td>81</td>
<td>265</td>
</tr>
<tr>
<td>Request pick</td>
<td>GRQPK</td>
<td>85</td>
<td>267</td>
</tr>
<tr>
<td>Function</td>
<td>Name</td>
<td>Number</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------------------</td>
<td>--------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Request string (FORTRAN only)</td>
<td>GRQST</td>
<td>86</td>
<td>270</td>
</tr>
<tr>
<td>Request string</td>
<td>GRQSTS</td>
<td>86</td>
<td>271</td>
</tr>
<tr>
<td>Request stroke</td>
<td>GRQSK</td>
<td>82</td>
<td>268</td>
</tr>
<tr>
<td>Request valuator</td>
<td>GRQVL</td>
<td>83</td>
<td>272</td>
</tr>
<tr>
<td>Sample choice</td>
<td>GSMCH</td>
<td>90</td>
<td>299</td>
</tr>
<tr>
<td>Sample locator</td>
<td>GSMLC</td>
<td>87</td>
<td>304</td>
</tr>
<tr>
<td>Sample pick</td>
<td>GSPK</td>
<td>91</td>
<td>305</td>
</tr>
<tr>
<td>Sample string (FORTRAN only)</td>
<td>GMSST</td>
<td>92</td>
<td>307</td>
</tr>
<tr>
<td>Sample string</td>
<td>GMSSTS</td>
<td>92</td>
<td>308</td>
</tr>
<tr>
<td>Sample stroke</td>
<td>GMSK</td>
<td>88</td>
<td>306</td>
</tr>
<tr>
<td>Sample valuator</td>
<td>GSMVL</td>
<td>89</td>
<td>310</td>
</tr>
<tr>
<td>Select normalization transformation</td>
<td>GSELTNT</td>
<td>52</td>
<td>287</td>
</tr>
<tr>
<td>Set aspect source flags</td>
<td>GSASF</td>
<td>41</td>
<td>274</td>
</tr>
<tr>
<td>Set character expansion factor</td>
<td>GSCHXP</td>
<td>28</td>
<td>280</td>
</tr>
<tr>
<td>Set character height</td>
<td>GSCHH</td>
<td>31</td>
<td>276</td>
</tr>
<tr>
<td>Set character spacing</td>
<td>GSCHSP</td>
<td>29</td>
<td>278</td>
</tr>
<tr>
<td>Set character up vector</td>
<td>GSCHUP</td>
<td>32</td>
<td>279</td>
</tr>
<tr>
<td>Set choice mode</td>
<td>GSCHM</td>
<td>78</td>
<td>277</td>
</tr>
<tr>
<td>Set clipping indicator</td>
<td>GSCLIP</td>
<td>53</td>
<td>281</td>
</tr>
<tr>
<td>Set color representation</td>
<td>GSCR</td>
<td>48</td>
<td>282</td>
</tr>
<tr>
<td>Set deferral state</td>
<td>GSDS</td>
<td>9</td>
<td>284</td>
</tr>
<tr>
<td>Set detectability</td>
<td>GSDTEC</td>
<td>68</td>
<td>286</td>
</tr>
<tr>
<td>Set fill area color index</td>
<td>GSFACI</td>
<td>38</td>
<td>288</td>
</tr>
<tr>
<td>Set fill area index</td>
<td>GSAI</td>
<td>35</td>
<td>289</td>
</tr>
<tr>
<td>Set fill area interior style</td>
<td>GSFAS1</td>
<td>36</td>
<td>290</td>
</tr>
<tr>
<td>Set fill area representation</td>
<td>GSFAR</td>
<td>46</td>
<td>291</td>
</tr>
<tr>
<td>Set fill area style index</td>
<td>GSFASI</td>
<td>37</td>
<td>293</td>
</tr>
<tr>
<td>Set highlighting</td>
<td>GSHLIT</td>
<td>66</td>
<td>295</td>
</tr>
<tr>
<td>Set linetype</td>
<td>GSLN</td>
<td>19</td>
<td>297</td>
</tr>
<tr>
<td>Set linewidth scale factor</td>
<td>GSLWSC</td>
<td>20</td>
<td>298</td>
</tr>
<tr>
<td>Set locator mode</td>
<td>GSLCM</td>
<td>75</td>
<td>296</td>
</tr>
<tr>
<td>Set marker size scale factor</td>
<td>GSKMC</td>
<td>24</td>
<td>302</td>
</tr>
<tr>
<td>Set marker type</td>
<td>GSKM</td>
<td>23</td>
<td>301</td>
</tr>
<tr>
<td>Set pattern reference point</td>
<td>GSPARF</td>
<td>40</td>
<td>314</td>
</tr>
<tr>
<td>Set pattern representation</td>
<td>GSPAR</td>
<td>47</td>
<td>312</td>
</tr>
<tr>
<td>Set pattern size</td>
<td>GSPA</td>
<td>39</td>
<td>311</td>
</tr>
<tr>
<td>Set pick identifier</td>
<td>GSPKID</td>
<td>42</td>
<td>315</td>
</tr>
<tr>
<td>Set pick mode</td>
<td>GSPKM</td>
<td>79</td>
<td>316</td>
</tr>
<tr>
<td>Set polyline color index</td>
<td>GSPLCI</td>
<td>21</td>
<td>317</td>
</tr>
<tr>
<td>Set polyline index</td>
<td>GSPLI</td>
<td>18</td>
<td>318</td>
</tr>
<tr>
<td>Set polyline representation</td>
<td>GSPLR</td>
<td>43</td>
<td>319</td>
</tr>
<tr>
<td>Set polymarker color index</td>
<td>GSPMCI</td>
<td>25</td>
<td>321</td>
</tr>
<tr>
<td>Set polymarker index</td>
<td>GSPMI</td>
<td>22</td>
<td>322</td>
</tr>
<tr>
<td>Set polymarker representation</td>
<td>GSPMR</td>
<td>44</td>
<td>323</td>
</tr>
<tr>
<td>Set segment priority</td>
<td>GSSGP</td>
<td>67</td>
<td>325</td>
</tr>
<tr>
<td>Set segment transformation</td>
<td>GSSGT</td>
<td>64</td>
<td>327</td>
</tr>
<tr>
<td>Set string mode</td>
<td>GSSTM</td>
<td>80</td>
<td>329</td>
</tr>
<tr>
<td>Set stroke mode</td>
<td>GSSKM</td>
<td>76</td>
<td>338</td>
</tr>
<tr>
<td>Set text alignment</td>
<td>GSTXAL</td>
<td>34</td>
<td>331</td>
</tr>
<tr>
<td>Set text color index</td>
<td>GSTXCI</td>
<td>30</td>
<td>333</td>
</tr>
<tr>
<td>Set text font and precision</td>
<td>GSTXFP</td>
<td>27</td>
<td>334</td>
</tr>
<tr>
<td>Set text index</td>
<td>GSTXI</td>
<td>26</td>
<td>337</td>
</tr>
<tr>
<td>Set text path</td>
<td>GSTXP</td>
<td>33</td>
<td>338</td>
</tr>
<tr>
<td>Set text representation</td>
<td>GSTXR</td>
<td>45</td>
<td>339</td>
</tr>
<tr>
<td>Set valuator mode</td>
<td>GSVL</td>
<td>77</td>
<td>342</td>
</tr>
<tr>
<td>Set viewport</td>
<td>GSV</td>
<td>50</td>
<td>343</td>
</tr>
<tr>
<td>Set viewport input priority</td>
<td>GSVPIP</td>
<td>51</td>
<td>344</td>
</tr>
</tbody>
</table>
### GDDM-GKS functions

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

### List of functions by function type

#### Control functions

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

#### Output functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPL</td>
<td>Polyl ine</td>
<td>12</td>
<td>138</td>
</tr>
<tr>
<td>GPM</td>
<td>Polymarker</td>
<td>13</td>
<td>139</td>
</tr>
<tr>
<td>GTX</td>
<td>Text (VS FORTRAN only)</td>
<td>14</td>
<td>349</td>
</tr>
<tr>
<td>GTXS</td>
<td>Text</td>
<td>14</td>
<td>351</td>
</tr>
<tr>
<td>GFA</td>
<td>Fill area</td>
<td>15</td>
<td>100</td>
</tr>
<tr>
<td>GCA</td>
<td>Cell array</td>
<td>16</td>
<td>82</td>
</tr>
<tr>
<td>GGDP</td>
<td>Generalized drawing primitive</td>
<td>17</td>
<td>103</td>
</tr>
</tbody>
</table>

#### Output attributes

### Workstation-independent primitive attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSPLI</td>
<td>Set polyline index</td>
<td>18</td>
<td>318</td>
</tr>
<tr>
<td>GSLN</td>
<td>Set linetype</td>
<td>19</td>
<td>297</td>
</tr>
<tr>
<td>GSLWSC</td>
<td>Set linewidth scale factor</td>
<td>20</td>
<td>298</td>
</tr>
<tr>
<td>GSPLCI</td>
<td>Set polyline color index</td>
<td>21</td>
<td>317</td>
</tr>
<tr>
<td>GSPMI</td>
<td>Set polymarker index</td>
<td>22</td>
<td>322</td>
</tr>
</tbody>
</table>
### GDDM-GKS functions

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

#### Workstation attributes (representations)

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSPLR</td>
<td>Set polyline representation</td>
<td>43</td>
<td>319</td>
</tr>
<tr>
<td>GSPMR</td>
<td>Set polymarker representation</td>
<td>44</td>
<td>323</td>
</tr>
<tr>
<td>GSTXR</td>
<td>Set text representation</td>
<td>45</td>
<td>339</td>
</tr>
<tr>
<td>GSFAI</td>
<td>Set fill area representation</td>
<td>46</td>
<td>291</td>
</tr>
<tr>
<td>GSPAR</td>
<td>Set pattern representation</td>
<td>47</td>
<td>312</td>
</tr>
<tr>
<td>GSCR</td>
<td>Set color representation</td>
<td>48</td>
<td>282</td>
</tr>
</tbody>
</table>

### Transformation functions

#### Normalization transformation

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSWN</td>
<td>Set window</td>
<td>49</td>
<td>348</td>
</tr>
<tr>
<td>GSVP</td>
<td>Set viewport</td>
<td>50</td>
<td>343</td>
</tr>
<tr>
<td>GSVPIP</td>
<td>Set viewport input priority</td>
<td>51</td>
<td>344</td>
</tr>
<tr>
<td>GSELT</td>
<td>Select normalization transformation</td>
<td>52</td>
<td>287</td>
</tr>
<tr>
<td>GSCLIP</td>
<td>Set clipping indicator</td>
<td>53</td>
<td>281</td>
</tr>
</tbody>
</table>

#### Workstation transformation

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSWKWN</td>
<td>Set workstation window</td>
<td>54</td>
<td>347</td>
</tr>
<tr>
<td>GSWKVP</td>
<td>Set workstation viewport</td>
<td>55</td>
<td>345</td>
</tr>
</tbody>
</table>
# Segment functions

## Segment manipulation functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCRSG</td>
<td>Create segment</td>
<td>56</td>
<td>89</td>
</tr>
<tr>
<td>GCLSG</td>
<td>Close segment</td>
<td>57</td>
<td>87</td>
</tr>
<tr>
<td>GRENSG</td>
<td>Rename segment</td>
<td>58</td>
<td>262</td>
</tr>
<tr>
<td>GDSG</td>
<td>Delete segment</td>
<td>59</td>
<td>92</td>
</tr>
<tr>
<td>GDSGWK</td>
<td>Delete segment from workstation</td>
<td>60</td>
<td>93</td>
</tr>
<tr>
<td>GASGWK</td>
<td>Associate segment with workstation</td>
<td>61</td>
<td>81</td>
</tr>
<tr>
<td>GCSGWK</td>
<td>Copy segment to workstation</td>
<td>62</td>
<td>90</td>
</tr>
<tr>
<td>GINS</td>
<td>Insert segment</td>
<td>63</td>
<td>122</td>
</tr>
</tbody>
</table>

## Segment attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSSGT</td>
<td>Set segment transformation</td>
<td>64</td>
<td>327</td>
</tr>
<tr>
<td>GSVIS</td>
<td>Set visibility</td>
<td>65</td>
<td>341</td>
</tr>
<tr>
<td>GSHLIT</td>
<td>Set highlighting</td>
<td>66</td>
<td>295</td>
</tr>
<tr>
<td>GSSGP</td>
<td>Set segment priority</td>
<td>67</td>
<td>325</td>
</tr>
<tr>
<td>GSDTEC</td>
<td>Set detectability</td>
<td>68</td>
<td>286</td>
</tr>
</tbody>
</table>

## Input functions

### Initialization of input devices

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GINLC</td>
<td>Initialize locator</td>
<td>69</td>
<td>116</td>
</tr>
<tr>
<td>GINSK</td>
<td>Initialize stroke</td>
<td>70</td>
<td>123</td>
</tr>
<tr>
<td>GINVL</td>
<td>Initialize valuator</td>
<td>71</td>
<td>132</td>
</tr>
<tr>
<td>GINCH</td>
<td>Initialize choice</td>
<td>72</td>
<td>114</td>
</tr>
<tr>
<td>GINPK</td>
<td>Initialize pick</td>
<td>73</td>
<td>120</td>
</tr>
<tr>
<td>GINST</td>
<td>Initialize string (FORTRAN only)</td>
<td>74</td>
<td>127</td>
</tr>
<tr>
<td>GINSTS</td>
<td>Initialize string</td>
<td>74</td>
<td>129</td>
</tr>
</tbody>
</table>
### Setting the mode of input devices

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSLCM</td>
<td>Set locator mode</td>
<td>75</td>
<td>296</td>
</tr>
<tr>
<td>GSSKM</td>
<td>Set stroke mode</td>
<td>76</td>
<td>328</td>
</tr>
<tr>
<td>GSVM</td>
<td>Set valuator mode</td>
<td>77</td>
<td>342</td>
</tr>
<tr>
<td>GSCHM</td>
<td>Set choice mode</td>
<td>78</td>
<td>277</td>
</tr>
<tr>
<td>GSPKM</td>
<td>Set pick mode</td>
<td>79</td>
<td>316</td>
</tr>
<tr>
<td>GSSTM</td>
<td>Set string mode</td>
<td>80</td>
<td>329</td>
</tr>
</tbody>
</table>

### Request input functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRQLC</td>
<td>Request locator</td>
<td>81</td>
<td>265</td>
</tr>
<tr>
<td>GRQSK</td>
<td>Request stroke</td>
<td>82</td>
<td>268</td>
</tr>
<tr>
<td>GRQVL</td>
<td>Request valuator</td>
<td>83</td>
<td>272</td>
</tr>
<tr>
<td>GRQCH</td>
<td>Request choice</td>
<td>84</td>
<td>263</td>
</tr>
<tr>
<td>GRQPK</td>
<td>Request pick</td>
<td>85</td>
<td>267</td>
</tr>
<tr>
<td>GRQST</td>
<td>Request string (FORTRAN only)</td>
<td>86</td>
<td>270</td>
</tr>
<tr>
<td>GRQSTS</td>
<td>Request string</td>
<td>86</td>
<td>271</td>
</tr>
</tbody>
</table>

### Sample input functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSMLC</td>
<td>Sample locator</td>
<td>87</td>
<td>304</td>
</tr>
<tr>
<td>GSMSK</td>
<td>Sample stroke</td>
<td>88</td>
<td>306</td>
</tr>
<tr>
<td>GSMVL</td>
<td>Sample valuator</td>
<td>89</td>
<td>310</td>
</tr>
<tr>
<td>GSMCH</td>
<td>Sample choice</td>
<td>90</td>
<td>299</td>
</tr>
<tr>
<td>GSMPK</td>
<td>Sample pick</td>
<td>91</td>
<td>305</td>
</tr>
<tr>
<td>GSMST</td>
<td>Sample string (FORTRAN only)</td>
<td>92</td>
<td>307</td>
</tr>
<tr>
<td>GSMSTS</td>
<td>Sample string</td>
<td>92</td>
<td>308</td>
</tr>
</tbody>
</table>

### Event input functions

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

### Metafile functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GWITM</td>
<td>Write item to GKSM</td>
<td>101</td>
<td>358</td>
</tr>
<tr>
<td>GGTITM</td>
<td>Get item type from GKSM</td>
<td>102</td>
<td>106</td>
</tr>
<tr>
<td>GRDITM</td>
<td>Read item from GKSM</td>
<td>103</td>
<td>261</td>
</tr>
<tr>
<td>GIITM</td>
<td>Interpret item</td>
<td>104</td>
<td>112</td>
</tr>
</tbody>
</table>
### Utility functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Number</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEVTM</td>
<td>Evaluate transformation matrix</td>
<td>105</td>
<td>98</td>
</tr>
<tr>
<td>GACTM</td>
<td>Accumulate transformation matrix</td>
<td>106</td>
<td>78</td>
</tr>
<tr>
<td>GPRECS</td>
<td>Pack data record</td>
<td>107</td>
<td>140</td>
</tr>
<tr>
<td>GPRECS</td>
<td>Pack data record</td>
<td>107</td>
<td>141</td>
</tr>
<tr>
<td>GUREC</td>
<td>Unpack data record (FORTRAN only)</td>
<td>108</td>
<td>352</td>
</tr>
<tr>
<td>GURECS</td>
<td>Unpack data record</td>
<td>108</td>
<td>353</td>
</tr>
</tbody>
</table>

### Inquiry functions

#### Inquiry function for operating state value

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQOPS</td>
<td>Inquire operating state value</td>
<td>201</td>
</tr>
</tbody>
</table>

#### Inquiry functions for GKS description table

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQLVKS</td>
<td>Inquire level of GKS</td>
<td>194</td>
</tr>
<tr>
<td>GQEWK</td>
<td>Inquire list element of available workstation types</td>
<td>181</td>
</tr>
<tr>
<td>GQWKM</td>
<td>Inquire workstation maximum numbers</td>
<td>258</td>
</tr>
<tr>
<td>GQMNTN</td>
<td>Inquire maximum normalization transformation number</td>
<td>199</td>
</tr>
</tbody>
</table>
### Inquiry functions for GKS state list

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQOPWK</td>
<td>Inquire set member of open workstations</td>
<td>202</td>
</tr>
<tr>
<td>GQACWK</td>
<td>Inquire set member of active workstations</td>
<td>143</td>
</tr>
<tr>
<td>GQPLI</td>
<td>Inquire polyline index</td>
<td>214</td>
</tr>
<tr>
<td>GQPMI</td>
<td>Inquire polymarker index</td>
<td>219</td>
</tr>
<tr>
<td>GQTXI</td>
<td>Inquire text index</td>
<td>244</td>
</tr>
<tr>
<td>GQCHH</td>
<td>Inquire character height</td>
<td>148</td>
</tr>
<tr>
<td>GQCHUP</td>
<td>Inquire character up vector</td>
<td>151</td>
</tr>
<tr>
<td>GQCHW</td>
<td>Inquire character width</td>
<td>152</td>
</tr>
<tr>
<td>GQCHB</td>
<td>Inquire character base vector</td>
<td>147</td>
</tr>
<tr>
<td>GQTXP</td>
<td>Inquire text path</td>
<td>245</td>
</tr>
<tr>
<td>GQCHUP</td>
<td>Inquire text alignment</td>
<td>240</td>
</tr>
<tr>
<td>GQFAI</td>
<td>Inquire fill area index</td>
<td>184</td>
</tr>
<tr>
<td>GQPA</td>
<td>Inquire pattern size</td>
<td>203</td>
</tr>
<tr>
<td>GQPARF</td>
<td>Inquire pattern reference point</td>
<td>206</td>
</tr>
<tr>
<td>GQPND</td>
<td>Inquire pick definition</td>
<td>209</td>
</tr>
<tr>
<td>GQLN</td>
<td>Inquire linetype</td>
<td>193</td>
</tr>
<tr>
<td>GQLWSC</td>
<td>Inquire linewidth scale factor</td>
<td>196</td>
</tr>
<tr>
<td>GQPLCI</td>
<td>Inquire polyline color index</td>
<td>212</td>
</tr>
<tr>
<td>GQMK</td>
<td>Inquire marker type</td>
<td>197</td>
</tr>
<tr>
<td>GQMKSC</td>
<td>Inquire marker size scale factor</td>
<td>198</td>
</tr>
<tr>
<td>GQPMCI</td>
<td>Inquire polymarker color index</td>
<td>216</td>
</tr>
<tr>
<td>GQTXFP</td>
<td>Inquire text font and precision</td>
<td>243</td>
</tr>
<tr>
<td>GQCHXP</td>
<td>Inquire character expansion factor</td>
<td>153</td>
</tr>
<tr>
<td>GQCHSP</td>
<td>Inquire character spacing</td>
<td>150</td>
</tr>
<tr>
<td>GQTXCI</td>
<td>Inquire text color index</td>
<td>241</td>
</tr>
<tr>
<td>GQFAIS</td>
<td>Inquire fill area interior style</td>
<td>185</td>
</tr>
<tr>
<td>GQFASI</td>
<td>Inquire fill area style index</td>
<td>187</td>
</tr>
<tr>
<td>GQFACI</td>
<td>Inquire fill area color index</td>
<td>182</td>
</tr>
<tr>
<td>GQASF</td>
<td>Inquire aspect source flags</td>
<td>144</td>
</tr>
<tr>
<td>GQCNTR</td>
<td>Inquire current normalization transformation number</td>
<td>134</td>
</tr>
<tr>
<td>GQENTN</td>
<td>Inquire list element of normalization transformation numbers</td>
<td>175</td>
</tr>
<tr>
<td>GQNT</td>
<td>Inquire normalization transformation</td>
<td>200</td>
</tr>
<tr>
<td>GQCLIP</td>
<td>Inquire clipping indicator</td>
<td>154</td>
</tr>
<tr>
<td>GQPSG</td>
<td>Inquire name of open segment</td>
<td>201</td>
</tr>
<tr>
<td>GQSIM</td>
<td>Inquire more simultaneous events</td>
<td>232</td>
</tr>
</tbody>
</table>

### Inquiry functions for workstation state list

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQWKC</td>
<td>Inquire workstation connection and type</td>
<td>254</td>
</tr>
<tr>
<td>GQWS</td>
<td>Inquire workstation state</td>
<td>259</td>
</tr>
<tr>
<td>GQWKDU</td>
<td>Inquire workstation deferral and update states</td>
<td>257</td>
</tr>
<tr>
<td>GQEPGI</td>
<td>Inquire list element of polyline indexes</td>
<td>177</td>
</tr>
<tr>
<td>GQPLR</td>
<td>Inquire polyline representation</td>
<td>215</td>
</tr>
<tr>
<td>GQEPMI</td>
<td>Inquire list element of polymarker indexes</td>
<td>178</td>
</tr>
<tr>
<td>GQPMR</td>
<td>Inquire polymarker representation</td>
<td>219</td>
</tr>
<tr>
<td>GQTXI</td>
<td>Inquire list element of text indexes</td>
<td>179</td>
</tr>
<tr>
<td>GQTXR</td>
<td>Inquire text representation</td>
<td>246</td>
</tr>
<tr>
<td>GQTXS</td>
<td>Inquire text extent (VS FORTRAN only)</td>
<td>247</td>
</tr>
<tr>
<td>GQEFAS</td>
<td>Inquire list element of fill area indexes</td>
<td>173</td>
</tr>
<tr>
<td>GQFAR</td>
<td>Inquire fill area representation</td>
<td>186</td>
</tr>
<tr>
<td>GQEPAL</td>
<td>Inquire list element of pattern indexes</td>
<td>176</td>
</tr>
<tr>
<td>GQPAR</td>
<td>Inquire pattern representation</td>
<td>205</td>
</tr>
</tbody>
</table>
### Inquiry functions for workstation description table

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQWKCA</td>
<td>Inquire workstation category</td>
<td>255</td>
</tr>
<tr>
<td>GQWKCL</td>
<td>Inquire workstation classification</td>
<td>256</td>
</tr>
<tr>
<td>GQDSP</td>
<td>Inquire display space size</td>
<td>166</td>
</tr>
<tr>
<td>GQDNKA</td>
<td>Inquire dynamic modification of workstation attributes</td>
<td>170</td>
</tr>
<tr>
<td>GQDDS</td>
<td>Inquire default deferral state values</td>
<td>158</td>
</tr>
<tr>
<td>GQLPF</td>
<td>Inquire polyline facilities</td>
<td>213</td>
</tr>
<tr>
<td>GQPLLR</td>
<td>Inquire predefined polyline representation</td>
<td>222</td>
</tr>
<tr>
<td>GQPMF</td>
<td>Inquire polymarker facilities</td>
<td>217</td>
</tr>
<tr>
<td>GQPPMR</td>
<td>Inquire predefined polymarker representation</td>
<td>223</td>
</tr>
<tr>
<td>GQTXF</td>
<td>Inquire text facilities</td>
<td>241</td>
</tr>
<tr>
<td>GQPTXR</td>
<td>Inquire predefined text representation</td>
<td>224</td>
</tr>
<tr>
<td>GQFAF</td>
<td>Inquire fill area facilities</td>
<td>182</td>
</tr>
<tr>
<td>GQPFAR</td>
<td>Inquire predefined fill area representation</td>
<td>208</td>
</tr>
<tr>
<td>GQPAF</td>
<td>Inquire pattern facilities</td>
<td>204</td>
</tr>
<tr>
<td>GQPPAR</td>
<td>Inquire predefined pattern representation</td>
<td>221</td>
</tr>
<tr>
<td>GQCF</td>
<td>Inquire color facilities</td>
<td>146</td>
</tr>
<tr>
<td>GQPCR</td>
<td>Inquire predefined color representation</td>
<td>207</td>
</tr>
<tr>
<td>GQEGDP</td>
<td>Inquire list element of available generalized drawing primitives</td>
<td>174</td>
</tr>
<tr>
<td>GQGDP</td>
<td>Inquire generalized drawing primitive</td>
<td>188</td>
</tr>
<tr>
<td>GQLWK</td>
<td>Inquire maximum length of workstation state tables</td>
<td>195</td>
</tr>
<tr>
<td>GQSGP</td>
<td>Inquire number of segment priorities supported</td>
<td>230</td>
</tr>
<tr>
<td>GQDSGA</td>
<td>Inquire dynamic modification of segment attributes</td>
<td>163</td>
</tr>
<tr>
<td>GQL1</td>
<td>Inquire number of available logical input devices</td>
<td>192</td>
</tr>
<tr>
<td>GQLDC</td>
<td>Inquire default locator device data</td>
<td>160</td>
</tr>
<tr>
<td>GQDSK</td>
<td>Inquire default stroke device data</td>
<td>164</td>
</tr>
<tr>
<td>GQDVLD</td>
<td>Inquire default valuater device data</td>
<td>168</td>
</tr>
<tr>
<td>GQDCH</td>
<td>Inquire default choice device data</td>
<td>157</td>
</tr>
<tr>
<td>GQDPK</td>
<td>Inquire default pick device data</td>
<td>161</td>
</tr>
<tr>
<td>GQDST</td>
<td>Inquire default string device data</td>
<td>167</td>
</tr>
</tbody>
</table>

### Inquiry functions for segment state list

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQASWK</td>
<td>Inquire set member of associated workstations</td>
<td>145</td>
</tr>
<tr>
<td>GQSGA</td>
<td>Inquire segment attributes</td>
<td>229</td>
</tr>
</tbody>
</table>
Pixel inquiries

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQPXA</td>
<td>Inquire pixel array</td>
<td>227</td>
</tr>
<tr>
<td>GPX</td>
<td>Inquire pixel</td>
<td>225</td>
</tr>
</tbody>
</table>

Inquiry function for GKS error state list

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQIQOV</td>
<td>Inquire input queue overflow</td>
<td>189</td>
</tr>
</tbody>
</table>

Error-handling functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GECLKS</td>
<td>Emergency close GKS</td>
<td>94</td>
</tr>
<tr>
<td>GERHND</td>
<td>Error handling</td>
<td>95</td>
</tr>
<tr>
<td>GERLOG</td>
<td>Error logging</td>
<td>96</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Function: To accumulate transformation matrix.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utility function. Composes a new transformation matrix from a pre-existing matrix and new input values for rotation, shift, and scale.</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>minp (specified by user) (array of short floating-point numbers)</th>
</tr>
</thead>
<tbody>
<tr>
<td>An existing transformation matrix. You can use the Inquire segment attributes (GQSGA) function to obtain the values for this parameter.</td>
</tr>
<tr>
<td>xo (specified by user) (short floating point)</td>
</tr>
<tr>
<td>yo (specified by user) (short floating point)</td>
</tr>
<tr>
<td>A given point, in world coordinates or normalized device coordinates.</td>
</tr>
</tbody>
</table>
**GDDM-GKS functions**

**dx (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.

**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)**  
The scale factors. These factors control the size of the segment on the display surface.

**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 \( \text{minp} \) 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 (GINS).

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:

\[ \text{radians} = \frac{\pi}{180} \times \text{degrees} \]

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 matrices. 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:

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

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**

<table>
<thead>
<tr>
<th>Function:</th>
<th>To activate workstation.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control function. Makes the workstation ready for GKS output.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GACWK</th>
<th>(wkid)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X’38000400’ (939525120)</td>
</tr>
<tr>
<td>APL code</td>
<td>1307</td>
</tr>
</tbody>
</table>

GDDM-GKS Programming Guide and Reference
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

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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>GKS not in proper state: GKS shall be either in the state WSOP or in the state WSAC</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>27</td>
<td>Workstation Independent Segment Storage is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>120</td>
<td>Specified segment name is invalid</td>
</tr>
<tr>
<td>124</td>
<td>Specified segment does not exist on Workstation Independent Segment Storage</td>
</tr>
</tbody>
</table>

GCA

Purpose
GDDM-GKS functions

<table>
<thead>
<tr>
<th>Function:</th>
<th>To cell array.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output function. Generates a cell array primitive.</td>
<td></td>
</tr>
</tbody>
</table>

**Parameters**

- \(\text{px} \) (specified by user) (short floating point)
- \(\text{py} \) (specified by user) (short floating point)
- \(\text{qx} \) (specified by user) (short floating point)
- \(\text{qy} \) (specified by user) (short floating point)
- \(\text{dimx} \) (specified by user) (fullword integer)
- \(\text{dimy} \) (specified by user) (fullword integer)
- \(\text{isc} \) (specified by user) (fullword integer)
- \(\text{isr} \) (specified by user) (fullword integer)
- \(\text{dx} \) (specified by user) (fullword integer)
- \(\text{dy} \) (specified by user) (fullword integer)
- \(\text{colia} \) (specified by user) (an array of fullword integers)

**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, \(dx \times dy\), of equal-sized cells. The width of each cell is \(vpx - qx\) \(+ dx\), and the height is \(vpy - qy\) \(+ dy\). (The symbol “\(v\)” 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 \(\text{colia}\). 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)/4,4,4,4,5,4,4,4,4,4,
  $ 4,4,4,4,1,1,4,4,4,4,
  $ 2,4,4,4,1,1,4,4,4,2,
  $ 2,2,1,2,2,2,1,2,2,2,
  $ 2,4,1,1,3,1,1,4,2,
  $ 4,4,4,4,3,3,4,4,4,4,
  $ 4,4,4,1,3,3,4,4,4,4,
  $ 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

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

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).
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.

colf (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 colf.

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

If colf 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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>GKS not in proper state: GKS shall be either in the state WSOP or in the state WSAC</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>

GCLSG

Purpose

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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>GKS not in proper state: GKS shall be in the state SGOP</td>
</tr>
</tbody>
</table>

GCLWK
GDDM-GKS functions

### Purpose

**GCLWK**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wkid</td>
<td>(specified by user) (fullword integer)</td>
</tr>
</tbody>
</table>

The identifier selected when the workstation was opened.

### 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

<table>
<thead>
<tr>
<th>GCRSG</th>
<th>(sgna)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1357</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38003800' (939538432)</td>
</tr>
</tbody>
</table>

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

- **GKS not in proper state:** GKS shall be in the state WSAC
- **Specified segment name is invalid**
- **Specified segment name is already in use**

### GCSGWK

**Purpose**

To copy segment to workstation.

**Function:** 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.

**APL code** 1363

**GKS RCP code** X’38003E00’ (939539968)
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

<table>
<thead>
<tr>
<th>GDAWK</th>
<th>(wkid)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1308</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38000500' (939525376)</td>
</tr>
</tbody>
</table>
**GDDM-GKS functions**

**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**

<table>
<thead>
<tr>
<th>GDSG</th>
<th>(sgna)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1360</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X ‘38003B00’ (939539200)</td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>120</td>
<td>Specified segment name is invalid</td>
</tr>
<tr>
<td>122</td>
<td>Specified segment does not exist</td>
</tr>
<tr>
<td>125</td>
<td>Specified segment is open</td>
</tr>
</tbody>
</table>

GDSGWK

Purpose

<table>
<thead>
<tr>
<th>GDSGWK</th>
<th>(wkid, sgna)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>APL code</th>
<th>1361</th>
</tr>
</thead>
</table>

| GKS RCP code | X‘38003C00’ (939539456) |

Function: To delete segment from workstation.

Segment function. Deletes a segment from the specified workstation.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>120</td>
<td>Specified segment name is invalid</td>
</tr>
<tr>
<td>123</td>
<td>Specified segment does not exist on specified workstation</td>
</tr>
<tr>
<td>125</td>
<td>Specified segment is open</td>
</tr>
</tbody>
</table>

GECLKS

Purpose

<table>
<thead>
<tr>
<th>GECLKS</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>APL code</th>
<th>1304</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X'3800F000' (939585536)</td>
</tr>
</tbody>
</table>
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).
GDDM-GKS functions

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)

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

Function: To escape.

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

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fctid</td>
<td>specified by user (fullword integer)</td>
</tr>
<tr>
<td>lidr</td>
<td>specified by user (fullword integer)</td>
</tr>
<tr>
<td>idr</td>
<td>specifies array of 80-byte character tokens</td>
</tr>
<tr>
<td>mlodr</td>
<td>specified by user (fullword integer)</td>
</tr>
<tr>
<td>lodr</td>
<td>returned by GDDM (fullword integer)</td>
</tr>
<tr>
<td>odr</td>
<td>returned by GDDM (array of 80-byte character tokens)</td>
</tr>
</tbody>
</table>

Chapter 5. GDDM-GKS functions
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>180</td>
<td>Specified escape function is not supported</td>
</tr>
<tr>
<td>181</td>
<td>Specified escape function identification is invalid</td>
</tr>
</tbody>
</table>

GEVTM

Purpose

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)

The given point in either world or normalized device coordinates.

yo (specified by user) (short floating point)

The given point in either world or normalized device coordinates.

dx (specified by user) (short floating point)

The shift vector in either world or normalized device coordinates.

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:
  \[ \text{radians} = \frac{\pi}{180} \times \text{degrees} \]

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:
GDDM-GKS functions

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

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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>

GFA

Purpose

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.
GDDM-GKS functions

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

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


<table>
<thead>
<tr>
<th>Function:</th>
<th>To flush device events.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input function. Deletes all specified input class events from the input queue. (See the section “GDDM-GKS restrictions” below.)</td>
<td></td>
</tr>
</tbody>
</table>

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
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GGDP</th>
<th>(n, px, py, primid, ldr, datrec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1493</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38001100' (939528448)</td>
</tr>
</tbody>
</table>
**GDDM-GKS functions**

*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)*
  The coordinates of the points in world coordinates.

- **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

<table>
<thead>
<tr>
<th>Function:</th>
<th>To get choice.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input function. Returns a choice number. (See the section &quot;GDDM-GKS restrictions&quot; below.)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>stat</strong> <em>(returned by GDDM) (fullword integer)</em></td>
</tr>
<tr>
<td>The status of the choice device. The possible choices are:</td>
</tr>
<tr>
<td>0 (GOK) OK</td>
</tr>
<tr>
<td>0 (GNCHOI) No choice</td>
</tr>
<tr>
<td>The status indicates whether or not the choice number obtained from the operator is valid.</td>
</tr>
<tr>
<td><strong>chnr</strong> <em>(returned by GDDM) (fullword integer)</em></td>
</tr>
<tr>
<td>The choice number. The choice number is an integer that represents one of a number of choices available on a choice device.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operating states</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSOP, WSAC, SGOP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Related functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize choice (GINCH), Request choice (GRQCH), Set choice mode (GSCHM)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(See the section “GDDM-GKS restrictions” below.)</td>
</tr>
<tr>
<td>This function returns the choice logical input value in the current event report.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GDDM-GKS Restrictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS does not support this function at level 2b. If this function is invoked, GDDM-GKS reports error 143.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Principal errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>143 EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>150 No input value of the correct class is in the current event report</td>
</tr>
</tbody>
</table>
GGDM-GKS functions

GGTITM

Purpose

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

Function: To get locator.

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

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tnr</td>
<td>returned by GDDM (fullword integer)</td>
</tr>
<tr>
<td></td>
<td>The normalization transformation number.</td>
</tr>
<tr>
<td>lpx</td>
<td>returned by GDDM (short floating point)</td>
</tr>
<tr>
<td>lpy</td>
<td>returned by GDDM (short floating point)</td>
</tr>
<tr>
<td></td>
<td>The locator position in world coordinates.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>150</td>
<td>No input value of the correct class is in the current event report</td>
</tr>
</tbody>
</table>
GGTPK

Purpose

<table>
<thead>
<tr>
<th>Function</th>
<th>GGTPK</th>
<th>(stat, sgna, pkid)</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>APL code</th>
<th>1399</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X’38006300’ (939549440)</td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>GGTSK</th>
<th>(n, tnr, np, px, py)</th>
</tr>
</thead>
</table>

**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.
GDDM-GKS functions

Principal errors

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>150</td>
<td>No input value of the correct class is in the current event report</td>
</tr>
</tbody>
</table>

GGTST

Purpose

| 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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lostr</td>
<td>(returned by GDDM) (fullword integer) The number of characters returned.</td>
</tr>
<tr>
<td>str</td>
<td>(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.</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>150</td>
<td>No input value of the correct class is in the current event report</td>
</tr>
</tbody>
</table>
GGTSTS

Purpose

<table>
<thead>
<tr>
<th>Function:</th>
<th>To get string.</th>
</tr>
</thead>
<tbody>
<tr>
<td>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.)</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>150</td>
<td>No input value of the correct class is in the current event report</td>
</tr>
</tbody>
</table>
GGTVL

**Purpose**

**GGTVL**  
(val)

<table>
<thead>
<tr>
<th>APL code</th>
<th>1396</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X'38006100' (939548928)</td>
</tr>
</tbody>
</table>

*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**

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>150</td>
<td>No input value of the correct class is in the current event report</td>
</tr>
</tbody>
</table>

GIITM
Purpose

**GIITM**

<table>
<thead>
<tr>
<th>Function:</th>
<th>To interpret item.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metafile function. Interprets the contents of a metafile item.</td>
<td></td>
</tr>
</tbody>
</table>

**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:
GDDM-GKS functions

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

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.
GDDM-GKS functions

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

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

GINLC

Purpose

**GINLC**

<table>
<thead>
<tr>
<th>GINLC</th>
<th>((wkid, lcdnr, tnr, ipx, ipy, pet, xmin, xmax, ymin, ymax, ldr, datrec))</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1370</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38004500’ (939541760)</td>
</tr>
</tbody>
</table>

*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)*
  The initial locator position in world coordinates.
- **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:

```plaintext
il = 1
ia = (attribute control flag=GCURNT)
r1 = 0
ra = ()
s1 = 0
lstr = ()
str = ()
```

If attributes are to be specified in the data record, the parameters of the Pack data record (GPRECS) function are:

```plaintext
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)
r1 = 1
ra = (line width scale factor)
s1 = 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:

```plaintext
il = 2
ia = (polyline/fill area control flag=GPLINE or
    GFILLA, attribute control flag=GCURNT)
r1 = 0
ra = ()
s1 = 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

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 \textit{(specified by user)} \textit{(fullword integer)}

The initial segment name. If \textit{istat} is \textit{GOK}, this parameter must identify a segment at the workstation that is visible and detectable.

\textbf{ipkid} \textit{(specified by user)} \textit{(fullword integer)}

The initial pick identifier. If \textit{istat} is \textit{GOK}, this parameter must be the pick identifier of at least one primitive in the segment \textit{isgna}.

\textbf{pet} \textit{(specified by user)} \textit{(fullword integer)}

The prompt and echo type.

\textbf{xmin} \textit{(specified by user)} \textit{(short floating point)} \textbf{xmax} \textit{(specified by user)} \textit{(short floating point)} \textbf{ymin} \textit{(specified by user)} \textit{(short floating point)} \textbf{ymax} \textit{(specified by user)} \textit{(short floating point)}

The echo area in device coordinates.

\textbf{ldr} \textit{(specified by user)} \textit{(fullword integer)}

The dimension of the data record array \textit{datrec}.

\textbf{datrec} \textit{(specified by user)} \textit{(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:

\textbf{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 \textit{istat} is \textit{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**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>37</td>
<td>Specified workstation is not of category OUTIN</td>
</tr>
<tr>
<td>51</td>
<td>Rectangle definition is invalid</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>141</td>
<td>Input device is not in REQUEST mode</td>
</tr>
<tr>
<td>144</td>
<td>Specified prompt and echo type is not supported on this workstation</td>
</tr>
<tr>
<td>145</td>
<td>Echo area is outside display space</td>
</tr>
<tr>
<td>146</td>
<td>Contents of input data record are invalid</td>
</tr>
<tr>
<td>152</td>
<td>Initial value is invalid</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>
GINSG

Purpose
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

\textbf{sgna} (specified by user) (fullword integer)
The segment name selected when the segment was created.

\textbf{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 \textit{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 \textit{m} is generated by calling the Evaluate transformation matrix (GEVTM) or Accumulate transformation matrix (GACTM) utility functions. The coordinates of the primitives in segment \textit{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 matrices 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**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP</td>
</tr>
<tr>
<td>27</td>
<td>Workstation Independent Segment Storage is not open</td>
</tr>
<tr>
<td>120</td>
<td>Specified segment name is invalid</td>
</tr>
<tr>
<td>124</td>
<td>Specified segment does not exist on Workstation Independent Segment Storage</td>
</tr>
<tr>
<td>125</td>
<td>Specified segment is open</td>
</tr>
</tbody>
</table>

**GINSK**

**Purpose**

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)*
  The initial stroke in world coordinates.

- **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.
**GDDM-GKS functions**

- **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:
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:

\[
\begin{align*}
    i1 &= 1 \\
    ia &= (\text{attribute control flag} = \text{GCURNT}) \\
    rl &= 3 \\
    ra &= (x\text{-interval}, \ y\text{-interval}, \ \text{time interval in seconds}) \\
    sl &= 0 \\
    lstr &= () \\
    str &= ()
\end{align*}
\]

If attributes are to be specified in the data record, the parameters of the Pack data record (GPRECS) function are:

\[
\begin{align*}
    i1 &= 7 \\
    ia &= (\text{attribute control flag} = \text{GSPEC}, \ \text{marker type aspect source flag (ASF)}, \ \text{marker size scale factor ASF}, \ \text{polymarker color index ASF}, \ \text{polymarker index}, \ \text{marker type}, \ \text{polymarker color index}) \\
    rl &= 4 \\
    ra &= (x\text{-interval}, \ y\text{-interval}, \ \text{time interval in seconds}, \ \text{marker size scale factor}) \\
    sl &= 0 \\
    lstr &= () \\
    str &= ()
\end{align*}
\]

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:
GDDM-GKS functions

\[
\begin{align*}
il & = 1 \\
ia & = (\text{attribute control flag}=\text{GCURNT}) \\
r1 & = 3 \\
ra & = (\text{x-interval}, \\
& \quad \text{y-interval}, \\
& \quad \text{time interval in seconds}) \\
s1 & = 0 \\
lstr & = () \\
str & = () \\
\end{align*}
\]

If attributes are to be specified in the data record, the parameters of the Pack data record (GPRECS) function are:

\[
\begin{align*}
il & = 7 \\
ia & = (\text{attribute control flag}=\text{GSPEC}), \\
& \quad \text{linetype ASF}, \\
& \quad \text{linewidth scale factor ASF}, \\
& \quad \text{polyline color index ASF}, \\
& \quad \text{polyline index}, \\
& \quad \text{linetype}, \\
& \quad \text{polyline color index}) \\
r1 & = 4 \\
ra & = (\text{x-interval}, \\
& \quad \text{y-interval}, \\
& \quad \text{time interval in seconds,} \\
& \quad \text{linewidth scale factor}) \\
s1 & = 0 \\
lstr & = () \\
str & = () \\
\end{align*}
\]

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
GINST

<table>
<thead>
<tr>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>GINST</td>
</tr>
</tbody>
</table>

**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.

```plaintext
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**

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. 
GDDM-GKS functions

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.

![Initial cursor position=7](image)

**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
GDDM-GKS functions

GINVL

Purpose

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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GMSGS</th>
<th>(wkid, lstr, mess)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1490</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38000A00’ (939526656)</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>GOPKS</th>
<th>(errfil, bufa)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1300</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38000000' (939524096)</td>
</tr>
</tbody>
</table>
**GDDM-GKS functions**

**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**

<table>
<thead>
<tr>
<th>GOPWK</th>
<th>(wkid, conid, wtype)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1302</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38000200’ (939524608)</td>
</tr>
</tbody>
</table>
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)
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GPL</th>
<th>(n, px, py)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1313</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38000C00' (939527168)</td>
</tr>
</tbody>
</table>

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)*

<table>
<thead>
<tr>
<th>APL code</th>
<th>1314</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X '38000D00' (939527424)</td>
</tr>
</tbody>
</table>

*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)
The coordinates of the n markers, in world coordinates.
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.

Istr (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.

erind (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

<table>
<thead>
<tr>
<th>Purpose</th>
<th>GPRECS</th>
<th>(ii, ia, rl, ra, sl, mstr, Istr, str, mldr, errind, ldr, datrec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1513</td>
<td></td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38006B00’ (939551488)</td>
<td></td>
</tr>
</tbody>
</table>
**GDDM-GKS functions**

**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

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
GDDM-GKS functions

GQASF

Purpose

<table>
<thead>
<tr>
<th>GQASF</th>
<th>(errind, lastf)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1460</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X '3800A100' 939565312</td>
</tr>
</tbody>
</table>

*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.

**lastf** *(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
- Polyl ine 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.
GDDM-GKS functions

Principal errors

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

GQASWK

Purpose

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:
GDDM-GKS functions

n=0
CALL GQASWK (sgna, n, errind, ol, wkid)
DO 55 n = 1, ol
CALL GQASWK (sgna, n, errind, ol, wkid)
WRITE (*) , 5=zerodot) 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

<table>
<thead>
<tr>
<th>GQCF</th>
<th>(wtype, errind, ncoli, cola, npci)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1415</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'3800D000' (939577344)</td>
</tr>
</tbody>
</table>

*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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>39</td>
<td>Specified workstation is neither of category OUTPUT nor of category OUTIN</td>
</tr>
</tbody>
</table>

GQCHB

Purpose

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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Function:</th>
<th>To inquire character height.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inquiry function. Returns the current character height from the GKS state list.</td>
<td></td>
</tr>
</tbody>
</table>

Parameters

errind \( (\text{returned by GDDM}) \) \((\text{fullword integer})\)
The error indicator.

chh \( (\text{returned by GDDM}) \) \((\text{short floating point})\)
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

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).
GDDM-GKS functions

**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**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>2001</td>
<td>Output parameter size insufficient</td>
</tr>
</tbody>
</table>

**GQCHSP**

**Purpose**

<table>
<thead>
<tr>
<th>GQCHSP</th>
<th>(errind, chsp)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1455</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X‘38009C00’ (939564032)</td>
</tr>
</tbody>
</table>
**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**

<table>
<thead>
<tr>
<th>GQCHUP</th>
<th>(errind, chux, chuy)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1433</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'3800B00' (939559680)</td>
</tr>
</tbody>
</table>

**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)*
  The current character up vector in world coordinates.
- **chuy** *(returned by GDDM) (short floating point)*
  The current character up vector in world coordinates.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Function:</th>
<th>To inquire character width.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inquiry function. Returns the current character width from the GKS state list.</td>
<td></td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>errind</td>
<td>(returned by GDDM) (fullword integer)</td>
<td>The error indicator.</td>
</tr>
<tr>
<td>chw</td>
<td>(returned by GDDM) (short floating point)</td>
<td>The current character width in world coordinates.</td>
</tr>
</tbody>
</table>

Operating states
GKOP, WSOP, WSAC, SGOP

Related functions
Set character height (GSCHH)
GDDM-GKS functions

**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**
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
GDDM-GKS functions

GQCLIP

Purpose

<table>
<thead>
<tr>
<th>Function:</th>
<th>To inquire clipping indicator.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inquiry function. Returns the current clipping indicator from the GKS state list.</td>
<td></td>
</tr>
</tbody>
</table>

Parameters

<p>| 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: |</p>
<table>
<thead>
<tr>
<th>clrect (returned by GDDM) (array of short floating-point numbers)</th>
<th>The clipping rectangle. This parameter returns the clipping rectangle boundaries in normalized device coordinates.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (GNCLIP) No clipping</td>
<td></td>
</tr>
<tr>
<td>1 (GCLIP) Clipping</td>
<td></td>
</tr>
</tbody>
</table>

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
### GQCNTN

**Purpose**

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)*

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1482</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'3800B800' (939571200)</td>
</tr>
</tbody>
</table>

---

Chapter 5. GDDM-GKS functions 155
**GDDM-GKS functions**

*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

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.
GDDM-GKS functions

- **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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>2001</td>
<td>Output parameter size insufficient</td>
</tr>
<tr>
<td>2002</td>
<td>List element or set member not available</td>
</tr>
</tbody>
</table>

GQDDS

Purpose
**GDDM-GKS functions**

**GQDDS**

<table>
<thead>
<tr>
<th>(wtype, errind, defmod, regmod)</th>
</tr>
</thead>
</table>

- **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
**GDDM-GKS functions**

---

**GQDLC**

**Purpose**

<table>
<thead>
<tr>
<th>GQDLC</th>
<th>(wtype, devno, n, mldr, errind, dpx, dpy, ol, pet, earea, ldr, datrec)</th>
</tr>
</thead>
</table>

APL code

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)*
  Default initial position of locator in world coordinates.
- **dpy** *(returned by GDDM) (short floating point)*
- **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 \texttt{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \texttt{errind} returns an error indicator.

The Unpack data record (GURECS) utility is available to unpack the data record returned by GQDLC.

The parameter \texttt{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 \texttt{errind}. Usually, a maximum length of 80 (\texttt{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

Function: To inquire default pick device data.

Inquiry function. Returns the default data for a \textit{pick} device at a workstation, from the workstation description table. (See the section “GDDM-GKS restrictions” below.)

Parameters

- **\texttt{wtype} (specified by user) (fullword integer)**  
The workstation type.
- **\texttt{devno} (specified by user) (fullword integer)**  
The pick device number.
- **\texttt{n} (specified by user) (fullword integer)**  
The element requested from the list of available prompt and echo types.
GDDM-GKS functions

**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**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>2001</td>
<td>Output parameter size insufficient</td>
</tr>
<tr>
<td>2002</td>
<td>List element or set member not available</td>
</tr>
</tbody>
</table>
GQDSGA

Purpose

<table>
<thead>
<tr>
<th>Function</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQDSGA</td>
<td>To inquire dynamic modification of segment attributes.</td>
</tr>
</tbody>
</table>

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
GDDM-GKS functions

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 (GINS)

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
Function: To inquire default stroke device data.

GQDSK (wtype, devno, n, mldr, errind, dbufsk, ol, pet, earea, buflen, ldr, datrec)

APL code 1501
GKS RCP code X`3800D900` (939579648)
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
GDDM-GKS functions

2001 Output parameter size insufficient
2002 List element or set member not available

GQDSP

Purpose

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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wtype</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>errind</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>dcunit</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>rx</td>
<td>(returned by GDDM) (short floating point)</td>
</tr>
<tr>
<td>ry</td>
<td>(returned by GDDM) (short floating point)</td>
</tr>
<tr>
<td>lx</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>ly</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1388</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’3800C300’ (939574016)</td>
</tr>
</tbody>
</table>
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)

<table>
<thead>
<tr>
<th>Function:</th>
<th>To inquire default string device data.</th>
</tr>
</thead>
</table>

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.
GDDM-GKS functions

**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**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>2001</td>
<td>Output parameter size insufficient</td>
</tr>
<tr>
<td>2002</td>
<td>List element or set member not available</td>
</tr>
</tbody>
</table>

---

**GQDVL**

**Purpose**

<table>
<thead>
<tr>
<th>GQDVL</th>
<th>(wtype, devno, n, mldr, errind, dval, ol, pet, earea, loval, hival, ldr, datrec)</th>
</tr>
</thead>
</table>

| 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)*
  - The minimal and maximal values of the valuator.
- **hival** *(returned by GDDM) (short floating point)*
  - The number of elements used in the data record array.
- **ldr** *(returned by GDDM) (fullword integer)*
  - 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**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>wtype</code></td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td><code>errind</code></td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td><code>plbun</code></td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td><code>pmbun</code></td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td><code>txbun</code></td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td><code>fabun</code></td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td><code>parep</code></td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td><code>colrep</code></td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td><code>wktr</code></td>
<td>(fullword integer)</td>
</tr>
</tbody>
</table>

**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

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. GQECI 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 GQECI 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:
GDDM-GKS 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
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

<table>
<thead>
<tr>
<th>GQEFAI</th>
<th>(wkid, n, errind, ol, faind)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>APL code 1477</td>
</tr>
<tr>
<td></td>
<td>GKS RCP code X‘3800B300’ (939569920)</td>
</tr>
</tbody>
</table>

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.

erind (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.
GDDM-GKS functions

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
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.
**GDDM-GKS functions**

**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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>39</td>
<td>Specified workstation is neither of category OUTPUT nor of category OUTIN</td>
</tr>
</tbody>
</table>

---

### GQENTN

**Purpose**

<table>
<thead>
<tr>
<th>GQENTN</th>
<th>(n, errind, ol, nprio)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1462</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’3800A300’ (939565824)</td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>GQEPLI</th>
<th>(wkid, n, errind, ol, pli)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1470</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X‘3800AC00’ (939568128)</td>
</tr>
</tbody>
</table>

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

\textbf{wkid} (specified by user) (fullword integer)
The workstation identifier.
\textbf{n} (specified by user) (fullword integer)
The list element requested.
\textbf{errind} (returned by GDDM) (fullword integer)
The error indicator.
\textbf{ol} (returned by GDDM) (fullword integer)
Number of polyline bundle table entries.
\textbf{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 \textit{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \textit{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

\begin{itemize}
\item 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
\item 20 Specified workstation identifier is invalid
\item 25 Specified workstation is not open
\item 33 Specified workstation is of category MI
\item 35 Specified workstation is of category INPUT
\item 36 Specified workstation is Workstation Independent Segment Storage
\item 2002 List element or set member not available
\end{itemize}

GQEPMI

Purpose

\begin{tabular}{|c|c|}
\hline
\textbf{GQEPMI} & \textbf{(wkid, n, errind, ol, pmi)} \\
\hline
APL code & 1472 \\
GKS RCP code & X'3800AE00' (939568640) \\
\hline
\end{tabular}
**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**
GDDM-GKS functions

Purpose

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

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 GQF(AF 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
GDDM-GKS functions

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:

```fortran
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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>39</td>
<td>Specified workstation is neither of category OUTPUT nor of category OUTIN</td>
</tr>
<tr>
<td>2002</td>
<td>List element or set member not available</td>
</tr>
</tbody>
</table>

GQFAI

Purpose

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.
GDDM-GKS functions

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
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
GDDM-GKS functions

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
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
**GDDM-GKS functions**

**styll** *(returned by GDDM) (fullword integer)*
The fill area interior style index. This gives the hatch style or pattern according to `ints`.

**coll** *(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**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>36</td>
<td>Specified workstation is Workstation Independent Segment Storage</td>
</tr>
<tr>
<td>80</td>
<td>Fill area index is invalid</td>
</tr>
<tr>
<td>81</td>
<td>A representation for the specified fill area index has not been defined on this workstation</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>

**GQFASI**

**Purpose**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>GQFASI</td>
<td>(errind, styl)</td>
</tr>
<tr>
<td>APL code</td>
<td>1458</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38009F00' (939564800)</td>
</tr>
</tbody>
</table>

**Function:** To inquire fill area style index.

Inquiry function. Returns the current fill area style index from the GKS state list.
GDDM-GKS functions

Parameters

errind (returned by GDDM) (fullword integer)
The error indicator.

styl (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

\[
\begin{array}{ll}
\text{GQGDP} & (\text{wtype}, \text{gdp}, \text{errind}, \text{nbnd}, \text{bndl}) \\
\hline
\text{APL code} & 1421 \\
\text{GKS RCP code} & X'3800D300' (939578112)
\end{array}
\]

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

<table>
<thead>
<tr>
<th>GQIQOV</th>
<th>(errind, wkid, icl, idn)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1437</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’3800E300’ (939582208)</td>
</tr>
</tbody>
</table>

**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)

<table>
<thead>
<tr>
<th>APL code</th>
<th>1500</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X'3800BB00' (939571968)</td>
</tr>
</tbody>
</table>

**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 (GRESL) 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)*
  - The initial locator position in world coordinates.
- **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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
<tr>
<td>2001</td>
<td>Output parameter size insufficient</td>
</tr>
</tbody>
</table>

GQLI

Purpose
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.

<table>
<thead>
<tr>
<th>GQLI</th>
<th>(vtype, errind, nlcd, nskd, nvld, nchd, npkd, nstd)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1426</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’3800D700’ (939579136)</td>
</tr>
</tbody>
</table>

Function: To inquire number of available logical input devices.
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

<table>
<thead>
<tr>
<th>GQLN</th>
<th>(errind, ltype)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1447</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38009400' (939561984)</td>
</tr>
</tbody>
</table>
GDDM-GKS functions

**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**

<table>
<thead>
<tr>
<th>GQLVKS</th>
<th>(errind, level)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1499</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38008100' (939557120)</td>
</tr>
</tbody>
</table>

**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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GQLWSC</th>
<th>(errind, lwidth)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1448</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38009500' (939562240)</td>
</tr>
</tbody>
</table>
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 | '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

Chapter 5. GDDM-GKS functions
GDDM-GKS functions

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
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 \textit{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \textit{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
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
\textbf{errind} (returned by GDDM) (fullword integer)
The error indicator.

\textbf{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 \textit{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \textit{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
GDDM-GKS functions

GQNT

Purpose

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

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
GDDM-GKS functions

**Purpose**

**GQOPSG**  
(errind, sgna)

- **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)

- **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.

\( \text{errind} \) (returned by GDDM) (fullword integer)

The error indicator.

\( \text{ol} \) (returned by GDDM) (fullword integer)

The number of open workstations.

\( \text{wkid} \) (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 \( \text{errind} \) is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \( \text{errind} \) 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, \( \text{ol} \) 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)

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.
GDDM-GKS functions

\textbf{dimx} (specified by user) (fullword integer)

\textbf{dimy} (specified by user) (fullword integer)

The maximum dimensions of the pattern array \textit{colia}.

\textbf{errind} (returned by GDDM) (fullword integer)

The error indicator.

\textbf{dx} (returned by GDDM) (fullword integer)

\textbf{dy} (returned by GDDM) (fullword integer)

The dimensions of the pattern array \textit{colia}.

\textbf{colia} (returned by GDDM) (an array of fullword integers)

The pattern array (\textit{dimx} groups of \textit{dimy} elements).

\section*{Operating states}

WSOP, WSAC, SGOP

\section*{Related functions}

Set pattern representation (GSPAR)

\section*{Description}

If the inquired information is available, it is returned as output, and \textit{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \textit{errind} returns an error indicator.

\section*{Principal errors}

\begin{itemize}
  \item 7 \hspace{1em} GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP
  \item 20 \hspace{1em} Specified workstation identifier is invalid
  \item 25 \hspace{1em} Specified workstation is not open
  \item 33 \hspace{1em} Specified workstation is of category MI
  \item 35 \hspace{1em} Specified workstation is of category INPUT
  \item 36 \hspace{1em} Specified workstation is Workstation Independent Segment Storage
  \item 85 \hspace{1em} Specified pattern index is invalid
  \item 88 \hspace{1em} A representation for the specified pattern index has not been defined on this workstation
  \item 90 \hspace{1em} Interior style PATTERN is not supported on this workstation
\end{itemize}

\section*{GQPARF}

\section*{Purpose}

<table>
<thead>
<tr>
<th>GQPARF</th>
<th>(errinf, rfx, rfy)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1445</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38009200' (939561472)</td>
</tr>
</tbody>
</table>

\textbf{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.
GDDM-GKS functions

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
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:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(GHOLLO) Hollow</td>
</tr>
<tr>
<td>1</td>
<td>(GSOLID) Solid</td>
</tr>
<tr>
<td>2</td>
<td>(GPATTR) Pattern</td>
</tr>
<tr>
<td>3</td>
<td>(GHATCH) Hatch</td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>39</td>
<td>Specified workstation is neither of category OUTPUT nor of category OUTIN</td>
</tr>
<tr>
<td>80</td>
<td>Fill area index is invalid</td>
</tr>
<tr>
<td>82</td>
<td>A representation for the specified fill area index has not been predefined on this workstation</td>
</tr>
</tbody>
</table>

---

**GQPKID**

### Purpose

<table>
<thead>
<tr>
<th>GQPKID</th>
<th>(errind, pkid)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1446</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X*38009300* (939561728)</td>
</tr>
</tbody>
</table>

**Function:** To inquire pick identifier.

Inquiry function. Returns the current *pick* identifier from the GKS state list.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GQPKS</th>
<th>(wkid, pkdnr, type, mldr, errind, mode, esw, istat, isgna, ipkid, pet, earea, ldr, datrec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1488</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X‘3800BF00’ (939572992)</td>
</tr>
</tbody>
</table>

*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.
GDDM-GKS 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
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

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**

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 \textit{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \textit{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 \textit{n} of the list. You can obtain the entire list using the logic of the following example. Make an initial call to GQPLF with \textit{n} = 0. The output includes the total number of available line types (\textit{nlt}). Now make repeated calls to GQPLF until you have accumulated list elements equal to the total number of types.

Here is an example:

\begin{verbatim}
  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
\end{verbatim}

Principal errors

- \textbf{8} GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP
- \textbf{22} Specified workstation type is invalid
- \textbf{23} Specified workstation type does not exist
- \textbf{39} Specified workstation is neither of category OUTPUT nor of category OUTIN
- \textbf{2002} List element or set member not available

GQPLI

Purpose

<table>
<thead>
<tr>
<th>GQPLI</th>
<th>(errind, pli)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1427</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38008700’ (939558656)</td>
</tr>
</tbody>
</table>

\textit{Function}: To inquire polyline index.

Inquiry function. Returns the current \textit{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

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
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GQPMCI</th>
<th>(errind, coli)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1452</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38009900’ (939563264)</td>
</tr>
</tbody>
</table>
**GDDM-GKS functions**

**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**

<table>
<thead>
<tr>
<th>GQPMF</th>
<th><em>(wtype, n, errind, nmt, mt, nms, nomms, rmsmin, rmsmax, nppmi)</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1402</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'3800C800' (939575296)</td>
</tr>
</tbody>
</table>

**Function:** To inquire polymarker facilities.

Inquiry function. Returns values giving the *polymarker* facilities available for a given workstation type, from the workstation description table.
GDDM-GKS functions

Parameters

\textbf{wtype (specified by user) (fullword integer)}

The workstation type.

\textbf{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 \textit{nmt} but \textit{mt} is undefined.

\textbf{errind (returned by GDDM) (fullword integer)}

The error indicator.

\textbf{nmt (returned by GDDM) (fullword integer)}

The number of available marker types (5 ... n).

\textbf{mt (returned by GDDM) (fullword integer)}

Element \textit{n} of the list of available marker types.

\textbf{nms (returned by GDDM) (fullword integer)}

The number of available marker sizes (0 ... n).

\textbf{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.

\textbf{rmsmin (returned by GDDM) (short floating point)}

\textbf{rmsmax (returned by GDDM) (short floating point)}

The range of marker sizes available in device coordinates.

\textbf{nppmi (returned by GDDM) (fullword integer)}

The number of predefined polymarker indexes.

Operating states

\textbf{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 \textit{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \textit{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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>39</td>
<td>Specified workstation is neither of category OUTPUT nor of category OUTIN</td>
</tr>
<tr>
<td>2002</td>
<td>List element or set member not available</td>
</tr>
</tbody>
</table>
GQPMI

Purpose

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)

Chapter 5. GDDM-GKS functions 219
**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
GQPPAR

Purpose

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)*
  The maximum pattern array dimensions.
- **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)*
  The dimensions of the pattern array returned in *parray*.
- **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)
GDDM-GKS functions

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

| 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.
- Intype (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.
colι (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
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.
**GDDM-GKS functions**

**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
GDDM-GKS functions

Purpose

GQPX (wkid, px, py, errind, coli)

<table>
<thead>
<tr>
<th>APL code</th>
<th>1436</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X’3800E200’ (939581952)</td>
</tr>
</tbody>
</table>

**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

**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)*
  - The upper left corner in world coordinates.
- **py** *(specified by user) (short floating point)*
  - The upper left corner in world coordinates.
- **dimx** *(specified by user) (fullword integer)*
  - The dimension of the color index array colia.
- **dimy** *(specified by user) (fullword integer)*
  - The dimension of the color index array colia.
- **isc** *(specified by user) (fullword integer)*
  - The start column within the array.
- **isr** *(specified by user) (fullword integer)*
  - The start row within the array.
- **dx** *(specified by user) (fullword integer)*
  - The size of the requested pixel array.
- **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
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GQPXAD</th>
<th>(wkid, px, py, qx, qy, errind, n, m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1434</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X‘3800E000’ (939581440)</td>
</tr>
</tbody>
</table>

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)*
The dimensions of the pixel array.

**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**

<table>
<thead>
<tr>
<th>GQSGA</th>
<th>(sgna, errind, segtm, vis, high, sgpr, det)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1432</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X‘3800DF00’ (939581184)</td>
</tr>
</tbody>
</table>

**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:

\[
egin{align*}
M(1) & & M(3) & & M(5) \\
M(2) & & M(4) & & M(6)
\end{align*}
\]

*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

<table>
<thead>
<tr>
<th>GQSGP</th>
<th>(wtype, errind, nsg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1424</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'3800D500' (939578624)</td>
</tr>
</tbody>
</table>

**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**

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>39</td>
<td>Specified workstation is neither of category OUTPUT nor of category OUTIN</td>
</tr>
</tbody>
</table>
**GQSGUS**

**Purpose**

The purpose of GQSGUS is to inquire set member of segment names in use. It is an inquiry function that returns a segment name currently in use, from the list of segments in the GKS state list.

**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

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

\textbf{wkid} (specified by user) (fullword integer)

The workstation identifier.

\textbf{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 \textit{ol} but \textit{sgna} is undefined.

\textbf{errind} (returned by GDDM) (fullword integer)

The error indicator.

\textbf{ol} (returned by GDDM) (fullword integer)

The number of segments at the workstation (0 ... \textit{n}).

\textbf{sgna} (returned by GDDM) (fullword integer)

Member \textit{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 \textit{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \textit{errind} returns an error indicator.

If \textit{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, \textit{ol} is returned as 0.
GDDM-GKS functions

Principal errors

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>2002</td>
<td>List element or set member not available</td>
</tr>
</tbody>
</table>

GQSIM

Purpose

<table>
<thead>
<tr>
<th>GQSIM</th>
<th>(errind, flag)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

APL code 1466

GKS RCP code 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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
</tbody>
</table>

GQSKS

Purpose

GQSKS (wkid, skdnr, type, n, mldr, errind, mode, esw, itnr, np, pxa, pya, pet, earea, buflen, ldr, datrec)

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

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.
GDDM-GKS functions

**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)*
- The initial points of the stroke in world coordinates.

**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.

**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.
GDDM-GKS functions

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
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.
**GDDM-GKS functions**

**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

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

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

Set text color index (GSTXCI)

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
**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.

```fortran
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
   WRITE (*, 52) prec
50 FORMAT (' font', I6)
52 FORMAT (' prec',I6)
55 CONTINUE
```

### Principal errors

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>22</td>
<td>Specified workstation type is invalid</td>
</tr>
<tr>
<td>23</td>
<td>Specified workstation type does not exist</td>
</tr>
<tr>
<td>39</td>
<td>Specified workstation is neither of category OUTPUT nor of category OUTIN</td>
</tr>
<tr>
<td>2002</td>
<td>List element or set member not available</td>
</tr>
</tbody>
</table>

### GQTXFP

**Purpose**

<table>
<thead>
<tr>
<th>GQTXFP</th>
<th>(errind, font, prec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1453</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38009A00' (939563520)</td>
</tr>
</tbody>
</table>

**Function:** To inquire text font and precision.

Inquiry function. Returns the current text font and precision from the GKS state list.
GDDM-GKS functions

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:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(GSTRP) String precision</td>
</tr>
<tr>
<td>1</td>
<td>(GCHARP) Character precision</td>
</tr>
<tr>
<td>2</td>
<td>(GSTRKP) Stroke precision</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
</tbody>
</table>

GQTXI

Purpose

<table>
<thead>
<tr>
<th>GQTXI</th>
<th>(errind, txi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1430</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38008900’ (939559168)</td>
</tr>
</tbody>
</table>

*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
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
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
</tbody>
</table>

GQTXR

Purpose

**GQTXR**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wkid</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>txi</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>type</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>errind</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>font</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>prec</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>chxp</td>
<td>(fullword integer)</td>
</tr>
<tr>
<td>chsp</td>
<td>(fullword integer)</td>
</tr>
<tr>
<td>coli</td>
<td>(fullword integer)</td>
</tr>
</tbody>
</table>

**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.)
GDDM-GKS functions

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.

erind *(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

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**
GDDM-GKS functions

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, cp, cpy, txexpx, txexpy)

<table>
<thead>
<tr>
<th>APL code</th>
<th>1476</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X'3800B200' (939569664)</td>
</tr>
</tbody>
</table>

*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)*
  The text position in world coordinates.
- **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.
- **cp** *(returned by GDDM) (short floating point)*
  The concatenation point in world coordinates.
- **str** *(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

<table>
<thead>
<tr>
<th>GQVLS</th>
<th>(wkid, vldnr, midr, errind, mode, esw, ival, pet, earea, loval, hival, ldr, datrec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1486</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’3800BD00’ (939572480)</td>
</tr>
</tbody>
</table>

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

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

Function: To inquire workstation category.

Inquiry function. Returns the category of the specified type of workstation, from the workstation description table.

Parameters

- **wtype** \(\text{(specified by user) (fullword integer)}\)
  The workstation type.
- **errind** \(\text{(returned by GDDM) (fullword integer)}\)
  The error indicator.
- **wkcat** \(\text{(returned by GDDM) (fullword integer)}\)
  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 \(\text{GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP}\)
- 22 \(\text{Specified workstation type is invalid}\)
- 23 \(\text{Specified workstation type does not exist}\)
GQWKCL

Purpose

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.

erind *(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

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
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>36</td>
<td>Specified workstation is Workstation Independent Segment Storage</td>
</tr>
</tbody>
</table>

GQWKM

Purpose

**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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>errind</em></td>
<td><em>returned by GDDM</em> (fullword integer) The error indicator.</td>
</tr>
<tr>
<td><em>mxopwk</em></td>
<td><em>returned by GDDM</em> (fullword integer) The maximum number of simultaneously open workstations.</td>
</tr>
<tr>
<td><em>mxacwk</em></td>
<td><em>returned by GDDM</em> (fullword integer) The maximum number of simultaneously active workstations.</td>
</tr>
<tr>
<td><em>mxwkas</em></td>
<td><em>returned by GDDM</em> (fullword integer) The maximum number of workstations associated with a segment.</td>
</tr>
</tbody>
</table>

Operating states

GKOP, WSOP, WSAC, SGOP
Related functions
None

Description
If the inquired information is available, it is returned as output, and \textit{errind} is returned as 0 (zero). If the inquired information is not available, all output is invalid, and \textit{errind} returns an error indicator.

Principal errors

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
</tbody>
</table>

GQWKS

Purpose

\textbf{Function:} To inquire workstation state.

Inquiry function. Returns the activity state of a workstation, from the workstation state list.

Parameters

- \textit{wkid} (specified by user) (fullword integer)
  The workstation identifier.
- \textit{errind} (returned by GDDM) (fullword integer)
  The error indicator.
- \textit{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)
GDDM-GKS functions

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

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 (GSKWN), Set workstation viewport (GSKVP)

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
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.
GDDM-GKS functions

**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**

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>34</td>
<td>Specified workstation is not of category MI</td>
</tr>
<tr>
<td>162</td>
<td>No item is left in GKS Metafile input</td>
</tr>
<tr>
<td>163</td>
<td>Metafile item is invalid</td>
</tr>
<tr>
<td>165</td>
<td>Content of item data record is invalid for the specified item type</td>
</tr>
<tr>
<td>166</td>
<td>Maximum item data record length is invalid</td>
</tr>
<tr>
<td>2001</td>
<td>Output parameter size insufficient</td>
</tr>
</tbody>
</table>

---

**GRENSG**

**Purpose**

<table>
<thead>
<tr>
<th>GRENSG</th>
<th>(old, new)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1359</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38003A00' (939538944)</td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>GRQCH</th>
<th>(wkid, chdnr, stat, chnr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1380</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>'X'38005400' (939545600)</td>
</tr>
</tbody>
</table>

*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
GDDM-GKS functions

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 **chnr**.

**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 **chnr** 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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>141</td>
<td>Input device is not in REQUEST mode</td>
</tr>
</tbody>
</table>

### GRQLC

**Purpose**

<table>
<thead>
<tr>
<th>Function:</th>
<th>To request locator.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input function. Obtains <em>request</em> mode input from a <em>locator</em> input device at a workstation.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GRQLC</th>
<th>(wkid, lcdnr, stat, tnr, px, py)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1377</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>'X' '38005100' (939544832)</td>
</tr>
</tbody>
</table>
GDDM-GKS functions

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)*
- The locator position in world coordinates.

`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

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>141</td>
<td>Input device is not in REQUEST mode</td>
</tr>
</tbody>
</table>
GRQPK

Purpose

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.
GDDM-GKS functions

The pick identifier returned is the one associated with the primitive within the segment that was picked.

Principal errors

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>37</td>
<td>Specified workstation is not of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
</tbody>
</table>

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 pxa, pya.
- **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)*
  The world coordinates of the points returned. The number of elements used is given by np.
- **pya** *(returned by GDDM)* *(array of short floating-point numbers)*
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 description 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

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

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

<table>
<thead>
<tr>
<th>GRQVL</th>
<th>(wkid, vldnr, stat, val)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1379</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38005300’ (939545344)</td>
</tr>
</tbody>
</table>
**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**

**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**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>36</td>
<td>Specified workstation is Workstation Independent Segment Storage</td>
</tr>
</tbody>
</table>

**GSASF**
Purpose

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

<table>
<thead>
<tr>
<th>GSCHH</th>
<th>(chh)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1331</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X‘38001F00’ (939532032)</td>
</tr>
</tbody>
</table>

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

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.
GDDM-GKS 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
143 EVENT and SAMPLE input mode are not available at this level of GKS
2000 Enumeration type out of range

GSCHSP

Purpose

<table>
<thead>
<tr>
<th>GSCHSP</th>
<th>(chsp)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1329</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38001D00' (939531520)</td>
</tr>
</tbody>
</table>

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**

<table>
<thead>
<tr>
<th>GSCHUP</th>
<th>(chux, chuy)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1332</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38002000' (939532288)</td>
</tr>
</tbody>
</table>

**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

<table>
<thead>
<tr>
<th>GSCHXP</th>
<th>(chxp)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1328</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38001C00’ (939531264)</td>
</tr>
</tbody>
</table>

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

Function: To set clipping indicator.

Transformation function. Sets the clipping indicator for clipping to the normalization
transformation viewport.

Parameters

\( \text{clsw} \) (specified by user) (fullword integer)

The clipping indicator. The possible values are:

0 (GNCLIP) No clipping
1 (GCLIP) Clipping
GDDM-GKS functions

Operating states
GKOP, WSOP, WSAC, SGOP

Related functions
Select normalization transformation (GSELT), 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
Function: To set color representation.

APL code 1349
GKS RCP code X‘38003000’ (939536384)
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)
GDDM-GKS functions

GSDS

Purpose

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 (GSSSGP).
GDDM-GKS 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
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)

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sgna (specified by user)</td>
<td>fullword integer</td>
</tr>
<tr>
<td>det (specified by user)</td>
<td>fullword integer</td>
</tr>
</tbody>
</table>

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

Function: To select normalization transformation.
Transformation function. Selects the current normalization transformation number.

Parameters

\textbf{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 \textit{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.
GDDM-GKS functions

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

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

<table>
<thead>
<tr>
<th>GSFAI</th>
<th>(index)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1335</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X‘38002300’ (939533056)</td>
</tr>
</tbody>
</table>

*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

<table>
<thead>
<tr>
<th>Function:</th>
<th>To set fill area interior style.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute function. Selects the current fill area interior style for fill area output primitives.</td>
<td></td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>ints (specified by user) (fullword integer)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The interior fill area style. The possible values are:</td>
</tr>
<tr>
<td>0  (GHOLLO) Hollow</td>
</tr>
<tr>
<td>1  (GSOLID) Solid</td>
</tr>
<tr>
<td>2  (GPATTR) Pattern</td>
</tr>
<tr>
<td>3  (GHATCH) Hatch</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>GSFAR</th>
<th>(wkid, fai, ints, styli, coli)</th>
</tr>
</thead>
</table>

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
GDDM-GKS functions

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.
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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>36</td>
<td>Specified workstation is Workstation Independent Segment Storage</td>
</tr>
<tr>
<td>80</td>
<td>Fill area index is invalid</td>
</tr>
<tr>
<td>83</td>
<td>Specified fill area interior style is not supported on this workstation</td>
</tr>
<tr>
<td>85</td>
<td>Specified pattern index is invalid</td>
</tr>
<tr>
<td>86</td>
<td>Specified hatch style is not supported on this workstation</td>
</tr>
<tr>
<td>93</td>
<td>Color index is invalid</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>

### GSFASI

**Purpose**

<table>
<thead>
<tr>
<th>GSFASI</th>
<th>(styli)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Function:** To set fill area style index.

Attribute function. Selects the current *fill area* style index for fill area output primitives.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>styli</em></td>
<td>(specified by user) (fullword integer)</td>
<td>The fill area style index.</td>
</tr>
</tbody>
</table>
GDDM-GKS functions

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 \textit{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 \textit{individual}.

For fill areas that are displayed with interior styles \textit{hollow} and \textit{solid}, the index is not used.

For fill areas displayed with interior style \textit{pattern}, the index \textit{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 \textit{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.
GDDM-GKS functions

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

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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>

### GSLN

**Purpose**

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:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(GLSOLI) Solid</td>
</tr>
<tr>
<td>2</td>
<td>(GLDASH) Dash</td>
</tr>
<tr>
<td>3</td>
<td>(GLDOT) Dotted</td>
</tr>
<tr>
<td>4</td>
<td>(GLDASD) Dash-dotted</td>
</tr>
</tbody>
</table>

### Operating states

GKOP, WSOP, WSAC, SGOP

### Related functions

Polyline (GPL), Set aspect source flags (GSASF), Inquire polyline facilities (GQPLF), Inquire linetype (GQLN)
GDDM-GKS functions

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 \textit{ltype}. This value is used for the display of subsequent polyline primitives created when the current \textit{linetype} aspect source flag (ASF) is \textit{individual}.

The line types specified by negative values of \textit{ltype} are implementation-dependent and availability is workstation-dependent. The following table shows the values for \textit{ltype} supported by GDDM-GKS.

<table>
<thead>
<tr>
<th>Value</th>
<th>Line type</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>Solid line (same as 1)</td>
</tr>
<tr>
<td>-3</td>
<td>Dash-double-dot line</td>
</tr>
<tr>
<td>-2</td>
<td>Long-dashed line</td>
</tr>
<tr>
<td>-1</td>
<td>Double-dotted line</td>
</tr>
<tr>
<td>1</td>
<td>Solid</td>
</tr>
<tr>
<td>2</td>
<td>Dashed line (short dashes)</td>
</tr>
<tr>
<td>3</td>
<td>Dotted line</td>
</tr>
<tr>
<td>4</td>
<td>dashed-dotted line</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>GSLWSC</th>
<th>(lwidth)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1319</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X ‘38001400’ (939529216)</td>
</tr>
</tbody>
</table>

Function:  To set linewidth scale factor.

Attribute function. Sets the current line width scale factor for polyline output primitives.

Parameters

\textbf{lwidth} \textit{(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 \textit{lwidth}. This value is used for the display of subsequent polyline primitives created when the current linewidth scale factor aspect source flag (ASF) is \textit{individual}. (If the ASF is \textit{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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>65</td>
<td>Linewidth scale factor is less than zero</td>
</tr>
</tbody>
</table>

GSMCH

Purpose

<table>
<thead>
<tr>
<th>GSMCH</th>
<th>(wkid, chdnr, stat, chnr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1385</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38005A00' (939547136)</td>
</tr>
</tbody>
</table>
**GDDM-GKS functions**

*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.

- **chnnr** *(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
GSDM-GKS functions

GSMK

Purpose

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.

<table>
<thead>
<tr>
<th>APL code</th>
<th>GKS RCP code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1322</td>
<td>X’38001700’ (939529984)</td>
</tr>
</tbody>
</table>
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
   \[ \cdot \]

2. plus
   \[ \pm \]

3. asterisk
   \[ \star \star \]

4. circle
   \[ \bigcirc \]

5. diagonal cross
   \[ \times \times \]

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

<table>
<thead>
<tr>
<th>GSMKSC</th>
<th>(mszsf)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1323</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38001800' (939530240)</td>
</tr>
</tbody>
</table>
**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.

```
  .  +  *  o  x
  .  +  *  o  x
  .  +  *  o  x
  .  +  *  o  x
  .  +  *  o  x
```
GDDM-GKS functions

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

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)
The locator position in world coordinates.
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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>142</td>
<td>Input device is not in SAMPLE mode</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
</tbody>
</table>

GSMPK

Purpose

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:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>OK</td>
</tr>
<tr>
<td>1</td>
<td>No pick</td>
</tr>
</tbody>
</table>

**sgna** *(returned by GDDM) (fullword integer)*
The segment name.

**pkid** *(returned by GDDM) (fullword integer)*
The pick identifier.

Operating states

WSOP, WSAC, SGOP
GDDM-GKS functions

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.)
GDDM-GKS functions

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
GDDM-GKS functions

### GSMSTS

<table>
<thead>
<tr>
<th>(wkid, stdnr, mstr, lostr, str)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code: 1389</td>
</tr>
<tr>
<td>GKS RCP code: X'38005C00' (939547648)</td>
</tr>
</tbody>
</table>

#### 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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
</tbody>
</table>
GDDM-GKS functions

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

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 (GSVLNM)

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.
GDDM-GKS 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
142 Input device is not in SAMPLE mode
143 EVENT and SAMPLE input mode are not available at this level of GKS

GSPA

Purpose

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**

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)**
  The dimensions of the color index array colia.
- **dimy (specified by user) (fullword integer)**
  The dimensions of the color index array colia.
- **isc (specified by user) (fullword integer)**
  The start column and start row of the pattern array in the color index array colia.
- **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)**
  The number of columns and rows of the pattern array.
- **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

<table>
<thead>
<tr>
<th>Error code</th>
<th>Error message</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>36</td>
<td>Specified workstation is Workstation Independent Segment Storage</td>
</tr>
<tr>
<td>85</td>
<td>Specified pattern index is invalid</td>
</tr>
<tr>
<td>90</td>
<td>Interior style PATTERN is not supported on this workstation</td>
</tr>
<tr>
<td>91</td>
<td>Dimensions of color array are invalid</td>
</tr>
<tr>
<td>93</td>
<td>Color index is invalid</td>
</tr>
</tbody>
</table>
GDDM-GKS functions

GSPARF

Purpose

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**

| 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
GDDM-GKS functions

GSPKM

Purpose

**GSPKM**  
(wkid, pkdnr, mode, esw)

<table>
<thead>
<tr>
<th>APL code</th>
<th>1375</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X'38004F00' (939544320)</td>
</tr>
</tbody>
</table>

**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

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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GSPLI</th>
<th>(pli)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1317</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X ‘38001200’ (939528704)</td>
</tr>
</tbody>
</table>

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**

<table>
<thead>
<tr>
<th>GSPLR</th>
<th>(wkid, pli, ltype, lwidth, coli)</th>
</tr>
</thead>
</table>

**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.
GDDM-GKS functions

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 \textit{pli} at workstation \textit{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 \textit{ltype} are implementation-dependent and availability is workstation-dependent. The following table shows the values for \textit{ltype} supported by GDDM-GKS.

<table>
<thead>
<tr>
<th>Value</th>
<th>Line type</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>Solid line (same as 1)</td>
</tr>
<tr>
<td>-3</td>
<td>Dash-double-dot line</td>
</tr>
<tr>
<td>-2</td>
<td>Long-dashed line</td>
</tr>
<tr>
<td>-1</td>
<td>Double-dotted line</td>
</tr>
<tr>
<td>1</td>
<td>Solid</td>
</tr>
<tr>
<td>2</td>
<td>Dashed line (short dashes)</td>
</tr>
<tr>
<td>3</td>
<td>Dotted line</td>
</tr>
<tr>
<td>4</td>
<td>dash-dotted line</td>
</tr>
</tbody>
</table>

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
GSPMCI

Purpose

GSPMCI

| 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

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>92</td>
<td>Color index is less than zero</td>
</tr>
</tbody>
</table>
GDDM-GKS functions

GSPMI

Purpose

<table>
<thead>
<tr>
<th>Function:</th>
<th>To set polymarker index.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute function. Selects a bundle of polymarker attributes to be used for subsequent polymarker output primitives.</td>
<td></td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>pmi (specified by user) (fullword integer)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The polymarker index. A number that represents a bundle of polymarker output attributes at the workstation.</td>
</tr>
</tbody>
</table>

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.
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GSPMR</th>
<th>(wkid, pmi, mtype, mkszsf, coli)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1345</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X’38002C00’ (939535360)</td>
</tr>
</tbody>
</table>

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

mkszsf (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
GDDM-GKS functions

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.

<table>
<thead>
<tr>
<th>Value</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>-6</td>
<td>Shaded box</td>
</tr>
<tr>
<td>-5</td>
<td>Shaded diamond</td>
</tr>
<tr>
<td>-4</td>
<td>Asterisk (6 points)</td>
</tr>
<tr>
<td>-3</td>
<td>Hollow box</td>
</tr>
<tr>
<td>-2</td>
<td>Hollow diamond</td>
</tr>
<tr>
<td>-1</td>
<td>Scalable dot</td>
</tr>
<tr>
<td>1</td>
<td>Dot</td>
</tr>
<tr>
<td>2</td>
<td>Plus</td>
</tr>
<tr>
<td>3</td>
<td>Asterisk (8 points)</td>
</tr>
<tr>
<td>4</td>
<td>Circle</td>
</tr>
<tr>
<td>5</td>
<td>Diagonal cross</td>
</tr>
</tbody>
</table>

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
GSSGP

Purpose

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

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
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>120</td>
<td>Specified segment name is invalid</td>
</tr>
<tr>
<td>122</td>
<td>Specified segment does not exist</td>
</tr>
</tbody>
</table>

GSSKM

Purpose

<table>
<thead>
<tr>
<th>GSSKM</th>
<th>(wkid, skdnr, mode, esw)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

APL code 1372

GKS RCP code '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

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>

GSSTM

Purpose

<table>
<thead>
<tr>
<th>Function:</th>
<th>To set string mode.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input function.</td>
<td>Sets the operating mode and echo switch for a string device at a workstation. (See the section “GDDM-GKS restrictions” below.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GSSTM</th>
<th>(wkid, stdnr, mode, esw)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1376</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38005000' (939544576)</td>
</tr>
</tbody>
</table>
GDDM-GKS functions

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

**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

---

Chapter 5. GDDM-GKS functions 331
GDDM-GKS functions

*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:

```
   input   input   input

   input   input   input

   input   input   input
```

**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**

<table>
<thead>
<tr>
<th>GSTXCI</th>
<th>coli</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1330</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X'38001E00' (939531776)</td>
</tr>
</tbody>
</table>

**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.
GDDM-GKS functions

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 \textit{coli}.
This value is used for the display of subsequent text output primitives when the text
color index ASF is \textit{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 \textit{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

<table>
<thead>
<tr>
<th>GSTXFP</th>
<th>(font, prec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1326</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X '38001B00' (939531008)</td>
</tr>
</tbody>
</table>

\textit{Function}: To set text font and precision.

Attribute function. Sets the current text font and precision for text output primitives.

Parameters

\textit{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.

\textit{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.

<table>
<thead>
<tr>
<th>Font number</th>
<th>Precisions provided</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>All</td>
<td>Default font</td>
</tr>
<tr>
<td>-1</td>
<td>Character, stroke</td>
<td>Italic characters</td>
</tr>
<tr>
<td></td>
<td>GDDM national language symbol sets:</td>
<td></td>
</tr>
</tbody>
</table>
### GDDM-GKS functions

<table>
<thead>
<tr>
<th>Font number</th>
<th>Precisions provided</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2</td>
<td>Stroke</td>
<td>Brazilian</td>
</tr>
<tr>
<td>-3</td>
<td>Stroke</td>
<td>Danish</td>
</tr>
<tr>
<td>-4</td>
<td>Stroke</td>
<td>English</td>
</tr>
<tr>
<td>-5</td>
<td>Stroke</td>
<td>French</td>
</tr>
<tr>
<td>-6</td>
<td>Stroke</td>
<td>German</td>
</tr>
<tr>
<td>-7</td>
<td>Stroke</td>
<td>Italian</td>
</tr>
<tr>
<td>-8</td>
<td>Stroke</td>
<td>Japanese</td>
</tr>
<tr>
<td>-9</td>
<td>Stroke</td>
<td>Norwegian</td>
</tr>
<tr>
<td>-10</td>
<td>Stroke</td>
<td>Spanish</td>
</tr>
<tr>
<td>-11</td>
<td>Stroke</td>
<td>Swedish</td>
</tr>
</tbody>
</table>

#### Proportionally-spaced symbol sets:

<table>
<thead>
<tr>
<th>Font number</th>
<th>Precisions provided</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-20</td>
<td>Stroke</td>
<td>Area Filled Roman</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-21</td>
<td>Stroke</td>
<td>Outline Roman</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-22</td>
<td>Stroke</td>
<td>Complex Italic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-23</td>
<td>Stroke</td>
<td>Complex Roman</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-24</td>
<td>Stroke</td>
<td>Complex Script</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-25</td>
<td>Stroke</td>
<td>Duplex Roman</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-26</td>
<td>Stroke</td>
<td>Simplex Roman</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-27</td>
<td>Stroke</td>
<td>Triplex Italic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-28</td>
<td>Stroke</td>
<td>Triplex Roman</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-29</td>
<td>Stroke</td>
<td>Gothic English</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-30</td>
<td>Stroke</td>
<td>Gothic German</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-31</td>
<td>Stroke</td>
<td>Gothic Italian</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Principal</td>
</tr>
<tr>
<td>-40</td>
<td>Stroke</td>
<td>Shadow</td>
</tr>
<tr>
<td>-41</td>
<td>Stroke</td>
<td>Thin Filled</td>
</tr>
<tr>
<td>-42</td>
<td>Stroke</td>
<td>Thin Outline</td>
</tr>
<tr>
<td>-43</td>
<td>Stroke</td>
<td>Thick Square Filled</td>
</tr>
<tr>
<td>-44</td>
<td>Stroke</td>
<td>Thick Square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Outlined</td>
</tr>
<tr>
<td>-45</td>
<td>Stroke</td>
<td>Thick Round Filled</td>
</tr>
<tr>
<td>-46</td>
<td>Stroke</td>
<td>Thick Round</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Outlined</td>
</tr>
<tr>
<td>-47</td>
<td>Stroke</td>
<td>Moderne</td>
</tr>
</tbody>
</table>

#### Double-byte character symbol sets:

<table>
<thead>
<tr>
<th>Font number</th>
<th>Precisions provided</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-65</td>
<td>Character, Stroke</td>
<td>DBCS Kanji symbols</td>
</tr>
</tbody>
</table>

### Principal errors

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>75</td>
<td>Text font is equal to zero</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>
GSTXI

Purpose

GSTXI

<table>
<thead>
<tr>
<th>Function:</th>
<th>To set text index.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute function. Selects a bundle of text attributes to be used for text output primitives.</td>
<td></td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>txi (specified by user) (fullword integer)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The text index.</td>
</tr>
</tbody>
</table>

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.
GDDM-GKS functions

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

| 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**

| 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
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>33</td>
<td>Specified workstation is of category MI</td>
</tr>
<tr>
<td>35</td>
<td>Specified workstation is of category INPUT</td>
</tr>
<tr>
<td>36</td>
<td>Specified workstation is Workstation Independent Segment Storage</td>
</tr>
<tr>
<td>72</td>
<td>Text index is invalid</td>
</tr>
<tr>
<td>75</td>
<td>Text font is equal to zero</td>
</tr>
<tr>
<td>76</td>
<td>Requested text font is not supported for the specified precision on this workstation</td>
</tr>
<tr>
<td>77</td>
<td>Character expansion factor is less than or equal to zero</td>
</tr>
<tr>
<td>93</td>
<td>Color index is invalid</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>
GSVIS

Purpose

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 (GQDSCGA)

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

<table>
<thead>
<tr>
<th>GSVLM</th>
<th>(wkid, vldnr, mode, esw)</th>
</tr>
</thead>
</table>

**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**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>20</td>
<td>Specified workstation identifier is invalid</td>
</tr>
<tr>
<td>25</td>
<td>Specified workstation is not open</td>
</tr>
<tr>
<td>38</td>
<td>Specified workstation is neither of category INPUT nor of category OUTIN</td>
</tr>
<tr>
<td>140</td>
<td>Specified input device is not present on workstation</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>

**GSVP**

**Purpose**

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 functions

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

<table>
<thead>
<tr>
<th>GSVPIP</th>
<th>(tnr, rtnr, relpri)</th>
</tr>
</thead>
<tbody>
<tr>
<td>APL code</td>
<td>1352</td>
</tr>
<tr>
<td>GKS RCP code</td>
<td>X '38003300' (939537152)</td>
</tr>
</tbody>
</table>

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 \( rnr \). 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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>50</td>
<td>Transformation number is invalid</td>
</tr>
<tr>
<td>2000</td>
<td>Enumeration type out of range</td>
</tr>
</tbody>
</table>

GSWKVP

Purpose

<table>
<thead>
<tr>
<th>Function:</th>
<th>To set workstation viewport.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformation function. Defines the device coordinate viewport for a workstation.</td>
<td></td>
</tr>
</tbody>
</table>
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 (GSKWKN), 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

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

<table>
<thead>
<tr>
<th>Function:</th>
<th>To set window.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformation function. Defines the world coordinate window for a normalization transformation.</td>
<td></td>
</tr>
</tbody>
</table>

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 (GSVPPIP), 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
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>50</td>
<td>Transformation number is invalid</td>
</tr>
<tr>
<td>51</td>
<td>Rectangle definition is invalid</td>
</tr>
</tbody>
</table>

GTX

Purpose

| 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.) |
|---|---|
| GTX (px,py,chars) | }
GDDM-GKS functions

Parameters

\textbf{px} (specified by user) (short floating point)
\textbf{py} (specified by user) (short floating point)

The position of the string in world coordinates.

\textbf{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 \textit{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**

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)
- **lstr** (specified by user) (fullword integer)
- **chars** (specified by user) (character)

**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 \textit{lstr} is less than zero, GDDM-GKS will report error 2001.

**Principal errors**

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP</td>
</tr>
<tr>
<td>101</td>
<td>Invalid code in string</td>
</tr>
<tr>
<td>2001</td>
<td>Output parameter size insufficient</td>
</tr>
</tbody>
</table>

**GUREC**

**Purpose**

GUREC \((ldr,\text{datrec},iil,irl,isl,errind,il,ia,rl,ra,sl,lstr,str)\)

*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.
GDDM-GKS functions

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, il, il, isl, mstr, errind, il, ia, rl, ra, sl, lstr, str)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ldr</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>datrec</td>
<td>(specified by user) (array of 80-byte character tokens)</td>
</tr>
<tr>
<td>iil</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>irl</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>isl</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>mstr</td>
<td>(specified by user) (fullword integer)</td>
</tr>
<tr>
<td>errind</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>il</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>ia</td>
<td>(returned by GDDM) (an array of fullword integers)</td>
</tr>
<tr>
<td>rl</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>ra</td>
<td>(returned by GDDM) (array of short floating-point numbers)</td>
</tr>
<tr>
<td>sl</td>
<td>(returned by GDDM) (fullword integer)</td>
</tr>
<tr>
<td>lstr</td>
<td>(returned by GDDM) (an array of fullword integers)</td>
</tr>
<tr>
<td>str</td>
<td>(returned by GDDM) (character)</td>
</tr>
</tbody>
</table>

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)

<table>
<thead>
<tr>
<th>APL code</th>
<th>1311</th>
</tr>
</thead>
<tbody>
<tr>
<td>GKS RCP code</td>
<td>X'38000800' (939526144)</td>
</tr>
</tbody>
</table>

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).
GDDM-GKS functions

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

<table>
<thead>
<tr>
<th>GWAIT</th>
<th>(tout, wkid, icl, idnr)</th>
</tr>
</thead>
</table>

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.
lcl *(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

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>GKS not in proper state: GKS shall be in one of the states WSOP, WSAC, or SGOP</td>
</tr>
<tr>
<td>143</td>
<td>EVENT and SAMPLE input mode are not available at this level of GKS</td>
</tr>
<tr>
<td>147</td>
<td>Input queue has overflowed</td>
</tr>
<tr>
<td>151</td>
<td>Timeout is invalid</td>
</tr>
</tbody>
</table>
**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.
### Operating state

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating state value</td>
<td>—</td>
<td>GKCL</td>
<td>E</td>
<td>GKCL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GKOP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WSOP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>WSAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SGOP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### GKS description table

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level of GKS</td>
<td>—</td>
<td>0a, 0b, 0c, 1a, 1b, 1c, 2a, 2b, 2c</td>
<td>E</td>
<td>2b</td>
</tr>
<tr>
<td>Number of available workstation types</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>13</td>
</tr>
<tr>
<td>List of available workstation types</td>
<td>—</td>
<td>—</td>
<td>(n) N</td>
<td>1 through 13</td>
</tr>
<tr>
<td>Maximum number of simultaneously open workstations</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>Maximum number of simultaneously active workstations</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>Maximum number of workstations associated with a segment</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>Maximum normalization transformation number</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>10</td>
</tr>
</tbody>
</table>
## GKS data structures

### GKS state list

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set of open workstations</td>
<td>—</td>
<td>(n) N</td>
<td>Empty</td>
<td></td>
</tr>
<tr>
<td>Set of active workstations</td>
<td>—</td>
<td>(n) N</td>
<td>Empty</td>
<td></td>
</tr>
<tr>
<td>Current polyline index</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current linetype</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current linewidth scale factor</td>
<td>—</td>
<td>&gt;= 0</td>
<td>R</td>
<td>1.0</td>
</tr>
<tr>
<td>Current polyline color index</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current linetype ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current linewidth scale factor ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current polyline color index ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current polymarker index</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current marker type</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>3</td>
</tr>
<tr>
<td>Current marker size scale factor</td>
<td>—</td>
<td>&gt;= 0</td>
<td>R</td>
<td>1.0</td>
</tr>
<tr>
<td>Current polymarker color index</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current marker type ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current marker size scale factor ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current polymarker color index ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current text index</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current text font and precision</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>String</td>
<td>1, String</td>
</tr>
<tr>
<td>Current character expansion factor</td>
<td>—</td>
<td>&gt;0</td>
<td>R</td>
<td>0.0</td>
</tr>
<tr>
<td>Current character spacing</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>0.0</td>
</tr>
<tr>
<td>Current text color index</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current text font and precision ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current character expansion factor ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current character spacing ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current text color index ASF¹</td>
<td>—</td>
<td>Bundled Individual</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current character height</td>
<td>WC</td>
<td>&gt;0</td>
<td>R</td>
<td>0.01</td>
</tr>
<tr>
<td>Entry name</td>
<td>Coordinate system</td>
<td>Permitted values</td>
<td>Data type</td>
<td>Initial value</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>-------------------</td>
<td>------------------</td>
<td>-----------</td>
<td>---------------</td>
</tr>
<tr>
<td>Current character up vector</td>
<td>WC</td>
<td>—</td>
<td>(2) R</td>
<td>0,1</td>
</tr>
<tr>
<td>Current character width</td>
<td>WC</td>
<td>&gt;0</td>
<td>R</td>
<td>0.01</td>
</tr>
<tr>
<td>Current character base vector</td>
<td>WC</td>
<td>—</td>
<td>(2) R</td>
<td>0,1</td>
</tr>
<tr>
<td>Current text path</td>
<td>—</td>
<td>Right</td>
<td>E</td>
<td>Right</td>
</tr>
<tr>
<td>Current text alignment</td>
<td>—</td>
<td>Horizontal</td>
<td>(2) E</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Vertical</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Normal</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Left</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Center</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Right</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Normal</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Top</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cap</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Half</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Base</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bottom</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Current fill area index</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current fill area interior style</td>
<td>—</td>
<td>Hollow</td>
<td>E</td>
<td>Hollow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Solid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pattern</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hatch</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Current fill area style index</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current fill area color index</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Current fill area interior style ASF¹</td>
<td>—</td>
<td>Bundled</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current fill area style index ASF¹</td>
<td>—</td>
<td>Bundled</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current fill area color index ASF¹</td>
<td>—</td>
<td>Bundled</td>
<td>E</td>
<td>Individual</td>
</tr>
<tr>
<td>Current pattern width vector</td>
<td>WC</td>
<td>—</td>
<td>(2) R</td>
<td>1,0</td>
</tr>
<tr>
<td>Current pattern height vector</td>
<td>WC</td>
<td>—</td>
<td>(2) R</td>
<td>0,1</td>
</tr>
<tr>
<td>Current pattern reference point</td>
<td>WC</td>
<td>—</td>
<td>P</td>
<td>(0,0)</td>
</tr>
<tr>
<td>Current pick identifier</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>0</td>
</tr>
<tr>
<td>Current normalization transformation number</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>0</td>
</tr>
</tbody>
</table>
### GKS data structures

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normalization transformation number²</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>Entry number</td>
</tr>
<tr>
<td>Window²</td>
<td>WC</td>
<td>—</td>
<td>(4) R</td>
<td>0,1,0,1</td>
</tr>
<tr>
<td>Viewport²</td>
<td>NDC</td>
<td>—</td>
<td>(4) R</td>
<td>0,1,0,1</td>
</tr>
<tr>
<td>Clipping indicator</td>
<td>—</td>
<td>Clip No clip</td>
<td>E</td>
<td>Clip</td>
</tr>
<tr>
<td>Clipping rectangle</td>
<td>NDC</td>
<td>—</td>
<td>(4) R</td>
<td>0,1,0,1</td>
</tr>
<tr>
<td>Name of open segment</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>undef</td>
</tr>
<tr>
<td>Set of segment names in use</td>
<td>—</td>
<td>—</td>
<td>(n) N</td>
<td>Empty</td>
</tr>
<tr>
<td>Set of segment state lists</td>
<td>—</td>
<td>—</td>
<td>Empty</td>
<td></td>
</tr>
</tbody>
</table>

**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.
**GKS data structures**

**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

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workstation identifier</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>—</td>
</tr>
<tr>
<td>Connection identifier</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>—</td>
</tr>
<tr>
<td>Workstation type</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>—</td>
</tr>
</tbody>
</table>

**Note:** The above three entries are initialized by the Open workstation (GOPWK) function.

### Relevant for output, output/input, WISS, and metafile output workstations

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workstation state</td>
<td>—</td>
<td>Active</td>
<td>E</td>
<td>Inactive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inactive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set of stored segments for this workstation</td>
<td>—</td>
<td>—</td>
<td>(n) N</td>
<td>Empty</td>
</tr>
<tr>
<td>Set of stored groups for this workstation</td>
<td>—</td>
<td>—</td>
<td>(n) N</td>
<td>Empty</td>
</tr>
</tbody>
</table>

### Relevant for output, output/input, and metafile output workstations

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deferral mode¹</td>
<td>—</td>
<td>ASAP, BNIG, BNIL, ASTI</td>
<td>E</td>
<td>wdt</td>
</tr>
<tr>
<td>Implicit regeneration mode</td>
<td>—</td>
<td>Suppressed, Allowed</td>
<td>E</td>
<td>wdt</td>
</tr>
<tr>
<td>Display surface empty</td>
<td>—</td>
<td>Empty, Not empty</td>
<td>E</td>
<td>Empty</td>
</tr>
<tr>
<td>New frame action necessary at update</td>
<td>—</td>
<td>No, Yes</td>
<td>E</td>
<td>No</td>
</tr>
<tr>
<td>Number of polyline bundle table entries</td>
<td>—</td>
<td>5 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Polyline index²</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Linetype²</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Linewidth scale factor²</td>
<td>—</td>
<td>&gt;=0</td>
<td>R</td>
<td>wdt</td>
</tr>
<tr>
<td>Polylne color index²</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
</tbody>
</table>
### GKS data structures

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of polymarker bundle table entries</td>
<td>—</td>
<td>5 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Polymarker index(^3)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Marker type(^3)</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Marker size scale factor(^3)</td>
<td>—</td>
<td>&gt;=0</td>
<td>R</td>
<td>wdt</td>
</tr>
<tr>
<td>Polymarker color index(^3)</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Number of text bundle table entries</td>
<td>—</td>
<td>2 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Text index(^4)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Text font and precision(^4)</td>
<td>—</td>
<td>-n ... -1, 1 ... n String Character Stroke</td>
<td>I E</td>
<td>wdt</td>
</tr>
<tr>
<td>Character expansion factor(^4)</td>
<td>—</td>
<td>&gt;0</td>
<td>R</td>
<td>wdt</td>
</tr>
<tr>
<td>Character spacing(^4)</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>wdt</td>
</tr>
<tr>
<td>Text color index(^4)</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Number of fill area bundle table entries</td>
<td>—</td>
<td>5 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Fill area index(^5)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Fill area interior style(^5)</td>
<td>—</td>
<td>Hollow Solid Pattern Hatch</td>
<td>E</td>
<td>wdt</td>
</tr>
<tr>
<td>Fill area style index(^5)</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Fill area color index(^5)</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Number of pattern table entries</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Pattern index(^6)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Pattern array dimensions(^6)</td>
<td>—</td>
<td>1 ... n</td>
<td>(2) I</td>
<td>wdt</td>
</tr>
<tr>
<td>Pattern array(^6)</td>
<td>—</td>
<td>0 ... n</td>
<td>(n) I</td>
<td>wdt</td>
</tr>
<tr>
<td>Number of color table entries</td>
<td>—</td>
<td>2 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Color index(^7)</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Color (red, green, blue intensities)(^7)</td>
<td>—</td>
<td>[0,1]</td>
<td>(3) R</td>
<td>wdt</td>
</tr>
<tr>
<td>Workstation transformation update state</td>
<td>—</td>
<td>Not pending Pending</td>
<td>E</td>
<td>Not pending</td>
</tr>
<tr>
<td>Requested workstation window</td>
<td>NDC</td>
<td>—</td>
<td>(4) R</td>
<td>0,1,0,1</td>
</tr>
<tr>
<td>Current workstation window</td>
<td>NDC</td>
<td>—</td>
<td>(4) R</td>
<td>0,1,0,1</td>
</tr>
<tr>
<td>Requested workstation viewport</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>8</td>
</tr>
</tbody>
</table>
GKS data structures

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current workstation viewport</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>8</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locator device number&lt;sup&gt;1&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Operating mode&lt;sup&gt;1&lt;/sup&gt;</td>
<td>—</td>
<td>Request</td>
<td>E</td>
<td>Request</td>
</tr>
<tr>
<td>Echo switch&lt;sup&gt;1&lt;/sup&gt;</td>
<td>—</td>
<td>Echo No echo</td>
<td>E</td>
<td>Echo</td>
</tr>
<tr>
<td>Initial normalization transformation number&lt;sup&gt;1&lt;/sup&gt;</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>0</td>
</tr>
<tr>
<td>Initial locator position&lt;sup&gt;1&lt;/sup&gt;</td>
<td>WC</td>
<td>—</td>
<td>P</td>
<td>wdt</td>
</tr>
<tr>
<td>Prompt and echo type&lt;sup&gt;1&lt;/sup&gt;</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Echo area&lt;sup&gt;1&lt;/sup&gt;</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>wdt</td>
</tr>
<tr>
<td>Locator data record&lt;sup&gt;1&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>wdt</td>
</tr>
<tr>
<td>Stroke device number&lt;sup&gt;2&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Operating mode&lt;sup&gt;2&lt;/sup&gt;</td>
<td>—</td>
<td>Request</td>
<td>E</td>
<td>Request</td>
</tr>
<tr>
<td>Echo switch&lt;sup&gt;2&lt;/sup&gt;</td>
<td>—</td>
<td>Echo No echo</td>
<td>E</td>
<td>Echo</td>
</tr>
<tr>
<td>Initial normalization transformation number&lt;sup&gt;2&lt;/sup&gt;</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>undef</td>
</tr>
<tr>
<td>Initial number of points&lt;sup&gt;2&lt;/sup&gt;</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>0</td>
</tr>
<tr>
<td>Initial points in stroke&lt;sup&gt;2&lt;/sup&gt;</td>
<td>WC</td>
<td>—</td>
<td>(n) P</td>
<td>Empty</td>
</tr>
<tr>
<td>Prompt and echo type&lt;sup&gt;2&lt;/sup&gt;</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Echo area&lt;sup&gt;2&lt;/sup&gt;</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>wdt</td>
</tr>
<tr>
<td>Stroke data record&lt;sup&gt;2&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>wdt</td>
</tr>
<tr>
<td>Input buffer size&lt;sup&gt;2,3&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Valuator device number&lt;sup&gt;4&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Operating mode&lt;sup&gt;4&lt;/sup&gt;</td>
<td>—</td>
<td>Request</td>
<td>E</td>
<td>Request</td>
</tr>
<tr>
<td>Echo switch&lt;sup&gt;4&lt;/sup&gt;</td>
<td>—</td>
<td>Echo No echo</td>
<td>E</td>
<td>Echo</td>
</tr>
<tr>
<td>Initial value&lt;sup&gt;4&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>wdt</td>
</tr>
<tr>
<td>Prompt and echo type&lt;sup&gt;4&lt;/sup&gt;</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Echo area&lt;sup&gt;4&lt;/sup&gt;</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>wdt</td>
</tr>
<tr>
<td>Valuator data record&lt;sup&gt;4&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>wdt</td>
</tr>
<tr>
<td>Low value&lt;sup&gt;4,5&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>wdt</td>
</tr>
<tr>
<td>High value&lt;sup&gt;4,5&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>wdt</td>
</tr>
<tr>
<td>Choice device number&lt;sup&gt;6&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Operating mode&lt;sup&gt;6&lt;/sup&gt;</td>
<td>—</td>
<td>Request</td>
<td>E</td>
<td>Request</td>
</tr>
<tr>
<td>Echo switch&lt;sup&gt;6&lt;/sup&gt;</td>
<td>—</td>
<td>Echo No echo</td>
<td>E</td>
<td>Echo</td>
</tr>
<tr>
<td>Initial choice number&lt;sup&gt;6&lt;/sup&gt;</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>undef</td>
</tr>
<tr>
<td>Prompt and echo type&lt;sup&gt;6&lt;/sup&gt;</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Echo area&lt;sup&gt;6&lt;/sup&gt;</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>wdt</td>
</tr>
<tr>
<td>Choice data record&lt;sup&gt;6&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>wdt</td>
</tr>
</tbody>
</table>
GKS data structures

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pick device number&lt;sup&gt;7&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Operating mode&lt;sup&gt;7&lt;/sup&gt;</td>
<td>—</td>
<td>Request</td>
<td>E</td>
<td>Request</td>
</tr>
<tr>
<td>Echo switch&lt;sup&gt;7&lt;/sup&gt;</td>
<td>—</td>
<td>Echo No echo</td>
<td>E</td>
<td>Echo</td>
</tr>
<tr>
<td>Initial status&lt;sup&gt;7&lt;/sup&gt;</td>
<td>—</td>
<td>Ok No pick</td>
<td>E</td>
<td>No pick</td>
</tr>
<tr>
<td>Initial segment&lt;sup&gt;7&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>undef</td>
</tr>
<tr>
<td>Initial pick identifier&lt;sup&gt;7&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>undef</td>
</tr>
<tr>
<td>Prompt and echo type&lt;sup&gt;7&lt;/sup&gt;</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Echo area&lt;sup&gt;7&lt;/sup&gt;</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>wdt</td>
</tr>
<tr>
<td>Pick data record&lt;sup&gt;7&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>wdt</td>
</tr>
<tr>
<td>String device number&lt;sup&gt;8&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Operating mode&lt;sup&gt;8&lt;/sup&gt;</td>
<td>—</td>
<td>Request</td>
<td>E</td>
<td>Request</td>
</tr>
<tr>
<td>Echo switch&lt;sup&gt;8&lt;/sup&gt;</td>
<td>—</td>
<td>Echo No echo</td>
<td>E</td>
<td>Echo</td>
</tr>
<tr>
<td>Initial string&lt;sup&gt;8&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>S</td>
<td>Blank ‘ ’</td>
</tr>
<tr>
<td>Prompt and echo type&lt;sup&gt;8&lt;/sup&gt;</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Echo area&lt;sup&gt;8&lt;/sup&gt;</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>wdt</td>
</tr>
<tr>
<td>String data record&lt;sup&gt;8&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>wdt</td>
</tr>
<tr>
<td>Input buffer size&lt;sup&gt;8,9&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
<tr>
<td>Initial cursor position&lt;sup&gt;8,9&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>wdt</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workstation type</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>—</td>
</tr>
<tr>
<td>Workstation category</td>
<td>—</td>
<td>Output Input</td>
<td>E</td>
<td>—</td>
</tr>
</tbody>
</table>

### 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

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device coordinate units</td>
<td>—</td>
<td>Meters Other</td>
<td>E</td>
<td>Meters</td>
</tr>
<tr>
<td>Maximum display surface size¹</td>
<td>DC Raster Units</td>
<td>&gt;0 Other</td>
<td>(2) R (2) I</td>
<td>—</td>
</tr>
</tbody>
</table>

### 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

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raster or vector display</td>
<td>—</td>
<td>Vector Raster Other</td>
<td>E</td>
<td>—</td>
</tr>
<tr>
<td>Polyline bundle representation¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Polymarker bundle representation¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Text bundle representation¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Fill area bundle representation¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Pattern representation¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Color representation¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Entry name</td>
<td>Coordinate system</td>
<td>Permitted values</td>
<td>Data type</td>
<td>Initial value</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-------------------</td>
<td>----------------------------------------</td>
<td>-----------</td>
<td>---------------</td>
</tr>
<tr>
<td>Workstation transformation¹</td>
<td>—</td>
<td>IRG, IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Deferral mode²</td>
<td>—</td>
<td>ASAP, BNI, BNIL</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>Implicit regeneration mode</td>
<td>—</td>
<td>Suppressed Allowed</td>
<td>E</td>
<td>Suppressed</td>
</tr>
<tr>
<td>Number of available linetypes</td>
<td>—</td>
<td>4, n</td>
<td>I</td>
<td>8</td>
</tr>
<tr>
<td>List of available linetypes</td>
<td>—</td>
<td>-n, -1, 1, n</td>
<td>(n) I</td>
<td>-4, -1, 1, 4</td>
</tr>
<tr>
<td>Number of available linewidths³</td>
<td>—</td>
<td>0, n</td>
<td>I</td>
<td>2</td>
</tr>
<tr>
<td>Nominal linewidth</td>
<td>DC</td>
<td>&gt;0</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Minimum linewidth</td>
<td>DC</td>
<td>&gt;0</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Maximum linewidth</td>
<td>DC</td>
<td>&gt;0</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Number of predefined polyline indexes (bundles)</td>
<td>—</td>
<td>5, n</td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>Linetype⁴</td>
<td>—</td>
<td>-n, -1, 1, n</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Linewidth scale factor⁴</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Polyl ine color index (within the range of predefined color indexes)⁴</td>
<td>—</td>
<td>0, n</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Number of available marker types</td>
<td>—</td>
<td>5, n</td>
<td>I</td>
<td>10</td>
</tr>
<tr>
<td>List of available marker types</td>
<td>—</td>
<td>-n, -1, 1, n</td>
<td>(n) I</td>
<td>-5, -1, 1, 5</td>
</tr>
<tr>
<td>Number of available marker sizes⁵</td>
<td>—</td>
<td>0, n</td>
<td>I</td>
<td>0</td>
</tr>
<tr>
<td>Nominal marker size</td>
<td>DC</td>
<td>&gt;0</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Minimum marker size</td>
<td>DC</td>
<td>&gt;0</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Maximum marker size</td>
<td>DC</td>
<td>&gt;0</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Number of predefined polymarker indexes (bundles)</td>
<td>—</td>
<td>5, n</td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>Marker type⁶</td>
<td>—</td>
<td>-n, -1, 1, n</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Marker size scale factor⁶</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Polymarker color index (within the range of predefined color indexes)⁶</td>
<td>—</td>
<td>0, n</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Number of text font and precision pairs</td>
<td>—</td>
<td>1, n</td>
<td>I</td>
<td>44</td>
</tr>
<tr>
<td>Entry name</td>
<td>Coordinate system</td>
<td>Permitted values</td>
<td>Data type</td>
<td>Initial value</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>-------------------</td>
<td>--------------------------------------</td>
<td>-----------</td>
<td>---------------</td>
</tr>
<tr>
<td>List of text font and precision pairs</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>(n) I</td>
<td>—</td>
</tr>
<tr>
<td>Number of available character expansion factors</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Minimum character expansion factor</td>
<td>—</td>
<td>&gt;0</td>
<td>R</td>
<td>1</td>
</tr>
<tr>
<td>Maximum character expansion factor</td>
<td>—</td>
<td>&gt;0</td>
<td>R</td>
<td>1</td>
</tr>
<tr>
<td>Number of available character heights</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Minimum character height</td>
<td>DC</td>
<td>&gt;0</td>
<td>R</td>
<td>—</td>
</tr>
<tr>
<td>Maximum character height</td>
<td>DC</td>
<td>&gt;0</td>
<td>R</td>
<td>—</td>
</tr>
<tr>
<td>Number of predefined text indexes (bundles)</td>
<td>—</td>
<td>2 ... n</td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>Text font and precision</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>E</td>
</tr>
<tr>
<td>Character expansion factor</td>
<td>—</td>
<td>&gt;0</td>
<td>R</td>
<td>—</td>
</tr>
<tr>
<td>Character spacing</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>—</td>
</tr>
<tr>
<td>Text color index (within the range of predefined color indexes)</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Number of available fill area interior styles</td>
<td>—</td>
<td>1 ... 4</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>List of available fill area interior styles</td>
<td>—</td>
<td>Hollow</td>
<td>(n) E</td>
<td>—</td>
</tr>
<tr>
<td>Number of available hatch styles</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>6</td>
</tr>
<tr>
<td>List of available hatch styles</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>(n) I</td>
<td>-6 ... -1</td>
</tr>
<tr>
<td>Number of predefined fill area indexes (bundles)</td>
<td>—</td>
<td>5 ... n</td>
<td>I</td>
<td>5</td>
</tr>
<tr>
<td>Fill area interior style</td>
<td>—</td>
<td>Hollow</td>
<td>E</td>
<td>—</td>
</tr>
<tr>
<td>Fill area style index</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
</tbody>
</table>
### GKS data structures

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fill area color index (within the range of predefined color indexes)</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Number of predefined pattern indexes (representations)</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Pattern array dimensions&lt;sup&gt;14&lt;/sup&gt;</td>
<td>—</td>
<td>1 ... n</td>
<td>(2) I</td>
<td>—</td>
</tr>
<tr>
<td>Pattern array&lt;sup&gt;14&lt;/sup&gt;</td>
<td>—</td>
<td>0 ... n</td>
<td>(n) n x I</td>
<td>—</td>
</tr>
<tr>
<td>Number of available colors or intensities&lt;sup&gt;15&lt;/sup&gt;</td>
<td>—</td>
<td>0,2 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Color available</td>
<td>—</td>
<td>Color</td>
<td>E</td>
<td>—</td>
</tr>
<tr>
<td>Number of predefined color indexes (representations)</td>
<td>—</td>
<td>2 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Color (red, green, blue intensities)&lt;sup&gt;16&lt;/sup&gt;</td>
<td>—</td>
<td>[0,1]</td>
<td>(3) R</td>
<td>—</td>
</tr>
<tr>
<td>Number of available generalized drawing primitives</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>0</td>
</tr>
<tr>
<td>Generalized drawing primitive identifier&lt;sup&gt;17&lt;/sup&gt;</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>—</td>
</tr>
<tr>
<td>Number of sets of attributes used&lt;sup&gt;17&lt;/sup&gt;</td>
<td>—</td>
<td>0 ... 4</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>List of sets of attributes used&lt;sup&gt;17&lt;/sup&gt;</td>
<td>—</td>
<td>Polyline, Polymarker, Text, Fill Area</td>
<td>(n) E</td>
<td>—</td>
</tr>
<tr>
<td>Maximum number of polyline bundle table entries</td>
<td>—</td>
<td>5 ... n</td>
<td>I</td>
<td>20</td>
</tr>
<tr>
<td>Maximum number of polymarker bundle table entries</td>
<td>—</td>
<td>5 ... n</td>
<td>I</td>
<td>20</td>
</tr>
<tr>
<td>Maximum number of text bundle table entries</td>
<td>—</td>
<td>2 ... n</td>
<td>I</td>
<td>10</td>
</tr>
<tr>
<td>Maximum number of fill area bundle table entries</td>
<td>—</td>
<td>5 ... n</td>
<td>I</td>
<td>20</td>
</tr>
<tr>
<td>Maximum number of pattern indexes</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Maximum number of color indexes</td>
<td>—</td>
<td>2 ... n</td>
<td>I</td>
<td>20</td>
</tr>
<tr>
<td>Number of segment priorities supported&lt;sup&gt;18&lt;/sup&gt;</td>
<td>—</td>
<td>0 ... n</td>
<td>I</td>
<td>0</td>
</tr>
</tbody>
</table>
### 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.

---

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment transformation¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Visibility (visible-&gt;invisible)¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Visibility (invisible-&gt;visible)¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Highlighting¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Segment priority¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Adding primitives to open segment overlapping segment of higher priority¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
<tr>
<td>Delete segment¹</td>
<td>—</td>
<td>IRG IMM</td>
<td>E</td>
<td>IRG</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locator device number¹</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Default initial locator position¹</td>
<td>WC</td>
<td>—</td>
<td>P</td>
<td>0.1, 0.1</td>
</tr>
<tr>
<td>Number of available prompt and echo types¹</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>List of available prompt and echo types¹</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>(n) I</td>
<td>—</td>
</tr>
<tr>
<td>Default echo area¹</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>—</td>
</tr>
<tr>
<td>Default locator data record¹</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>—</td>
</tr>
<tr>
<td>Stroke device number²</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Maximum input buffer size²</td>
<td>—</td>
<td>64 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Number of available prompt and echo types²</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>List of available prompt and echo types²</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>(n) I</td>
<td>—</td>
</tr>
<tr>
<td>Default echo area²</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>—</td>
</tr>
<tr>
<td>Default stroke data record²,³</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>—</td>
</tr>
<tr>
<td>Input buffer size²,³</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>64</td>
</tr>
<tr>
<td>Valuator device number⁴</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Default initial value⁴</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>—</td>
</tr>
<tr>
<td>Number of available prompt and echo types⁴</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>List of available prompt and echo types⁴</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>(n) I</td>
<td>1</td>
</tr>
<tr>
<td>Default echo area⁴</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>—</td>
</tr>
<tr>
<td>Default valuator data record⁴</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>—</td>
</tr>
<tr>
<td>Low value⁴,⁵</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>-1E18</td>
</tr>
<tr>
<td>High value⁴,⁵</td>
<td>—</td>
<td>—</td>
<td>R</td>
<td>1E18</td>
</tr>
<tr>
<td>Choice device number⁶</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Maximum number of choice alternatives⁶</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Number of available prompt and echo types⁶</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>List of available prompt and echo types⁶</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>(n) I</td>
<td>1</td>
</tr>
<tr>
<td>Default echo area⁶</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>—</td>
</tr>
<tr>
<td>Default choice data record⁶</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>—</td>
</tr>
<tr>
<td>Pick device number⁷</td>
<td>—</td>
<td>—</td>
<td>I</td>
<td>—</td>
</tr>
</tbody>
</table>
### GKS data structures

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of available prompt and echo types(^7)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>List of available prompt and echo types(^7)</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>(n) I</td>
<td>1</td>
</tr>
<tr>
<td>Default echo area(^7)</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>—</td>
</tr>
<tr>
<td>Default pick data record(^7)</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>—</td>
</tr>
<tr>
<td>String device number(^8)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>Maximum input buffer size(^8)</td>
<td>—</td>
<td>72 ... n</td>
<td>I</td>
<td>72</td>
</tr>
<tr>
<td>Number of available prompt and echo types(^8)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>—</td>
</tr>
<tr>
<td>List of available prompt and echo types(^8)</td>
<td>—</td>
<td>-n ... -1, 1 ... n</td>
<td>(n) I</td>
<td>—</td>
</tr>
<tr>
<td>Default echo area(^8)</td>
<td>DC</td>
<td>—</td>
<td>(4) R</td>
<td>—</td>
</tr>
<tr>
<td>Default string data record(^8)</td>
<td>—</td>
<td>—</td>
<td>D</td>
<td>—</td>
</tr>
<tr>
<td>Input buffer size(^8)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>72</td>
</tr>
<tr>
<td>Initial cursor position(^8)</td>
<td>—</td>
<td>1 ... n</td>
<td>I</td>
<td>1</td>
</tr>
</tbody>
</table>

### 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.

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment name</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>—</td>
</tr>
<tr>
<td>Set of associated workstations</td>
<td>—</td>
<td>—</td>
<td>(n) N</td>
<td>1</td>
</tr>
<tr>
<td>Segment transformation matrix</td>
<td>—</td>
<td>—</td>
<td>(2 x 3) R</td>
<td>1,0,0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0,1,0^2</td>
</tr>
<tr>
<td>Visibility</td>
<td>—</td>
<td>Visible</td>
<td>E</td>
<td>Visible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Invisible</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Highlighting</td>
<td>—</td>
<td>Normal</td>
<td>E</td>
<td>Normal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Highlighted</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Segment priority</td>
<td>—</td>
<td>[0,1]</td>
<td>R</td>
<td>0</td>
</tr>
<tr>
<td>Detectability</td>
<td>—</td>
<td>Undetectable</td>
<td>E</td>
<td>Undetectable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Detectable</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 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

<table>
<thead>
<tr>
<th>Entry name</th>
<th>Coordinate system</th>
<th>Permitted values</th>
<th>Data type</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error state</td>
<td>—</td>
<td>Off</td>
<td>E</td>
<td>Off</td>
</tr>
<tr>
<td></td>
<td></td>
<td>On</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error file</td>
<td>—</td>
<td>—</td>
<td>N</td>
<td>0</td>
</tr>
</tbody>
</table>
GKS data structures
## 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.

<table>
<thead>
<tr>
<th>Enumeration type</th>
<th>States</th>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspect source</td>
<td>Bundled</td>
<td>GBUNDL</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Individual</td>
<td>GINDIV</td>
<td>1</td>
</tr>
<tr>
<td>Attribute control flag</td>
<td>Current</td>
<td>GCURNT</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Specified</td>
<td>GSPEC</td>
<td>1</td>
</tr>
<tr>
<td>Clear control flag</td>
<td>Conditionally</td>
<td>GCOND</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Always</td>
<td>GALWAY</td>
<td>1</td>
</tr>
<tr>
<td>Clipping indicator</td>
<td>No clip</td>
<td>GNCLIP</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Clip</td>
<td>GCLIP</td>
<td>1</td>
</tr>
<tr>
<td>Color available</td>
<td>Monochrome</td>
<td>GMONOC</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Color</td>
<td>GCOLOR</td>
<td>1</td>
</tr>
<tr>
<td>Coordinate switch</td>
<td>World coordinates</td>
<td>GWC</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>NDC</td>
<td>GNDC</td>
<td>1</td>
</tr>
<tr>
<td>Deferral mode</td>
<td>As soon as possible</td>
<td>GASAP</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>BNIG</td>
<td>GBNIG</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>BNIL</td>
<td>GBNIL</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>At some time</td>
<td>GASTI</td>
<td>3</td>
</tr>
<tr>
<td>Detectability</td>
<td>Undetectable</td>
<td>GUNDET</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Detectable</td>
<td>GDETEC</td>
<td>1</td>
</tr>
<tr>
<td>Device coordinate units</td>
<td>Meters</td>
<td>GMETRE</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>GOTHU</td>
<td>1</td>
</tr>
<tr>
<td>Display surface empty</td>
<td>Not Empty</td>
<td>GNEMPT</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Empty</td>
<td>GEMPTY</td>
<td>1</td>
</tr>
<tr>
<td>Dynamic modification</td>
<td>Implicit regeneration</td>
<td>GIRG</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Immediate</td>
<td>GiMM</td>
<td>1</td>
</tr>
</tbody>
</table>
## Enumeration types

<table>
<thead>
<tr>
<th>Enumeration type</th>
<th>States</th>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Echo switch</td>
<td>No echo</td>
<td>GNECHO</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Echo</td>
<td>GECHO</td>
<td>1</td>
</tr>
<tr>
<td>Fill area interior style</td>
<td>Hollow</td>
<td>GHOLLO</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Solid</td>
<td>GSOLID</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Pattern</td>
<td>GPATTR</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Hatch</td>
<td>GHATCH</td>
<td>3</td>
</tr>
<tr>
<td>GDP attributes</td>
<td>Polyl ine bundle</td>
<td>GPLBND</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Polymarker bundle</td>
<td>GPMBND</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Text bundle</td>
<td>GTXBND</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Fill area bundle</td>
<td>GFABND</td>
<td>3</td>
</tr>
<tr>
<td>Highlighting</td>
<td>Normal</td>
<td>GNORMNL</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Highlighted</td>
<td>GHILIT</td>
<td>1</td>
</tr>
<tr>
<td>Initial choice prompt flag</td>
<td>Off</td>
<td>GPROFF</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>On</td>
<td>GPRON</td>
<td>1</td>
</tr>
<tr>
<td>Input device status</td>
<td>None</td>
<td>GNONE</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Ok</td>
<td>GOK</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>No pick</td>
<td>GN PICK</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>No choice</td>
<td>GNCHOI</td>
<td>2</td>
</tr>
<tr>
<td>Input class</td>
<td>None</td>
<td>GNCLAS</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Locator</td>
<td>GLOCAT</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Stroke</td>
<td>GSTROK</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Valuator</td>
<td>GVALUA</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Choice</td>
<td>GCHOIC</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Pick</td>
<td>GPICK</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>String</td>
<td>GSTRIN</td>
<td>6</td>
</tr>
<tr>
<td>Implicit regeneration mode</td>
<td>Suppressed</td>
<td>GSUPPD</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Allowed</td>
<td>GALLOW</td>
<td>1</td>
</tr>
<tr>
<td>Level of GKS</td>
<td>0a</td>
<td>GL0A</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0b</td>
<td>GL0B</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0c</td>
<td>GL0C</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1a</td>
<td>GL1A</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1b</td>
<td>GL1B</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>1c</td>
<td>GL1C</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>2a</td>
<td>GL2A</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>2b</td>
<td>GL2B</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>2c</td>
<td>GL2C</td>
<td>8</td>
</tr>
<tr>
<td>Line type</td>
<td>Solid</td>
<td>GLSOLI</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Dash</td>
<td>GLDASH</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Dot</td>
<td>GLDOT</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Dash-dot</td>
<td>GLDASD</td>
<td>4</td>
</tr>
<tr>
<td>Marker type</td>
<td>Period (.)</td>
<td>GPOINT</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Plus (+)</td>
<td>GPLUS</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Asterisk (*)</td>
<td>GAST</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Circle (o)</td>
<td>GOMARK</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Cross (x)</td>
<td>GXMARK</td>
<td>5</td>
</tr>
<tr>
<td>New frame action necessary</td>
<td>No</td>
<td>GNO</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>GYES</td>
<td>1</td>
</tr>
<tr>
<td>Operating mode</td>
<td>Request</td>
<td>GREQU</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Sample</td>
<td>GSAMPL</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Event</td>
<td>GEVENT</td>
<td>2</td>
</tr>
</tbody>
</table>
# Enumeration types

<table>
<thead>
<tr>
<th>Enumeration type</th>
<th>States</th>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating state value</td>
<td>GKS closed</td>
<td>GGKCL</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>GKS open</td>
<td>GGKOP</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>WSOP</td>
<td>GWSP</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>WSAC</td>
<td>GWSAC</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Segment open</td>
<td>GSGOP</td>
<td>4</td>
</tr>
<tr>
<td>Polyline/fill area control flag</td>
<td>Polyline</td>
<td>GPLINE</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Fill area</td>
<td>GFILL</td>
<td>1</td>
</tr>
<tr>
<td>Presence of invalid values</td>
<td>Absent</td>
<td>GABSNT</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Present</td>
<td>GPRSNT</td>
<td>1</td>
</tr>
<tr>
<td>Regeneration flag</td>
<td>Postpone</td>
<td>GPOSTP</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Perform</td>
<td>GPERFO</td>
<td>1</td>
</tr>
<tr>
<td>Relative input priority</td>
<td>Higher</td>
<td>GHIGHR</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Lower</td>
<td>GLOWER</td>
<td>1</td>
</tr>
<tr>
<td>Simultaneous events flag</td>
<td>No more</td>
<td>GNMORE</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>More</td>
<td>GMORE</td>
<td>1</td>
</tr>
<tr>
<td>Text alignment horizontal</td>
<td>Normal</td>
<td>GAHNOR</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Left</td>
<td>GALEFT</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Center</td>
<td>GACENT</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Right</td>
<td>GARITE</td>
<td>3</td>
</tr>
<tr>
<td>Text alignment vertical</td>
<td>Normal</td>
<td>GAVNOR</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Top</td>
<td>GATOP</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Cap</td>
<td>GACAP</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Half</td>
<td>GAHALF</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Base</td>
<td>GABASE</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Bottom</td>
<td>GABOTT</td>
<td>5</td>
</tr>
<tr>
<td>Text path</td>
<td>Right</td>
<td>GRIGHT</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Left</td>
<td>GLEFT</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Up</td>
<td>GUP</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Down</td>
<td>GDOWN</td>
<td>3</td>
</tr>
<tr>
<td>Text precision</td>
<td>String</td>
<td>GSTRP</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Character</td>
<td>GCHARP</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Stroke</td>
<td>GSTRKP</td>
<td>2</td>
</tr>
<tr>
<td>Type of returned value</td>
<td>Set</td>
<td>GSET</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Realized</td>
<td>GREALI</td>
<td>1</td>
</tr>
<tr>
<td>Update state</td>
<td>Pending</td>
<td>GNPEND</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Not pending</td>
<td>GPEND</td>
<td>1</td>
</tr>
<tr>
<td>Vector/raster other type</td>
<td>Vector</td>
<td>GVECTR</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Raster</td>
<td>GRASTR</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>GOTHWK</td>
<td>2</td>
</tr>
<tr>
<td>Visibility</td>
<td>Invisible</td>
<td>GINVIS</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Visible</td>
<td>GVISI</td>
<td>1</td>
</tr>
<tr>
<td>Workstation category</td>
<td>Output</td>
<td>GOUTPT</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Input</td>
<td>GINPUT</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Input/output</td>
<td>GOUTIN</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>WISS</td>
<td>GWISS</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Metafile output</td>
<td>GMO</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Metafile input</td>
<td>GMI</td>
<td>5</td>
</tr>
<tr>
<td>Workstation state</td>
<td>Inactive</td>
<td>GINACT</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Active</td>
<td>GACTIV</td>
<td>1</td>
</tr>
</tbody>
</table>
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.
### Metafile structure

#### Control functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Item types created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open workstation (GOPWK)</td>
<td>- (file header)</td>
</tr>
<tr>
<td>Close workstation (GCLWK)</td>
<td>0 (end item)</td>
</tr>
<tr>
<td>Activate workstation (GACWK)</td>
<td>61, 21-43</td>
</tr>
<tr>
<td>Deactivate workstation (GDAWK)</td>
<td>- (disable output)</td>
</tr>
<tr>
<td>Clear workstation (GCLRWK)</td>
<td>1</td>
</tr>
<tr>
<td>Redraw all segments on workstation (GRSGWK)</td>
<td>2</td>
</tr>
<tr>
<td>Update workstation (GUWK)</td>
<td>3</td>
</tr>
<tr>
<td>Set deferral state (GSDS)</td>
<td>4</td>
</tr>
<tr>
<td>Message (GMSGS)</td>
<td>5</td>
</tr>
</tbody>
</table>

#### Output primitives

<table>
<thead>
<tr>
<th>Function</th>
<th>Item types created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polyline (GPL)</td>
<td>11</td>
</tr>
<tr>
<td>Polymarker (GPM)</td>
<td>12</td>
</tr>
<tr>
<td>Text (GTXS)</td>
<td>13</td>
</tr>
<tr>
<td>Fill area (GFA)</td>
<td>14</td>
</tr>
<tr>
<td>Cell array (GCA)</td>
<td>15</td>
</tr>
</tbody>
</table>

#### Output attributes

<table>
<thead>
<tr>
<th>Function</th>
<th>Item types created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set polyline index (GSPLI)</td>
<td>21</td>
</tr>
<tr>
<td>Set linetype (GSLN)</td>
<td>22</td>
</tr>
<tr>
<td>Set linewidth scale factor (GSLWSC)</td>
<td>23</td>
</tr>
<tr>
<td>Set polyline color index (GSPLCI)</td>
<td>24</td>
</tr>
<tr>
<td>Set polymarker index (GSPMI)</td>
<td>25</td>
</tr>
<tr>
<td>Set marker type (GSMK)</td>
<td>26</td>
</tr>
<tr>
<td>Set marker size scale factor (GSMKSC)</td>
<td>27</td>
</tr>
<tr>
<td>Set polymarker color index (GSPMCI)</td>
<td>28</td>
</tr>
<tr>
<td>Set text index (GSTXI)</td>
<td>29</td>
</tr>
<tr>
<td>Set text font and precision (GSTXFP)</td>
<td>30</td>
</tr>
<tr>
<td>Set character expansion factor (GSCHXP)</td>
<td>31</td>
</tr>
<tr>
<td>Set character spacing (GSCHSP)</td>
<td>32</td>
</tr>
<tr>
<td>Set text color index (GSTXCI)</td>
<td>33</td>
</tr>
<tr>
<td>Set character height (GSCHH)</td>
<td>34</td>
</tr>
<tr>
<td>Set character up vector (GSCHUP)</td>
<td>34</td>
</tr>
<tr>
<td>Set text path (GSTXP)</td>
<td>35</td>
</tr>
<tr>
<td>Set text alignment (GSTXAL)</td>
<td>36</td>
</tr>
<tr>
<td>Set fill area index (GSFAL)</td>
<td>37</td>
</tr>
<tr>
<td>Set fill area interior style (GSFAIS)</td>
<td>38</td>
</tr>
<tr>
<td>Set fill area style index (GSFASI)</td>
<td>39</td>
</tr>
<tr>
<td>Set fill area color index (GSFACI)</td>
<td>40</td>
</tr>
<tr>
<td>Function</td>
<td>Item types created</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Set pattern size (GSPA)</td>
<td>41</td>
</tr>
<tr>
<td>Set pattern reference point (GSPARF)</td>
<td>42</td>
</tr>
<tr>
<td>Set aspect source flags (GASF)</td>
<td>43</td>
</tr>
<tr>
<td>Set pick identifier (GSPKID)</td>
<td>44</td>
</tr>
</tbody>
</table>

**Workstation attributes**

<table>
<thead>
<tr>
<th>Function</th>
<th>Item types created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set polyline representation (GSPLR)</td>
<td>51</td>
</tr>
<tr>
<td>Set polymarker representation (GSPMR)</td>
<td>52</td>
</tr>
<tr>
<td>Set text representation (GSTXR)</td>
<td>53</td>
</tr>
<tr>
<td>Set fill area representation (GSFAR)</td>
<td>54</td>
</tr>
<tr>
<td>Set pattern representation (GSPAR)</td>
<td>55</td>
</tr>
<tr>
<td>Set color representation (GSCR)</td>
<td>56</td>
</tr>
</tbody>
</table>

**Transformation functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Item types created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set window (GSWN)</td>
<td>34, 41, 42</td>
</tr>
<tr>
<td>Set viewport (GSVP)</td>
<td>61, 34, 41, 42</td>
</tr>
<tr>
<td>Select normalization transformation (GSELNT)</td>
<td>61, 34, 41, 42</td>
</tr>
<tr>
<td>Set clipping indicator (GSCLIP)</td>
<td>61</td>
</tr>
<tr>
<td>Set workstation window (GSWKWN)</td>
<td>71</td>
</tr>
<tr>
<td>Set workstation viewport (GSWKVP)</td>
<td>72</td>
</tr>
</tbody>
</table>

**Segment functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Item types created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create segment (GCRSG)</td>
<td>44, 81</td>
</tr>
<tr>
<td>Close segment (GCLSG)</td>
<td>82</td>
</tr>
<tr>
<td>Rename segment (GRENSG)</td>
<td>83</td>
</tr>
<tr>
<td>Delete segment (GDSG)</td>
<td>84</td>
</tr>
<tr>
<td>Delete segment from workstation (GDSGWK)</td>
<td>84</td>
</tr>
<tr>
<td>Associate segment with workstation (GASGWK)</td>
<td>81, (91-95), (21-44), (11-16), 61, 82</td>
</tr>
<tr>
<td>Copy segment to workstation (GCSGWK)</td>
<td>(21-44), (11-16), 61</td>
</tr>
<tr>
<td>Insert segment (GINSG)</td>
<td>(21-44), (11-16), 61</td>
</tr>
</tbody>
</table>

**Segment attributes**

<table>
<thead>
<tr>
<th>Function</th>
<th>Item types created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set segment transformation (GSSGT)</td>
<td>91</td>
</tr>
<tr>
<td>Set visibility (GSVIS)</td>
<td>92</td>
</tr>
<tr>
<td>Set highlighting (GSHLIT)</td>
<td>93</td>
</tr>
<tr>
<td>Set segment priority (GSSGP)</td>
<td>94</td>
</tr>
<tr>
<td>Set detectability (GSDTEC)</td>
<td>95</td>
</tr>
</tbody>
</table>

**Metafile functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Item types created</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write item to GKSM (GWITM)</td>
<td>&gt;100</td>
</tr>
</tbody>
</table>
Metafile structure

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
Metafile structure

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

- L(i) - The polyline index.

Linetype

- LT(i) - The line type.

Linewidth scale factor

- LW(r) - The line-width scale factor.

Polyline color index

- CI(i) - The polyline color index.

Polymarker index

- I(i) - The polymarker index.

Marker type

- MT(i) - The marker type.

Marker size scale factor

- MS(r) - The marker size scale factor.
Metafile structure

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

- P(i) - The text path.
  0 = Right
  1 = Left
  2 = Up
  3 = Down

Text alignment

- 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

- I(i) - The fill area index.

Fill area interior style

- S(i) - The fill area interior style.
  0 = Hollow
  1 = Solid
  2 = Pattern
  3 = Hatch

Fill area style index

- SI(i) - The fill area style index.
Metafile structure

Fill area color index

\[
\begin{array}{ccc}
40 & L & CI \\
\end{array}
\]

- CI(i) - The fill area color index.

Pattern vectors

\[
\begin{array}{ccc}
41 & L & PW & PH \\
\end{array}
\]

- PW(2r) - The pattern width vector.
- PH(2r) - The pattern height vector.

Pattern reference point

\[
\begin{array}{ccc}
42 & L & P \\
\end{array}
\]

- P(p) - The reference point.

Aspect source flags

\[
\begin{array}{ccc}
43 & L & F \\
\end{array}
\]

- F(13i) - The aspect source flags.
  0 = Bundled
  1 = Individual

Pick identifier

\[
\begin{array}{ccc}
44 & L & P \\
\end{array}
\]

- P(i) - The pick identifier.

Items for workstation attributes

Polyline representation

\[
\begin{array}{cccccc}
51 & L & I & LT & LW & CI \\
\end{array}
\]

- I(i) - The polyline index.
- LT(i) - The line type.
- LW(r) - The line-width scale factor.
- CI(i) - The polyline color index.
Metafile structure

Polymarker representation

<table>
<thead>
<tr>
<th>52</th>
<th>L</th>
<th>I</th>
<th>MT</th>
<th>MS</th>
<th>CI</th>
</tr>
</thead>
</table>

- l(i) - The polymarker index.
- MT(i) - The marker type.
- MS(r) - The marker size scale factor.
- CI(i) - The polymarker color index.

Text representation

<table>
<thead>
<tr>
<th>53</th>
<th>L</th>
<th>I</th>
<th>F</th>
<th>P</th>
<th>CEF</th>
<th>CS</th>
<th>CI</th>
</tr>
</thead>
</table>

- l(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

<table>
<thead>
<tr>
<th>54</th>
<th>L</th>
<th>I</th>
<th>S</th>
<th>SI</th>
<th>CI</th>
</tr>
</thead>
</table>

- l(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

<table>
<thead>
<tr>
<th>55</th>
<th>L</th>
<th>I</th>
<th>N</th>
<th>M</th>
<th>CT</th>
</tr>
</thead>
</table>

- l(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.
Metafile structure

Color representation

- CI(i) - The color index.
- RGB(3r) - The red, green, and blue intensities.

Items for transformations

Clipping rectangle

- C(4r) - The limits of the clipping rectangle (XMIN, XMAX, YMIN, YMAX).

Workstation window

- W(4r) - The limits of the workstation window (XMIN, XMAX, YMIN, YMAX).

Workstation viewport

- V(4r) - The limits of the workstation viewport (XMIN, XMAX, YMIN, YMAX).

Items for segment manipulation

Create segment

- S(i) - The segment name.

Close segment

This indicates the end of a segment.
rename segment

- SO(i) - The old segment name.
- SN(i) - The new segment name.

delete segment

- S(i) - The segment name.

items for segment attributes

segment transformation

- S(i) - The segment name.
- M(6r) - The transformation matrix:
  
  \[
  \begin{bmatrix}
  M_{11} & M_{12} & M_{13} \\
  M_{21} & M_{22} & M_{23}
  \end{bmatrix}
  \]

set visibility

- S(i) - The segment name.
- V(i) - The visibility setting:
  
  \[
  \begin{align*}
  0 &= \text{Visible} \\
  1 &= \text{Invisible}
  \end{align*}
  \]

set highlighting

- S(i) - The segment name.
- H(i) - The highlighting setting:
  
  \[
  \begin{align*}
  0 &= \text{Normal} \\
  1 &= \text{Highlighted}
  \end{align*}
  \]

set segment priority

- S(i) - The segment name.
- P(r) - The segment priority.

set detectability

- S(i) - The segment name.
- D(i) - The detectability setting:
  
  \[
  \begin{align*}
  0 &= \text{Undetectable}
  \end{align*}
  \]
Metafile structure

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****************************************************************************** ADM0010
C** ** ADM0020
C** 5666-802 ** ADM0030
C** (C) COPYRIGHT IBM CORP. 1987 ** ADM0040
C** LICENSED MATERIALS - PROPERTY OF IBM ** ADM0050
C** ** ADM0060
C** ADMJROOM ** ADM0070
C** ** ADM0080
C** A SAMPLE FORTRAN PROGRAM TO DO A SIMPLE ROOM LAYOUT WITH ** ADM0090
C** TWO FURNITURE SELECTIONS FOR PLACEMENT WITHIN A ROOM. ** ADM0100
C** ** ADM0110
C** ** ADM0120
C****************************************************************************** ADM0130
C ADM0140
C*** Control variables ADM0150
C ADM0160
C - Segment number, message flag ADM0170
INTEGER SEGNUM,MSGFLG ADM0180
C - Pick flag, position flag ADM0190
INTEGER PCKFLG,POSFLG ADM0200
C ADM0210
C*** Program data ADM0220
C ADM0230
C - Room Outline ADM0240
REAL XRRY1(7),YRRY1(7) ADM0250
C - Initial desk outline ADM0260
REAL XDRY(5),YDRY(5) ADM0270
C - Desk ADM0280
REAL XDSK(5),YDSK(5) ADM0290
C - Initial chair outline ADM0300
REAL XCHRY1(9),YCHRY1(9),XCHRY2(2) ADM0310
REAL YCHRY2(2),XCHRY3(5),YCHRY3(5) ADM0320
C - Chair ADM0330
REAL XCHR1(9),YCHR1(9),XCHR2(2),YCHR2(2) ADM0340
REAL XCHR3(5),YCHR3(5) ADM0350
C - Furniture width and height ADM0360
REAL FURWDT,FURHIG ADM0370
C ADM0380
C*** GKS input parameters ADM0390
C ADM0400
C - Workstation identifier, workstation type ADM0410
INTEGER WKID1,WKTYPE ADM0420
C - Normalization transformation viewport size ADM0430
REAL XNDC,YNDC ADM0440
C - Echo area for valuator input (minimum x/y values) ADM0450
REAL MINXDC,MINYDC ADM0460
C - X, Y shift for transformation matrix ADM0470
REAL SHFTX, SHFTY
C - Rotation angle
REAL ANG
C - Data record
   LOGICAL=1 DATREC(80)
C
C*** GKS output parameters
C
C - Error indicator, device coordinate units
INTEGER ERRIND, DCUNIT
C - Maximum display surface size in DC and Raster units
REAL XDC, YDC
INTEGER XRAS, YRAS
C - Number of segment names, nth set member of set of stored segments
INTEGER NUMSEG, SEGN
C - Input device status, choice number
INTEGER STAT, CHCNUM
C - Picked segment number, pick identifier
INTEGER SEGNM, PICKID
C - Locator returned transformation number and position
INTEGER TRANUM
REAL XPOP, YPOP
C - Transformation matrices
REAL MTX(3), MTX2(3)
C*** Work variables
C
REAL RTEMP
INTEGER J
C*** Initial values
C
C - Workstation identifier, workstation type
DATA WKTYPE /1/
DATA WKID1 /1/
C - Room Outline
DATA XRRY1 /00.0, 00.0, 60.0, 60.0, 00.0, 00.0, 00.0, 00.0, 10.5/
DATA YRRY1 /55.0, 99.8, 99.8, 34.0, 34.0, 36.0, 49.4/
C - Initial desk outline
DATA XDRY /75.0, 100.0, 100.0, 75.0, 75.0 /
DATA YDRY /87.0, 87.0, 100.0, 100.0, 87.0 /
C - Desk
DATA XDSDK /00.0, 00.0, 00.0, 00.0, 00.0, 00.0 /
DATA YDSDK /00.0, 00.0, 00.0, 00.0, 00.0, 00.0 /
C - Initial chair outline
DATA XCHRY1 /93.0, 93.0, 94.0, 94.0, 99.0, 99.0, 100.0, 100.0 /
   $ 100.0, 93.0 /
DATA YCHRY1 /73.0, 80.0, 80.0, 73.0, 73.0, 80.0, 80.0, 80.0 /
   $ 73.0, 73.0 /
DATA XCHRY2 /94.0, 99.0 /
DATA YCHRY2 /79.0, 79.0 /
DATA XCHRY3 /93.0, 93.0, 100.0, 100.0, 93.0 /
DATA YCHRY3 /73.0, 80.0, 80.0, 73.0, 73.0 /
C - Chair
DATA XCHR1 /00.0, 00.0, 00.0, 00.0, 00.0, 06.0, 06.0, 06.0, 07.0, 07.0, 00.0 /
DATA YCHR1 /00.0, 07.0, 07.0, 00.0, 00.0, 00.0, 07.0, 07.0, 00.0, 00.0 /
DATA XCHR2 /01.0, 06.0 /
DATA YCHR2 /06.0, 06.0 /
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
C**************************************************************************************
C*  INITIALISATION  *
C**************************************************************************************
C
C***  OPEN GKS
       CALL GOPKS(1,0)  ADM01090

C**************************************************************************************
C*  Open and activate all the workstations to be used  *
C**************************************************************************************
C
       CALL GOPWK(WKID1,1,WKTYPE)  ADM01100
       CALL GACWK(WKID1)  ADM01110

C**************************************************************************************
C*  Set normalization and workstation transformation  *
C**************************************************************************************
C
       CALL GSWN(1,0,0,0,0,0,0,0,0,0,100,0,100,0)  ADM01120
       CALL GQDSP(WKTYPE,ERRIND,DCUNIT,XDC,YDC,XRAS,YRAS)  ADM01130
       RTEMP = XDC  ADM01140
       IF (RTEMP .LT. YDC) RTEMP = YDC  ADM01150
       XNDC = XDC / RTEMP  ADM01160
       YNDC = YDC / RTEMP  ADM01170
       CALL GSVP(1,0,0,XNDC,0,0,YNDC)  ADM01180
       CALL GSWKWN(WKID1,0,0,XNDC,0,0,YNDC)  ADM01190
       CALL GSELNT(1)  ADM01200

C**************************************************************************************
C*  Output initial segments to display  *
C**************************************************************************************
C
       CALL GCRSG(1)  ADM01210

C
       CALL GSPKID(1)  ADM01220
       CALL GSFACI(1)  ADM01230
       CALL GSFAIS(1)  ADM01240
       CALL GFA(5,0,0)  ADM01250
       CALL GSPLCI(1)  ADM01260
       CALL GPL(5,0,0)  ADM01270
       CALL GSTXCI(3)  ADM01280

C***  Create segment 100: 2 items
       CALL GCRSG(100)  ADM01290

C
       CALL GSPKID(101)  ADM01300
       CALL GSFACI(0)  ADM01310
       CALL GSFAIS(1)  ADM01320
       CALL GFA(5,0,0)  ADM01330
       CALL GSPLCI(1)  ADM01340
       CALL GPL(5,0,0)  ADM01350
       CALL GSTXCI(3)  ADM01360

C***  Create segment 100: 2 items
       CALL GCRSG(100)  ADM01370

C
       CALL GSPKID(101)  ADM01380
       CALL GSFACI(0)  ADM01390
       CALL GSFAIS(1)  ADM01400
       CALL GFA(5,0,0)  ADM01410

C**************************************************************************************
C*  Output initial segments to display  *
C**************************************************************************************
C
       CALL GCRSG(1)  ADM01420

C**************************************************************************************
C*  Output initial segments to display  *
C**************************************************************************************
C
       CALL GCRSG(1)  ADM01430

C**************************************************************************************
C*  Output initial segments to display  *
C**************************************************************************************
C
       CALL GCRSG(1)  ADM01440

C**************************************************************************************
C*  Output initial segments to display  *
C**************************************************************************************
C
       CALL GCRSG(1)  ADM01450

C**************************************************************************************
C*  Output initial segments to display  *
C**************************************************************************************
C
       CALL GCRSG(1)  ADM01460

C***  Create segment 100: 2 items
       CALL GCRSG(100)  ADM01470

C
       CALL GSPKID(101)  ADM01480
       CALL GSFACI(0)  ADM01490
       CALL GSFAIS(1)  ADM01500
       CALL GFA(5,0,0)  ADM01510
       CALL GSPLCI(1)  ADM01520
       CALL GPL(5,0,0)  ADM01530
       CALL GSTXCI(3)  ADM01540

C***  Create segment 100: 2 items
       CALL GCRSG(100)  ADM01550

C
       CALL GSPKID(101)  ADM01560
       CALL GSFACI(0)  ADM01570
       CALL GSFAIS(1)  ADM01580
       CALL GFA(5,0,0)  ADM01590
       CALL GSPLCI(1)  ADM01600
       CALL GPL(5,0,0)  ADM01610
       CALL GSTXCI(3)  ADM01620
CALL GTXS(78.5,92.0,4,'desk')
C
C Set pick ID to 102 for the second item
CALL GSPKID(102)
C - Output an invisible fill area
CALL GFA(5,XCHRY3,YCHRY3)
C - Output 2 polylines to make a chair
CALL GPL(9,XCHRY1,YCHRY1)
CALL GPL(2,XCHRY2,YCHRY2)
C - Output text identifier
CALL GTXS(86.0,73.5,5,'chair')
C
C*** Close segment 100
CALL GCLSG
C*** Make the segment detectable
CALL GSDTEC(100,1)
C
C*** Create segment 200 : Room outline
CALL GCRSG(200)
C - Set line color
CALL GSPLCI(2)
C - Output lines for room outline
CALL GPL(7,XRRY1,YRRY1)
C
C*** Close segment 200
CALL GCLSG
C*** Create segment 300: Choice menu
CALL GCRSG(300)
CALL GSTXCI(6)
CALL GTXS(02.0,26.0,14,'PF1-pick furn.')
CALL GTXS(02.0,22.0,15,'PF2-place furn.')
CALL GTXS(53.0,26.0,16,'PF3-rotate furn.')
CALL GTXS(53.0,22.0,16,'PF4-remove furn.')
CALL GTXS(53.0,22.0,8,'PF5-exit')
C
C*** Close segment 300
CALL GCLSG
C
C*** Initialize input functions
MINXDC = XDC/8.0
MINYDC = YDC/10.0
CALL GINVL(WKID1,1,0.0,1,MINXDC,XDC,MINYDC,YDC,$ -360.0,*360.0,0,0,DATREC)
C
C*** Initialize locator with initial cursor position
CALL GINLC(WKID1,1,1,29.93,69.51,1,0.0,XDC,0.0,YDC,0,$ DATREC)
C
C*** Set viewport input priority
CALL GSVPIP(1,0,0)
C
C*** Set attributes and initial flags
ROOM program

C Set all the following output text color to 2
CALL GSTXCI(2)

C Set the type of all following lines to solid
CALL GSLN(1)

C Initialise segment number, message flag
SEGNUM = 0
MSGFLG = 0

C******************************************************************************
C* BEGIN USER INPUT - REPEAT UNTIL OPTION 5(EXIT) IS PICKED *
C******************************************************************************
C Inquire number of segments on workstation. If there are no
C segments other than the 3 segments in the initial screen, set
C pick flag and position flag to be off
C
1000 CALL GQSGWK(WKID1,1,ERRIND,NUMSEG,SEGN)
IF (NUMSEG .GT. 3) GOTO 11

PCKFLG = 0
POSFLG = 0

11 CONTINUE

C******************************************************************************
C* If a message already exists do not output another *
C******************************************************************************
C
IF (MSGFLG .NE. 0) GOTO 12

C Call REDRAW ALL SEGMENTS to clear msg area
CALL GRSGWK(WKID1)
CALL GTXS(1.0,1.0,30,'Enter option 1-5 using PF key ')

12 MSGFLG = 0

C******************************************************************************
C*** Request choice until status is OK and choice is 1 to 5 *
C******************************************************************************
C
1300 CALL GRQCH(WKID1,2,STAT,CHCNUM)
IF ((STAT .NE. 1) .OR. (CHCNUM .GT. 5) .OR. (CHCNUM .LT. 1))
GOTO 1300

C******************************************************************************
C***IF PICK FURNITURE IS CHOSEN *
C******************************************************************************
C
IF (CHCNUM .NE. 1) GOTO 24

C Call REDRAW ALL SEGMENTS to clear msg area
CALL GRSGWK(WKID1)
CALL GTXS(1.0,1.0,15,'Pick furniture ')

2= CALL GRQPK (WKID1, 1, STAT, SEGNM, PICKID)
IF (STAT .EQ. 2) GOTO 2

C******************************************************************************
C Test for invalid segment and status equal 2 'no pick'
C******************************************************************************
C
IF ((SEGNM .EQ. 1) .AND. (STAT .NE. 2)) GOTO 21

C Call REDRAW ALL SEGMENTS to clear msg area
CALL GRSGWK(WKID1)
CALL GSTXI(2)
ROOM program

CALL GTXS(1.0,1.0,0.35, 'Selection is invalid - pick again ')
CALL GSTXI(1)
GOTO 2000

2100 CONTINUE

C
PCKFLG = 1
SEGNUM = SEGNUM + 1

C
Switch according to the furniture picked

C
Create a desk segment
CALL GCRSG(SEGNUM)
CALL GSPKID(SEGNUM)

C
- Set the segment invisible before it is placed
CALL GSVIS(SEGNUM, 0)

C
- Output a fill area with background color
CALL GFA(5, XDSK, YDSK)
CALL GPL(5, XDSK, YDSK)

FURWDT=12.5
FURHIG=6.5

2200 CONTINUE

C

C
Create a chair segment
CALL GCRSG(SEGNUM)
CALL GSPKID(SEGNUM)

C
- Set the segment invisible before it is placed
CALL GSVIS(SEGNUM, 0)

C
- Output a fill with background color
CALL GFA(5, XCHR3, YCHR3)
CALL GPL(9, XCHR1, YCHR1)
CALL GPL(2, XCHR2, YCHR2)

FURWDT=3.5
FURHIG=3.5

2300 CONTINUE

C

C
Display msg asking user to place the furniture
CALL GCRSG(SEGNUM)
CALL GSPKID(SEGNUM)

C
- Set the segment invisible before it is placed
CALL GSVIS(SEGNUM, 0)

C
- Output a fill with background color
CALL GFA(5, XCHR3, YCHR3)
CALL GPL(9, XCHR1, YCHR1)
CALL GPL(2, XCHR2, YCHR2)

FURWDT=3.5
FURHIG=3.5

2400 CONTINUE

C

C
If no furniture is picked yet, display error message and exit

IF (CHCNUM .NE. 2) GOTO 3100

C
If no furniture is picked yet, display error message and exit

IF (PCKFLG .NE. 0) GOTO 2900

C
Call REDRAW ALL SEGMENTS to clear msg area
CALL GRSGWK(WKID1)
CALL GTXS(1.0,1.0,0.29,'You must pick furniture first')
MSGFLG = 1
GOTO 1000

2900 CONTINUE

C

C
Display msg asking user to place the furniture
CALL GCRSG(SEGNUM)
CALL GSPKID(SEGNUM)

C
Call REDRAW ALL SEGMENTS to clear msg area
CALL GRSGWK(WKID1)
CALL GTXS(1.0,1.0,33,'Place furniture in the room ') ADM03360
C**** Request locator until the furniture is inside the room ADM03370
3000 CALL GRLC(WKID1, 1, STAT, TRANUM, XPOP, YPOP) ADM03380
   IF ((STAT .EQ. 0) .OR. (XPOP .GT. 60) .OR. (YPOP .LT. 40)) ADM03400
     GOTO 3000 ADM03410
     POSFLG = 1 ADM03420
     CADM03430
C**** Transform the furniture and make the center point of the ADM03440
     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
     CADM03510
C**** Make the segment visible ADM03520
     CALL GSVIS(SEGNUM,1) ADM03530
     C**** and detectable ADM03540
     CALL GSDTEC(SEGNUM,1) ADM03550
3100 CONTINUE ADM03560
CADM03570
C ***IF ROTATE FURNITURE IS CHOSEN**
C *** If this furniture is not positioned yet, display error ADM03580
   and exit ADM03590
   CADM03600
   IF (CHCNUM .NE. 3) GOTO 4300 ADM03610
CADM03620
C**** Display msg asking user to enter rotation angle ADM03630
   Call REDRAW ALL SEGMENTS to clear msg area ADM03640
   CALL GRSGWK(WKID1) ADM03650
   CALL GSTXI(2) ADM03660
   CALL GTXS(1.0,1.0,38, 'You must position furniture first ') ADM03670
     GOTO 1000 ADM03680
3900 CONTINUE ADM03690
CADM03700
C**** Display msg asking user to enter rotation angle ADM03710
   Call REDRAW ALL SEGMENTS to clear msg area ADM03720
   CALL GRSGWK(WKID1) ADM03730
   CALL GTXS(1.0,1.0,33,'Enter rotation angle in degrees ') ADM03740
   CALL GQVL(WKID1,1,STAT,ANG) ADM03750
   IF (STAT .NE. 1 ) GOTO 4000 ADM03760
CADM03770
C Convert degrees to radians ADM03780
4200 ANG = ANG/180*3.1416 ADM03790
CADM03800
C**** Accumulate the transformation matrix and set transformation ADM03810
   CALL GACTM(MTX,XPOP,YPOP,1.0,1.0,ANG,1.0,1.0,MTX2) ADM03820
   CALL GSSGT(SEGNUM,MTX2) ADM03830
CADM03840
C 4300 CONTINUE ADM03850
CADM03860
Appendix D. ROOM program source code 403
ROOM program

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 GROPK(WKID1, 1, STAT, SEGNM, PICKID) ADM04160
IF ((STAT .NE. 1) .OR. (PICKID .GE. 1)) GOTO 5000 ADM04170
C ADM04180
C*** Delete the furniture segment ADM04190
CALL GDSEG(SEGNM) ADM04200
C*** Show segment does not exist any more ADM04210
PCKFLG = 0 ADM04220
POSFLG = 0 ADM04230
C ADM04240
C 5100 CONTINUE ADM04250
C ADM04260
C ***IF EXIT IS CHOSEN * ADM04270
C ****************************************************** 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
    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
```
Example programs

 INTEGER OPSTAT /0/
 C ERROR HANDLING
 C  INTEGER*4 ERRIND /0/
 C  
 C***************************************************************************************
 C**  ** MAIN PROGRAM
 C**  **
 C***************************************************************************************
 C*  **
 C*  ** Output header
 C 
 WRITE (*,*) ' '  
 WRITE (*,*) 'METOUT sample program '  
 WRITE (*,*) ' '  
 C Initialize the input metafile number
 C the output workstation connection identifier is 1 by default
 C  
 METAN = -1  
 C Request the metafile number from the operator
 C  
 WRITE (*,*) 'type the input metafile identifier (0 to 9999)'  
 READ (*,*,END=50) METAN  
 50 CONTINUE  
 C Check the returned metafile number
 C  
 IF ((METAN.LT.=zerodot).OR.(METAN.GT.9999)) THEN  
 WRITE(*,*) 'Metafile identifier is invalid'  
 GOTO 9999  
 ENDIF  
 C And set the requested workstation type: GDF file output
 WKTYPE = 5

 27600000
 28200000
 28800000
 29400000
 30000000
 30600000
 31200000
 31800000
 32400000
 33000000
 33600000
 34200000
 34800000
 35400000
 36000000
 36600000
 37200000
 37800000
 38400000
 39000000
 39600000
 40200000
 40800000
 41400000
 42000000
 42600000
 43200000
 43800000
 44400000
 45000000
 45600000
 46200000
 46800000
 47400000
 48000000
 48600000
 49200000
 49800000
 50400000
 51000000
 51600000
 52200000
 52800000
 53400000
 54000000
 54600000
 55200000
 55800000
 56400000
 57000000
 57600000
 58200000
 58800000
 59400000
 60000000
 60600000
 61200000
Example programs

ENDIF
C*****************************************************************************
C** OPEN GKS AND THE REQUIRED WORKSTATIONS
C*****************************************************************************
C This program uses error file ADMJ0000
CALL GOPKS(0,0)
C Open the workstation required for output
CALL GOPWK(1,CONNID,WKTYPE)
C and activate it
CALL GACWK(1)
C Open the metafile input workstation
CALL GOPWK(2,METAN,4)
C Now perform a loop retrieving the item type, item length and the data record for each item and then interpreting the item
C
CONTINUE
C- Get the item type and length
CALL GGTITM(2,MITYP,MILEN)
C- If the item is the END item we must leave the loop
IF (MITYP.EQ.0) GOTO 5999
C- else - skip the item if the data record will not fit in datrec or it is a user item
IF ((MILEN.GT.MAXLEN).OR.(MITYP.GT.1)) THEN
CALL GRDITM(2,MAXREC,DATREC)
ELSE
C- read and the item data record
CALL GRDITM(2,MILEN,MAXREC,DATREC)
C- and interpret the item
CALL GIITM(MITYP,MILEN,MAXREC,DATREC)
ENDIF
C- continue with the next item
GOTO 5000
5999 CONTINUE
C- update the workstation to ensure display is correct
CALL GUWK(1,1)
C- If the output workstation is the console request enter key from the operator
IF (WKTYPE.EQ.1) CALL GRQCH(1,1,STAT,CHNR)
C- Deactivate the output workstation
CALL GDAWK(1)
C- and close it
CALL GCLWK(1)
C- close the metafile input workstation
CALL GCLWK(2)
C- close GKS
CALL GCLKS
9999 CONTINUE
C***** Termination
STOP
END
C
SUBROUTINE GERHND (ERRNR,FCTID,ERRFIL)
C*****************************************************************************
C GERHND - This subroutine replaces GKS default error handling
C*****************************************************************************
STOP
END
Example programs

```
INTEGER ERRNR,FCTID,ERRFIL
C- Call GERLOG to perform error logging
   CALL GERLOG (ERRNR,FCTID,ERRFIL)
RETURN
END
```
**Example programs**

```
C******************************************************************************00100000
C ** PROGRAM METCNV
C******************************************************************************00200000
C
C******************************************************************************00300000
C **
C******************************************************************************00400000
C ** 5666-802
C******************************************************************************00600000
C ** (C) COPYRIGHT IBM CORP. 1987
C******************************************************************************00800000
C ** LICENSED MATERIALS - PROPERTY OF IBM
C******************************************************************************10000000
C **
C******************************************************************************12000000
C ** This program converts data between a GKS metafile and a character
C ** character equivalent of that data. The conversion can be done in either
c** direction (metafile to character or character to metafile).
C******************************************************************************18000000
C **
C******************************************************************************22000000
C ** Three I/O units (datasets) are used:
C******************************************************************************24000000
C ** - input options are read from unit "IIN"
C******************************************************************************26000000
C ** - output messages are written to unit "IOUT"
C******************************************************************************28000000
C ** - the character form of the metafile is read from or written to unit "ICHR".
C******************************************************************************32000000
C **
C******************************************************************************34000000
C******************************************************************************36000000
C
C******************************************************************************40000000
C PROGRAM METCNV
C******************************************************************************42000000
C**** DATA RECORD ARRAY
C******************************************************************************44000000
PARAMETER (MAXREC=10)
04600000
PARAMETER (MAXLEN=MAXREC+80)
04800000
CHARACTER=80 DATREC(MAXREC)
05000000
C
C******************************************************************************52000000
C**** CHAR/INTEGER/REAL EQUIVALENTS OF "DATREC"
C******************************************************************************54000000
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
C******************************************************************************70000000
C**** CONVERSION TYPE
C******************************************************************************72000000
PARAMETER (ICNVMC=1)
07400000
PARAMETER (ICNVCM=2)
07600000
C
C******************************************************************************80000000
C**** MAXIMUM (NON-USER) METAFILE ITEM TYPE
C******************************************************************************82000000
PARAMETER (MAXTYP=100)
08800000
C
C******************************************************************************84000000
C
C******************************************************************************90000000
C ** I N I T I A L I S A T I O N
C******************************************************************************92000000
C
C******************************************************************************94000000
C**** INITIALISE INPUT/OUTPUT UNIT NUMBERS
C******************************************************************************96000000
IIN=5
09600000
IOUT=6
09800000
ICHR=7
10000000
C
C******************************************************************************10000000
C**** INITIALISE GKS WORKSTATION IDENTIFIER
C******************************************************************************10200000
IWKID=1
10400000
C
C******************************************************************************10600000
C**** GET TYPE OF CONVERSION REQUIRED
C******************************************************************************11200000
WRITE(IOUT,10)ICNVMC,ICNVCM
11000000
10 FORMAT(/1X,'ENTER TYPE OF CONVERSION REQUIRED:'/)
```
Example programs

& 3X,I1,' = METAFILE TO CHARACTER'/
& 3X,I1,' = CHARACTER TO METAFILE')
READ(IIN,110)ICNV
110 FORMAT(I1) 12000000
C**** CHECK VALUE ENTERED
IF (ICNV.EQ.ICNVCM.OR.ICNV.EQ.ICNVMC) THEN
  WRITE(IOUT,120)ICNV
ELSE
  WRITE(IOUT,130)ICNV
120  FORMAT(/1X,'CONVERSION TYPE SPECIFIED = ',I1) 12800000
130  FORMAT(/1X,'CONVERSION TYPE ',I1,' IS NOT VALID.'/
& 3X,'EXECUTION IS TERMINATED.') 13400000
& 3X,'EXECUTION IS TERMINATED.')
GOTO 3333000000
ENDIF
C
C**** GET NUMBER OF METAFILE TO BE USED
WRITE(IOUT,150) 14400000
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
IF (IFILE.GE.1) THEN
  WRITE(IOUT,170)IFILE
ELSE
  WRITE(IOUT,180)IFILE
170  FORMAT(/1X,'METAFILE NUMBER SPECIFIED = ',I4) 16000000
180  FORMAT(/BN,1X,'METAFILE NUMBER ',I4,' IS NOT VALID.'/ 
& 3X,'EXECUTION IS TERMINATED.') 16600000
GOTO 3333000000
ENDIF
C
C**** OPEN GKS
CALL GOPKS 17600000
17800000
C**** OPEN APPROPRIATE METAFILE WORKSTATION (INPUT OR OUTPUT)
IF (ICNV.EQ.ICNVCM) THEN
  CALL GOPWK (IWKID,IFILE,3)
  CALL GACWK (IWKID)
ELSE
  CALL GOPWK (IWKID,IFILE,4)
ENDIF 19200000
C
C**** INITIALIZE METAFILE ITEM COUNTER
INUM=0 19400000
19600000
C
C *** ** METAFILE ITEM LOOP ** 20200000
C
C**** PROCESS METAFILE ITEMS
200 INUM=INUM+1 21200000
C
C**** READ/GET ITEM TYPE AND LENGTH
IF (ICNV.EQ.ICNVCM) THEN
  READ (ICHR,900)ITYP,ILEN
ELSE
  CALL GGTITM (IWKID,ITYP,ILEN)
ENDIF 22400000
GDDM-GKS Programming Guide and Reference
ENDIF

C

C**** CHECK THAT "DATREC" CAN CONTAIN ALL THE DATA
IF (ILEN.GT.MAXLEN) THEN
  WRITE(IOUT,220)INUM,ITYP,ILEN,MAXLEN
  FORMAT(/1X,'METAFILE ITEM ',I6,' WITH TYPE = ',I6,' AND LENGTH = ',I6,' IS TOO LONG./',3X,'EXECUTION IS TERMINATED. (MAX LENGTH SUPPORTED = ',I6,')')
GOTO 320
ENDIF

C

C**** GET ITEM DATA, AND WRITE ITEM TYPE AND LENGTH
IF (ICNV.EQ.ICNVMC) THEN
  CALL GRDITM (IWKID,MAXLEN,MAXREC,DATREC)
  WRITE(ICHR,90)ITYP,ILEN
ENDIF

C

C**** READ/WRITE ITEM DATA ACCORDING TO ITEM TYPE
IF (ITYP.EQ. 0) GOTO 1000
IF (ITYP.EQ. 1) GOTO 1001
IF (ITYP.EQ. 2) GOTO 1002
IF (ITYP.EQ. 3) GOTO 1003
IF (ITYP.EQ. 4) GOTO 1004
IF (ITYP.EQ. 5) GOTO 1005
IF (ITYP.EQ.11) GOTO 1011
IF (ITYP.EQ.12) GOTO 1012
IF (ITYP.EQ.13) GOTO 1013
IF (ITYP.EQ.14) GOTO 1014
IF (ITYP.EQ.15) GOTO 1015
IF (ITYP.EQ.21) GOTO 1021
IF (ITYP.EQ.22) GOTO 1022
IF (ITYP.EQ.23) GOTO 1023
IF (ITYP.EQ.24) GOTO 1024
IF (ITYP.EQ.25) GOTO 1025
IF (ITYP.EQ.26) GOTO 1026
IF (ITYP.EQ.27) GOTO 1027
IF (ITYP.EQ.28) GOTO 1028
IF (ITYP.EQ.29) GOTO 1029
IF (ITYP.EQ.30) GOTO 1030
IF (ITYP.EQ.31) GOTO 1031
IF (ITYP.EQ.32) GOTO 1032
IF (ITYP.EQ.33) GOTO 1033
IF (ITYP.EQ.34) GOTO 1034
IF (ITYP.EQ.35) GOTO 1035
IF (ITYP.EQ.36) GOTO 1036
IF (ITYP.EQ.37) GOTO 1037
IF (ITYP.EQ.38) GOTO 1038
IF (ITYP.EQ.39) GOTO 1039
IF (ITYP.EQ.40) GOTO 1040
IF (ITYP.EQ.41) GOTO 1041
IF (ITYP.EQ.42) GOTO 1042
IF (ITYP.EQ.43) GOTO 1043
IF (ITYP.EQ.44) GOTO 1044
IF (ITYP.EQ.51) GOTO 1051
IF (ITYP.EQ.52) GOTO 1052
IF (ITYP.EQ.53) GOTO 1053
IF (ITYP.EQ.54) GOTO 1054
IF (ITYP.EQ.55) GOTO 1055
Example programs

```plaintext
IF (ITYP.EQ.56) GOTO 1056
IF (ITYP.EQ.61) GOTO 1061
IF (ITYP.EQ.71) GOTO 1071
IF (ITYP.EQ.72) GOTO 1072
IF (ITYP.EQ.81) GOTO 1081
IF (ITYP.EQ.82) GOTO 1082
IF (ITYP.EQ.83) GOTO 1083
IF (ITYP.EQ.84) GOTO 1084
IF (ITYP.EQ.91) GOTO 1091
IF (ITYP.EQ.92) GOTO 1092
IF (ITYP.EQ.93) GOTO 1093
IF (ITYP.EQ.94) GOTO 1094
IF (ITYP.EQ.95) GOTO 1095
IF (ITYP.GT.100) GOTO 1100

C
C**** UNSUPPORTED ITEM TYPE
WRITE(IOUT,25)INUM,ITYP
GOTO 3200

C
C**** METAFILE-ITEM HEADER
900 FORMAT(2I15)
C**** METAFILE-ITEM DATA (CHARACTER)
901 FORMAT(8A1)
C**** METAFILE-ITEM DATA (INTEGER)
902 FORMAT(5I15)
C**** METAFILE-ITEM DATA (REAL)
903 FORMAT(1P5E15.6)
C
C**** END ITEM
1000 CONTINUE
C**** REDRAW ALL SEGMENTS
1002 CONTINUE
C**** CLOSE SEGMENT
1082 CONTINUE
C**** READ/WRITE ITEM DATA
GOTO 2000

C
C**** CLEAR WORKSTATION
1001 CONTINUE
C**** UPDATE WORKSTATION
1003 CONTINUE
C**** POLYLINE INDEX
1021 CONTINUE
C**** LINETYPE
1022 CONTINUE
C**** POLYLINE COLOR INDEX
1024 CONTINUE
C**** POLYMARKER INDEX
1025 CONTINUE
C**** MARKER TYPE
1026 CONTINUE
C**** POLYMARKER COLOR INDEX
1028 CONTINUE
C**** TEXT INDEX
1029 CONTINUE
C**** TEXT COLOR INDEX
```
C**** TEXT PATH
1035 CONTINUE
C**** FILL AREA INDEX
1037 CONTINUE
C**** FILL AREA INTERIOR STYLE
1038 CONTINUE
C**** FILL AREA STYLE INDEX
1039 CONTINUE
C**** FILL AREA COLOR INDEX
1040 CONTINUE
C**** PICK IDENTIFIER
1044 CONTINUE
C**** CREATE SEGMENT
1081 CONTINUE
C**** DELETE SEGMENT
1084 CONTINUE
C**** READ/WRITE ITEM DATA
IF (ICNV.EQ.ICNVCM) THEN
    READ (ICHR, 9) IDAT(1)
ELSE
    WRITE (ICHR, 9) IDAT(1)
ENDIF
GOTO 2
C
C**** DEFERRAL STATE
1004 CONTINUE
C**** TEXT FONT AND PRECISION
1030 CONTINUE
C**** TEXT ALIGNMENT
1036 CONTINUE
C**** RENAME SEGMENT
1083 CONTINUE
C**** SET VISIBILITY
1092 CONTINUE
C**** SET HIGHLIGHTING
1093 CONTINUE
C**** SET DETECTABILITY
1095 CONTINUE
C**** READ/WRITE ITEM DATA
IF (ICNV.EQ.ICNVCM) THEN
    READ (ICHR, 9) IDAT(1), IDAT(2)
ELSE
    WRITE (ICHR, 9) IDAT(1), IDAT(2)
ENDIF
GOTO 2
C
C**** MESSAGE
1005 IF (ICNV.EQ.ICNVCM) THEN
    READ (ICHR, 9) IDAT(1)
    IMAX = 4 + IDAT(1)
    READ (ICHR, 9) (CDAT(I), I = 5, IMAX)
ELSE
    WRITE (ICHR, 9) IDAT(1)
    IMAX = 4 + IDAT(1)
    WRITE (ICHR, 9) (CDAT(I), I = 5, IMAX)
ENDIF
Example programs

GOTO 2000
C
C**** POLYLINE
1011 CONTINUE
C
C**** POLYMARKER
1012 CONTINUE
C
C**** FILL AREA
1014 CONTINUE
C
C**** READ/WRITE ITEM DATA
IF (ICNV.EQ.ICNVCM) THEN
READ (ICHR,9)IDAT(1)
IMAX=1+IDAT(1)*2
READ (ICHR,9)(RDAT(I),I=2,IMAX)
ELSE
WRITE(ICHR,9)IDAT(1)
IMAX=1+IDAT(1)*2
WRITE(ICHR,9)(RDAT(I),I=2,IMAX)
ENDIF
GOTO 2000
C
C**** TEXT
1013 IF (ICNV.EQ.ICNVCM) THEN
READ (ICHR,9)RDAT(1),RDAT(2)
READ (ICHR,9)IDAT(3)
IMAX=12+IDAT(3)
READ (ICHR,9)(CDAT(I),I=13,IMAX)
ELSE
WRITE(ICHR,9)RDAT(1),RDAT(2)
WRITE(ICCHR,9)IDAT(3)
IMAX=12+IDAT(3)
WRITE(ICHR,9)(CDAT(I),I=13,IMAX)
ENDIF
GOTO 2000
C
C**** CELL ARRAY
1015 IF (ICNV.EQ.ICNVCM) THEN
READ (ICHR,9)(RDAT(I),I=1,6)
READ (ICHR,9)IDAT(7),IDAT(8)
IMAX=8+IDAT(7)*IDAT(8)
READ (ICHR,9)(IDAT(I),I=9,IMAX)
ELSE
WRITE(ICHR,9)(RDAT(I),I=1,6)
WRITE(ICCHR,9)IDAT(7),IDAT(8)
IMAX=8+IDAT(7)*IDAT(8)
WRITE(ICHR,9)(IDAT(I),I=9,IMAX)
ENDIF
GOTO 2000
C
C**** LINEWIDTH SCALE FACTOR
1023 CONTINUE
C
C**** MARKER SIZE SCALE FACTOR
1027 CONTINUE
C
C**** CHARACTER EXPANSION FACTOR
1031 CONTINUE
C
C**** CHARACTER SPACING
1032 CONTINUE
C
C**** READ/WRITE ITEM DATA
IF (ICNV.EQ.ICNVCM) THEN

414  GDDM-GKS Programming Guide and Reference
Example programs

```
READ (ICH,9=3) RDAT(1)
ELSE
  WRITE (ICH,9=3) RDAT(1)
ENDIF
GOTO 2000

C
C**** CHARACTER VECTORS
1034 CONTINUE
C**** PATTERN VECTORS
1041 CONTINUE
C**** CLIPPING RECTANGLE
1061 CONTINUE
C**** WORKSTATION WINDOW
1071 CONTINUE
C**** WORKSTATION VIEWPORT
1072 CONTINUE
C**** READ/WRITE ITEM DATA
  IF (ICNV.EQ.ICNVCM) THEN
    READ (ICH,9=3) RDAT(1), RDAT(2), RDAT(3), RDAT(4)
  ELSE
    WRITE (ICH,9=3) RDAT(1), RDAT(2), RDAT(3), RDAT(4)
  ENDIF
GOTO 2000

C
C**** PATTERN REFERENCE POINT
1042 IF (ICNV.EQ.ICNVCM) THEN
  READ (ICH,9=3) RDAT(1), RDAT(2)
ELSE
  WRITE (ICH,9=3) RDAT(1), RDAT(2)
ENDIF
GOTO 2000

C
C**** ASPECT SOURCE FLAGS
1043 IF (ICNV.EQ.ICNVCM) THEN
  READ (ICH,9=3) IDAT(I), I=1,13
ELSE
  WRITE (ICH,9=3) IDAT(I), I=1,13
ENDIF
GOTO 2000

C
C**** POLYLINE REPRESENTATION
1051 CONTINUE
C**** POLYMARKER REPRESENTATION
1052 CONTINUE
C**** READ/WRITE ITEM DATA
  IF (ICNV.EQ.ICNVCM) THEN
    READ (ICH,9=3) IDAT(1), IDAT(2)
    READ (ICH,9=3) RDAT(3)
    READ (ICH,9=3) IDAT(4)
  ELSE
    WRITE (ICH,9=3) IDAT(1), IDAT(2)
    WRITE (ICH,9=3) RDAT(3)
    WRITE (ICH,9=3) IDAT(4)
  ENDIF
GOTO 2000

C
C**** TEXT REPRESENTATION
1053 IF (ICNV.EQ.ICNVCM) THEN
```

Appendix E. Example programs 415
Example programs

```
READ (ICHR,9=zerodot2)IDAT(1),IDAT(2),IDAT(3)
READ (ICHR,9=zerodot3)RDAT(4),RDAT(5)
READ (ICHR,9=zerodot2)IDAT(6)
ELSE
  WRITE(ICHR,9=zerodot2)IDAT(1),IDAT(2),IDAT(3)
  WRITE(ICHR,9=zerodot3)RDAT(4),RDAT(5)
  WRITE(ICHR,9=zerodot2)IDAT(6)
ENDIF
GOTO 2000
C
C**** FILL AREA REPRESENTATION
1054 IF (ICNV.EQ.ICNVCM) THEN
  READ (ICHR,9=zerodot2)IDAT(1),IDAT(2),IDAT(3),IDAT(4)
ELSE
  WRITE(ICHR,9=zerodot2)IDAT(1),IDAT(2),IDAT(3),IDAT(4)
ENDIF
GOTO 2000
C
C**** FILL AREA REPRESENTATION
1055 IF (ICNV.EQ.ICNVCM) THEN
  READ (ICHR,9=zerodot2)IDAT(1),IDAT(2),IDAT(3)
  IMAX=3+IDAT(2)*IDAT(3)
  READ (ICHR,9=zerodot2)(IDAT(I),I=4,IMAX)
ELSE
  WRITE(ICHR,9=zerodot2)IDAT(1),IDAT(2),IDAT(3)
  IMAX=3+IDAT(2)*IDAT(3)
  WRITE(ICHR,9=zerodot2)(IDAT(I),I=4,IMAX)
ENDIF
GOTO 2000
C
C**** COLOR REPRESENTATION
1056 IF (ICNV.EQ.ICNVCM) THEN
  READ (ICHR,9=zerodot2)IDAT(1)
  READ (ICHR,9=zerodot3)RDAT(2),RDAT(3),RDAT(4)
ELSE
  WRITE(ICHR,9=zerodot2)IDAT(1)
  WRITE(ICHR,9=zerodot3)RDAT(2),RDAT(3),RDAT(4)
ENDIF
GOTO 2000
C
C**** SEGMENT TRANSFORMATION
1091 IF (ICNV.EQ.ICNVCM) THEN
  READ (ICHR,9=zerodot2)IDAT(1)
  READ (ICHR,9=zerodot3)(RDAT(I),I=2,7)
ELSE
  WRITE(ICHR,9=zerodot2)IDAT(1)
  WRITE(ICHR,9=zerodot3)(RDAT(I),I=2,7)
ENDIF
GOTO 2000
C
C**** SET SEGMENT PRIORITY
1094 IF (ICNV.EQ.ICNVCM) THEN
  READ (ICHR,9=zerodot2)IDAT(1)
  READ (ICHR,9=zerodot3)RDAT(2)
ELSE
  WRITE(ICHR,9=zerodot2)IDAT(1)
  WRITE(ICHR,9=zerodot3)RDAT(2)
ENDIF
```
Example programs

GOTO 2000
C
C**** USER ITEMS
1100 IF (ICNV.EQ.ICNVCM) THEN
     READ (ICH,901)(CDAT(I),I=1,ILEN)
ELSE
     WRITE(ICH,901)(CDAT(I),I=1,ILEN)
ENDIF
GOTO 2000
C
C**** WRITE ITEM DATA TO METAFILE (IF APPROPRIATE)
2000 IF (ICNV.EQ.ICNVCM) THEN
     IF (ITYP.LE.MAXTYP) THEN
         CALL GIITM (ITYP,ILEN,MAXREC,DATREC)
     ELSE
         CALL GWITM (IWKID,ITYP,ILEN,MAXREC,DATREC)
     ENDIF
ENDIF
C
C**** CHECK FOR LAST METAFILE ITEM
600 IF (ITYP.NE.0) GOTO 200
C
C
C ******************************************************************************6800000
C ** TERMINATION **97000000
C ******************************************************************************97200000
C
C**** SUCCESSFUL COMPLETION
97600000
WRITE(6,3100)INUM
3100 FORMAT(1X,'EXECUTION COMPLETED SUCCESSFULLY, ',I6, & ' METAFILE ITEMS PROCESSED. ')
C
C**** TERMINATE GKS
98600000
3200 IF (ICNV.EQ.ICNVCM) CALL GDWK (IWKID)
     CALL GCLWK (IWKID)
     CALL GCLKS
C
3300 STOP
END
Example programs
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

<table>
<thead>
<tr>
<th>Call Name</th>
<th>RCP Code (Hex.)</th>
<th>RCP Code (Decimal)</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>GACTM</td>
<td>38006A00</td>
<td>939551232</td>
<td>Accumulate transformation matrix</td>
</tr>
<tr>
<td>GACWK</td>
<td>38004000</td>
<td>939525120</td>
<td>Activate workstation</td>
</tr>
<tr>
<td>GASSWK</td>
<td>38003000</td>
<td>939539712</td>
<td>Associate segment with workstation</td>
</tr>
<tr>
<td>GCA</td>
<td>38001000</td>
<td>939528192</td>
<td>Cell array</td>
</tr>
<tr>
<td>GCLKS</td>
<td>38000100</td>
<td>939524352</td>
<td>Close GKS</td>
</tr>
<tr>
<td>GCLRWK</td>
<td>38000600</td>
<td>939526353</td>
<td>Clear workstation</td>
</tr>
<tr>
<td>GCLSG</td>
<td>38003900</td>
<td>939536888</td>
<td>Close segment</td>
</tr>
<tr>
<td>GCLWK</td>
<td>38000300</td>
<td>939524864</td>
<td>Close workstation</td>
</tr>
<tr>
<td>GCRSG</td>
<td>38003500</td>
<td>939536432</td>
<td>Create segment</td>
</tr>
<tr>
<td>GCSWK</td>
<td>38003E00</td>
<td>939539688</td>
<td>Copy segment to workstation</td>
</tr>
<tr>
<td>GDANK</td>
<td>38000500</td>
<td>939525376</td>
<td>Deactivate workstation</td>
</tr>
<tr>
<td>GDG</td>
<td>38003800</td>
<td>939539200</td>
<td>Delete segment</td>
</tr>
<tr>
<td>GDSWK</td>
<td>38003C00</td>
<td>939539456</td>
<td>Delete segment from workstation</td>
</tr>
<tr>
<td>GECLKS</td>
<td>38000F00</td>
<td>939585536</td>
<td>Emergency close GKS</td>
</tr>
<tr>
<td>GERLOG</td>
<td>38000F10</td>
<td>939585792</td>
<td>Error logging</td>
</tr>
<tr>
<td>GESC</td>
<td>38000800</td>
<td>939526912</td>
<td>Escape</td>
</tr>
<tr>
<td>GEVTM</td>
<td>38006900</td>
<td>939550976</td>
<td>Evaluate transformation matrix</td>
</tr>
<tr>
<td>GFA</td>
<td>3800FF00</td>
<td>939527936</td>
<td>Fill area</td>
</tr>
<tr>
<td>GFLUSH</td>
<td>38005E00</td>
<td>939548160</td>
<td>Flush device events</td>
</tr>
<tr>
<td>GGD</td>
<td>38001100</td>
<td>939528448</td>
<td>Generalized drawing primitive</td>
</tr>
<tr>
<td>GGTCH</td>
<td>38006200</td>
<td>939549184</td>
<td>Get choice</td>
</tr>
<tr>
<td>GGTIPM</td>
<td>38006600</td>
<td>939550208</td>
<td>Get item type from GKSM</td>
</tr>
<tr>
<td>GGTLC</td>
<td>38005F00</td>
<td>939548416</td>
<td>Get locator</td>
</tr>
<tr>
<td>GTPK</td>
<td>38006300</td>
<td>939549440</td>
<td>Get pick</td>
</tr>
<tr>
<td>GTSK</td>
<td>38006000</td>
<td>939548672</td>
<td>Get stroke</td>
</tr>
<tr>
<td>GTSSTS</td>
<td>38006400</td>
<td>939549696</td>
<td>Get string</td>
</tr>
<tr>
<td>GTVL</td>
<td>38006100</td>
<td>939548928</td>
<td>Get valuator</td>
</tr>
<tr>
<td>GYTCH</td>
<td>38006800</td>
<td>939550720</td>
<td>Interpret item</td>
</tr>
<tr>
<td>GYNCH</td>
<td>38004800</td>
<td>939542528</td>
<td>Initialize choice</td>
</tr>
<tr>
<td>GINLC</td>
<td>38004500</td>
<td>939541760</td>
<td>Initialize locator</td>
</tr>
<tr>
<td>GINPK</td>
<td>38004900</td>
<td>939542784</td>
<td>Initialize pick</td>
</tr>
<tr>
<td>GINSNG</td>
<td>38003F00</td>
<td>939540224</td>
<td>Insert segment</td>
</tr>
<tr>
<td>GINSK</td>
<td>38004600</td>
<td>939542016</td>
<td>Initialize stroke</td>
</tr>
<tr>
<td>GINSSTS</td>
<td>38004A00</td>
<td>939543040</td>
<td>Initialize string</td>
</tr>
<tr>
<td>GINVL</td>
<td>38004700</td>
<td>939542272</td>
<td>Initialize valuator</td>
</tr>
<tr>
<td>GMSGS</td>
<td>38000A00</td>
<td>939526656</td>
<td>Message</td>
</tr>
<tr>
<td>G0PKS</td>
<td>38000000</td>
<td>939524096</td>
<td>Open GKS</td>
</tr>
<tr>
<td>G0PKW</td>
<td>38000200</td>
<td>939524608</td>
<td>Open workstation</td>
</tr>
<tr>
<td>GPL</td>
<td>38000C00</td>
<td>939527168</td>
<td>Polylne</td>
</tr>
<tr>
<td>GPM</td>
<td>38000000</td>
<td>939527424</td>
<td>Polymarker</td>
</tr>
<tr>
<td>GPRDCS</td>
<td>38006800</td>
<td>939551488</td>
<td>Pack data record</td>
</tr>
<tr>
<td>GQACWK</td>
<td>38008600</td>
<td>939558400</td>
<td>Inquire set member of active workstations</td>
</tr>
<tr>
<td>GQASF</td>
<td>38001A00</td>
<td>939565312</td>
<td>Inquire aspect source flags</td>
</tr>
<tr>
<td>GQASKW</td>
<td>3800DE00</td>
<td>939580928</td>
<td>Inquire set member of associated workstations</td>
</tr>
</tbody>
</table>
### RCP codes

<table>
<thead>
<tr>
<th>Call Name</th>
<th>RCP Code (Hex.)</th>
<th>RCP Code (Decimal)</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQCF</td>
<td>38000000</td>
<td>939577344</td>
<td>Inquire color facilities</td>
</tr>
<tr>
<td>GQCHB</td>
<td>38008D00</td>
<td>939560192</td>
<td>Inquire character base vector</td>
</tr>
<tr>
<td>GQCHH</td>
<td>38008A00</td>
<td>939559424</td>
<td>Inquire character height</td>
</tr>
<tr>
<td>GQCHS</td>
<td>3800BE00</td>
<td>939572736</td>
<td>Inquire choice device state</td>
</tr>
<tr>
<td>GQCHSP</td>
<td>38009C00</td>
<td>939564032</td>
<td>Inquire character spacing</td>
</tr>
<tr>
<td>GQCHUP</td>
<td>38008B00</td>
<td>939559680</td>
<td>Inquire character up vector</td>
</tr>
<tr>
<td>GQCHW</td>
<td>38008C00</td>
<td>939559936</td>
<td>Inquire character width</td>
</tr>
<tr>
<td>GQCHXP</td>
<td>38009B00</td>
<td>939563776</td>
<td>Inquire character expansion factor</td>
</tr>
<tr>
<td>GQLIP</td>
<td>3800A500</td>
<td>939566336</td>
<td>Inquire clipping indicator</td>
</tr>
<tr>
<td>GQCMN</td>
<td>3800A200</td>
<td>939565568</td>
<td>Inquire current normalization transformation number</td>
</tr>
<tr>
<td>GQCR</td>
<td>3800B800</td>
<td>939571200</td>
<td>Inquire color representation</td>
</tr>
<tr>
<td>GQCH</td>
<td>38008B00</td>
<td>939580160</td>
<td>Inquire default choice device data</td>
</tr>
<tr>
<td>GQDS</td>
<td>3800C500</td>
<td>939574528</td>
<td>Inquire default deferral state values</td>
</tr>
<tr>
<td>GQLC</td>
<td>38008000</td>
<td>939579392</td>
<td>Inquire default locator device data</td>
</tr>
<tr>
<td>GQDP</td>
<td>3800D000</td>
<td>939580416</td>
<td>Inquire default pick device data</td>
</tr>
<tr>
<td>GQDSGA</td>
<td>38000600</td>
<td>939578880</td>
<td>Inquire dynamic modification of segment attributes</td>
</tr>
<tr>
<td>GQSK</td>
<td>38000900</td>
<td>939579648</td>
<td>Inquire default stroke device data</td>
</tr>
<tr>
<td>GQSP</td>
<td>3800C300</td>
<td>939574016</td>
<td>Inquire display space size</td>
</tr>
<tr>
<td>GQST</td>
<td>3800D000</td>
<td>939580672</td>
<td>Inquire default string device data</td>
</tr>
<tr>
<td>GQVL</td>
<td>3800A000</td>
<td>939579904</td>
<td>Inquire default valuator device data</td>
</tr>
<tr>
<td>GQDKA</td>
<td>3800C400</td>
<td>939574272</td>
<td>Inquire dynamic modification of workstation attributes</td>
</tr>
<tr>
<td>GQECI</td>
<td>3800B700</td>
<td>939570944</td>
<td>Inquire list element of color indexes</td>
</tr>
<tr>
<td>GQFAI</td>
<td>3800B300</td>
<td>939569920</td>
<td>Inquire list element of fill area indexes</td>
</tr>
<tr>
<td>GQEGOP</td>
<td>38000200</td>
<td>939577856</td>
<td>Inquire list element of available generalized drawing primitives</td>
</tr>
<tr>
<td>GQENTN</td>
<td>3800A300</td>
<td>939565824</td>
<td>Inquire list element of normalization transformation numbers</td>
</tr>
<tr>
<td>GQPAI</td>
<td>3800B500</td>
<td>939570432</td>
<td>Inquire list element of pattern indexes</td>
</tr>
<tr>
<td>GQPLI</td>
<td>3800AC00</td>
<td>939568128</td>
<td>Inquire list element of polyline indexes</td>
</tr>
<tr>
<td>GQPMI</td>
<td>3800AE00</td>
<td>939568640</td>
<td>Inquire list element of polymarker indexes</td>
</tr>
<tr>
<td>GQETXI</td>
<td>3800B000</td>
<td>939569152</td>
<td>Inquire list element of text indexes</td>
</tr>
<tr>
<td>GQEWK</td>
<td>3800B200</td>
<td>939557376</td>
<td>Inquire list element of available workstation types</td>
</tr>
<tr>
<td>GQFA</td>
<td>3800A000</td>
<td>939565556</td>
<td>Inquire fill area color index</td>
</tr>
<tr>
<td>GQFCI</td>
<td>3800CC00</td>
<td>939576320</td>
<td>Inquire fill area facilities</td>
</tr>
<tr>
<td>GQFAI</td>
<td>38009000</td>
<td>939560960</td>
<td>Inquire fill area index</td>
</tr>
<tr>
<td>GQFAIS</td>
<td>38009C00</td>
<td>939564544</td>
<td>Inquire fill area interior style</td>
</tr>
<tr>
<td>GQFAR</td>
<td>3800B400</td>
<td>939570176</td>
<td>Inquire fill area representation</td>
</tr>
<tr>
<td>GQFASI</td>
<td>38009F00</td>
<td>939564800</td>
<td>Inquire fill area style index</td>
</tr>
<tr>
<td>GQGD</td>
<td>38000300</td>
<td>939578112</td>
<td>Inquire generalized drawing primitive</td>
</tr>
<tr>
<td>GQIOV</td>
<td>3800E300</td>
<td>939582208</td>
<td>Inquire input queue overflow</td>
</tr>
<tr>
<td>GQLCS</td>
<td>3800B800</td>
<td>939571968</td>
<td>Inquire locator device state</td>
</tr>
<tr>
<td>GQLI</td>
<td>38000700</td>
<td>939579136</td>
<td>Inquire number of available logical input devices</td>
</tr>
<tr>
<td>GQLN</td>
<td>38009400</td>
<td>939561984</td>
<td>Inquire linetype</td>
</tr>
<tr>
<td>GQLVKS</td>
<td>38008100</td>
<td>939557120</td>
<td>Inquire level of GKS</td>
</tr>
<tr>
<td>GQLMK</td>
<td>38000400</td>
<td>939578368</td>
<td>Inquire maximum length of workstation state tables</td>
</tr>
<tr>
<td>GQLMSC</td>
<td>38009500</td>
<td>939562240</td>
<td>Inquire linewidth scale factor</td>
</tr>
<tr>
<td>GQMK</td>
<td>38009700</td>
<td>939562752</td>
<td>Inquire marker type</td>
</tr>
<tr>
<td>GQMKSC</td>
<td>38009800</td>
<td>939563008</td>
<td>Inquire marker size scale factor</td>
</tr>
<tr>
<td>GQMNMTN</td>
<td>38008400</td>
<td>939557888</td>
<td>Inquire maximum normalization transformation number</td>
</tr>
<tr>
<td>GQNT</td>
<td>3800A400</td>
<td>939566880</td>
<td>Inquire normalization transformation</td>
</tr>
<tr>
<td>GQOPS</td>
<td>38008000</td>
<td>939556864</td>
<td>Inquire operating state value</td>
</tr>
<tr>
<td>GQPSG</td>
<td>3800A600</td>
<td>939566592</td>
<td>Inquire name of open segment</td>
</tr>
<tr>
<td>GQPSKG</td>
<td>38008500</td>
<td>93958144</td>
<td>Inquire set member of open workstations</td>
</tr>
<tr>
<td>GQPA</td>
<td>38009100</td>
<td>939561216</td>
<td>Inquire pattern size</td>
</tr>
<tr>
<td>GQPAF</td>
<td>3800CE00</td>
<td>939576832</td>
<td>Inquire pattern facilities</td>
</tr>
<tr>
<td>GQPAR</td>
<td>3800B600</td>
<td>939570688</td>
<td>Inquire pattern representation</td>
</tr>
<tr>
<td>GQPARF</td>
<td>38009200</td>
<td>939561472</td>
<td>Inquire pattern reference point</td>
</tr>
<tr>
<td>GQPCR</td>
<td>38000100</td>
<td>939577600</td>
<td>Inquire predefined color representation</td>
</tr>
<tr>
<td>Call Name</td>
<td>RCP Code (Hex.)</td>
<td>RCP Code (Decimal)</td>
<td>Function</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------</td>
<td>-------------------</td>
<td>----------</td>
</tr>
<tr>
<td>GQPFAR</td>
<td>3800CD00</td>
<td>939576576</td>
<td>Inquire predefined fill area representation</td>
</tr>
<tr>
<td>GQPKID</td>
<td>38009300</td>
<td>939561728</td>
<td>Inquire pick identifier</td>
</tr>
<tr>
<td>GQPKS</td>
<td>3800BF00</td>
<td>939572992</td>
<td>Inquire pick device state</td>
</tr>
<tr>
<td>GQPLCI</td>
<td>38009600</td>
<td>939562496</td>
<td>Inquire polyline color index</td>
</tr>
<tr>
<td>GQPLF</td>
<td>3800C600</td>
<td>939574784</td>
<td>Inquire polyline facilities</td>
</tr>
<tr>
<td>GQPLI</td>
<td>3800B700</td>
<td>939558656</td>
<td>Inquire polyline index</td>
</tr>
<tr>
<td>GQPLR</td>
<td>3800A000</td>
<td>939568384</td>
<td>Inquire polyline representation</td>
</tr>
<tr>
<td>GQPMCI</td>
<td>38009900</td>
<td>939563264</td>
<td>Inquire polymarker color index</td>
</tr>
<tr>
<td>GQPMF</td>
<td>3800C800</td>
<td>939575296</td>
<td>Inquire polymarker facilities</td>
</tr>
<tr>
<td>GQPMI</td>
<td>3800B800</td>
<td>939558912</td>
<td>Inquire polymarker index</td>
</tr>
<tr>
<td>GQPMR</td>
<td>3800AF00</td>
<td>939568896</td>
<td>Inquire polymarker representation</td>
</tr>
<tr>
<td>GQPPAR</td>
<td>3800CF00</td>
<td>939577088</td>
<td>Inquire predefined pattern representation</td>
</tr>
<tr>
<td>GQPPLA</td>
<td>3800C700</td>
<td>939575040</td>
<td>Inquire predefined polyline representation</td>
</tr>
<tr>
<td>GQPPMR</td>
<td>3800C900</td>
<td>939575552</td>
<td>Inquire predefined polymarker representation</td>
</tr>
<tr>
<td>GQPTXR</td>
<td>3800CB00</td>
<td>939576064</td>
<td>Inquire predefined text representation</td>
</tr>
<tr>
<td>GQPX</td>
<td>3800E200</td>
<td>939581952</td>
<td>Inquire pixel</td>
</tr>
<tr>
<td>GQPXA</td>
<td>3800E100</td>
<td>939581696</td>
<td>Inquire pixel array</td>
</tr>
<tr>
<td>GQPXAD</td>
<td>3800E000</td>
<td>939581440</td>
<td>Inquire pixel array dimensions</td>
</tr>
<tr>
<td>GQSGA</td>
<td>3800DF00</td>
<td>939581184</td>
<td>Inquire segment attributes</td>
</tr>
<tr>
<td>GQSGP</td>
<td>3800D500</td>
<td>939578624</td>
<td>Inquire number of segment priorities supported</td>
</tr>
<tr>
<td>GQSGUS</td>
<td>3800A700</td>
<td>939566848</td>
<td>Inquire set member of segment names in use</td>
</tr>
<tr>
<td>GQSGWK</td>
<td>3800BA00</td>
<td>939571712</td>
<td>Inquire set member of segment names on workstation</td>
</tr>
<tr>
<td>GQSIM</td>
<td>3800A800</td>
<td>939567104</td>
<td>Inquire more simultaneous events</td>
</tr>
<tr>
<td>GQSJS</td>
<td>3800BC00</td>
<td>939572224</td>
<td>Inquire stroke device state</td>
</tr>
<tr>
<td>GQSSTSS</td>
<td>3800C000</td>
<td>939573248</td>
<td>Inquire string device state</td>
</tr>
<tr>
<td>GQTXML</td>
<td>3800AF00</td>
<td>939560704</td>
<td>Inquire text alignment</td>
</tr>
<tr>
<td>GQTXCI</td>
<td>3800D000</td>
<td>939564288</td>
<td>Inquire text color index</td>
</tr>
<tr>
<td>GQTXF</td>
<td>3800CA00</td>
<td>939575808</td>
<td>Inquire text facilities</td>
</tr>
<tr>
<td>GQTXFP</td>
<td>38009A00</td>
<td>939563520</td>
<td>Inquire text font and precision</td>
</tr>
<tr>
<td>GQTX1</td>
<td>38008900</td>
<td>939559168</td>
<td>Inquire text index</td>
</tr>
<tr>
<td>GQTXP</td>
<td>38008E00</td>
<td>939560448</td>
<td>Inquire text path</td>
</tr>
<tr>
<td>GQTXR</td>
<td>3800B100</td>
<td>939569408</td>
<td>Inquire text representation</td>
</tr>
<tr>
<td>GQTXFS</td>
<td>3800B200</td>
<td>939569664</td>
<td>Inquire text extent</td>
</tr>
<tr>
<td>GQVLS</td>
<td>3800B000</td>
<td>939572480</td>
<td>Inquire valuator device state</td>
</tr>
<tr>
<td>GQMVC</td>
<td>3800A900</td>
<td>939567360</td>
<td>Inquire workstation connection and type</td>
</tr>
<tr>
<td>GQMKCA</td>
<td>3800C100</td>
<td>939573504</td>
<td>Inquire workstation category</td>
</tr>
<tr>
<td>GQMKCL</td>
<td>3800C200</td>
<td>939573760</td>
<td>Inquire workstation classification</td>
</tr>
<tr>
<td>GQMHDU</td>
<td>3800AB00</td>
<td>939567872</td>
<td>Inquire workstation deferral and update states</td>
</tr>
<tr>
<td>GQMWS</td>
<td>3800B300</td>
<td>939556732</td>
<td>Inquire workstation maximum numbers</td>
</tr>
<tr>
<td>GQKMS</td>
<td>3800AA00</td>
<td>939567616</td>
<td>Inquire workstation state</td>
</tr>
<tr>
<td>GQKNT</td>
<td>3800B900</td>
<td>939571456</td>
<td>Inquire workstation transformation</td>
</tr>
<tr>
<td>GRDITM</td>
<td>38006700</td>
<td>939550464</td>
<td>Read item from GKSM</td>
</tr>
<tr>
<td>GRENSG</td>
<td>38003A00</td>
<td>939538944</td>
<td>Rename segment</td>
</tr>
<tr>
<td>GRQCH</td>
<td>38005400</td>
<td>939545600</td>
<td>Request choice</td>
</tr>
<tr>
<td>GRQLC</td>
<td>38005100</td>
<td>939544832</td>
<td>Request locator</td>
</tr>
<tr>
<td>GRQPK</td>
<td>38005500</td>
<td>939548856</td>
<td>Request pick</td>
</tr>
<tr>
<td>GRQSK</td>
<td>38005200</td>
<td>939545088</td>
<td>Request stroke</td>
</tr>
<tr>
<td>GRQSTS</td>
<td>38005600</td>
<td>939546112</td>
<td>Request string</td>
</tr>
<tr>
<td>GRQVL</td>
<td>38005300</td>
<td>939545344</td>
<td>Request valuator</td>
</tr>
<tr>
<td>GRGSKW</td>
<td>38007000</td>
<td>939525888</td>
<td>Redraw all segments on workstation</td>
</tr>
<tr>
<td>GSA5F</td>
<td>38002900</td>
<td>939534592</td>
<td>Set aspect source flags</td>
</tr>
<tr>
<td>GSCHH</td>
<td>38001F00</td>
<td>939532032</td>
<td>Set character height</td>
</tr>
<tr>
<td>GSCHM</td>
<td>38004E00</td>
<td>939544064</td>
<td>Set choice mode</td>
</tr>
<tr>
<td>GSCHP</td>
<td>38001D00</td>
<td>939531520</td>
<td>Set character spacing</td>
</tr>
<tr>
<td>GSCHPUP</td>
<td>38002000</td>
<td>939532288</td>
<td>Set character up vector</td>
</tr>
<tr>
<td>GSCHRXP</td>
<td>38001C00</td>
<td>939531264</td>
<td>Set character expansion factor</td>
</tr>
<tr>
<td>Call Name</td>
<td>RCP Code (Hex.)</td>
<td>RCP Code (Decimal)</td>
<td>Function</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------</td>
<td>--------------------</td>
<td>----------</td>
</tr>
<tr>
<td>GSCLIP</td>
<td>38003500</td>
<td>939537664</td>
<td>Set clipping indicator</td>
</tr>
<tr>
<td>GSCR</td>
<td>38003000</td>
<td>939536384</td>
<td>Set color representation</td>
</tr>
<tr>
<td>GSDS</td>
<td>38000900</td>
<td>939526400</td>
<td>Set deferral state</td>
</tr>
<tr>
<td>GSDTEC</td>
<td>38004400</td>
<td>939541504</td>
<td>Set detectability</td>
</tr>
<tr>
<td>GSELNT</td>
<td>38003400</td>
<td>939537408</td>
<td>Select normalization transformation</td>
</tr>
<tr>
<td>GSFAI</td>
<td>38002600</td>
<td>939533824</td>
<td>Set fill area color index</td>
</tr>
<tr>
<td>GSFAIS</td>
<td>38002300</td>
<td>939533056</td>
<td>Set fill area index</td>
</tr>
<tr>
<td>GSFAISI</td>
<td>38002400</td>
<td>939533312</td>
<td>Set fill area interior style</td>
</tr>
<tr>
<td>GSFAI</td>
<td>38002100</td>
<td>939535872</td>
<td>Set fill area representation</td>
</tr>
<tr>
<td>GSFAI</td>
<td>38002500</td>
<td>939533568</td>
<td>Set fill area style index</td>
</tr>
<tr>
<td>GSHELIT</td>
<td>38004200</td>
<td>939540992</td>
<td>Set highlighting</td>
</tr>
<tr>
<td>GSLCM</td>
<td>38004800</td>
<td>939543296</td>
<td>Set locator mode</td>
</tr>
<tr>
<td>GSLN</td>
<td>38001300</td>
<td>939528960</td>
<td>Set linetype</td>
</tr>
<tr>
<td>GSLNCS</td>
<td>38001400</td>
<td>939529216</td>
<td>Set linewidth scale factor</td>
</tr>
<tr>
<td>GSMCH</td>
<td>38005A00</td>
<td>939547136</td>
<td>Sample choice</td>
</tr>
<tr>
<td>GSMK</td>
<td>38001700</td>
<td>939529984</td>
<td>Set marker type</td>
</tr>
<tr>
<td>GSMKSC</td>
<td>38001800</td>
<td>939530240</td>
<td>Set marker size scale factor</td>
</tr>
<tr>
<td>GSMLC</td>
<td>38005700</td>
<td>939546368</td>
<td>Sample locator</td>
</tr>
<tr>
<td>GSMPK</td>
<td>38005B00</td>
<td>939547392</td>
<td>Sample pick</td>
</tr>
<tr>
<td>GSMSK</td>
<td>38005800</td>
<td>939546624</td>
<td>Sample stroke</td>
</tr>
<tr>
<td>GSMSTS</td>
<td>38005C00</td>
<td>939547648</td>
<td>Sample string</td>
</tr>
<tr>
<td>GSML</td>
<td>38005900</td>
<td>939546880</td>
<td>Sample valuator</td>
</tr>
<tr>
<td>GSPA</td>
<td>38002700</td>
<td>939534080</td>
<td>Set pattern size</td>
</tr>
<tr>
<td>GSPAR</td>
<td>38002F00</td>
<td>939536128</td>
<td>Set pattern representation</td>
</tr>
<tr>
<td>GSPARF</td>
<td>38002800</td>
<td>939534336</td>
<td>Set pattern reference point</td>
</tr>
<tr>
<td>GSPKID</td>
<td>38002A00</td>
<td>939534848</td>
<td>Set pick identifier</td>
</tr>
<tr>
<td>GSPKM</td>
<td>38004F00</td>
<td>939544320</td>
<td>Set pick mode</td>
</tr>
<tr>
<td>GSPLCI</td>
<td>38001500</td>
<td>939529472</td>
<td>Set polyline color index</td>
</tr>
<tr>
<td>GSPLI</td>
<td>38001200</td>
<td>939528704</td>
<td>Set polyline index</td>
</tr>
<tr>
<td>GSPLR</td>
<td>38002B00</td>
<td>939535104</td>
<td>Set polyline representation</td>
</tr>
<tr>
<td>GSPMCI</td>
<td>38001900</td>
<td>939530496</td>
<td>Set polymarker color index</td>
</tr>
<tr>
<td>GSPMI</td>
<td>38001600</td>
<td>939529728</td>
<td>Set polymarker index</td>
</tr>
<tr>
<td>GSPMR</td>
<td>38002C00</td>
<td>939535360</td>
<td>Set polymarker representation</td>
</tr>
<tr>
<td>GSSP</td>
<td>38004300</td>
<td>939541248</td>
<td>Set segment priority</td>
</tr>
<tr>
<td>GSSGT</td>
<td>38004000</td>
<td>939540480</td>
<td>Set segment transformation</td>
</tr>
<tr>
<td>GSSKM</td>
<td>38004C00</td>
<td>939543552</td>
<td>Set stroke mode</td>
</tr>
<tr>
<td>GSSMT</td>
<td>38005000</td>
<td>939544576</td>
<td>Set string mode</td>
</tr>
<tr>
<td>GSTXAL</td>
<td>38002200</td>
<td>939532800</td>
<td>Set text alignment</td>
</tr>
<tr>
<td>GSTXI</td>
<td>38001E00</td>
<td>939531776</td>
<td>Set text color index</td>
</tr>
<tr>
<td>GSTXFP</td>
<td>38001800</td>
<td>939531008</td>
<td>Set text font and precision</td>
</tr>
<tr>
<td>GSTXI</td>
<td>38001A00</td>
<td>939530752</td>
<td>Set text index</td>
</tr>
<tr>
<td>GSTXP</td>
<td>38002100</td>
<td>939532544</td>
<td>Set text path</td>
</tr>
<tr>
<td>GSTXR</td>
<td>38002000</td>
<td>939535616</td>
<td>Set text representation</td>
</tr>
<tr>
<td>GSVIS</td>
<td>38004100</td>
<td>939540736</td>
<td>Set visibility</td>
</tr>
<tr>
<td>GSVL</td>
<td>38004D00</td>
<td>939543808</td>
<td>Set valuators mode</td>
</tr>
<tr>
<td>GSVP</td>
<td>38003200</td>
<td>939536896</td>
<td>Set viewport</td>
</tr>
<tr>
<td>GSVPIP</td>
<td>38003300</td>
<td>939537152</td>
<td>Set viewport input priority</td>
</tr>
<tr>
<td>GSWKVP</td>
<td>38003700</td>
<td>939538176</td>
<td>Set workstation viewport</td>
</tr>
<tr>
<td>GSWKMN</td>
<td>38003600</td>
<td>939537920</td>
<td>Set workstation window</td>
</tr>
<tr>
<td>GSWN</td>
<td>38003100</td>
<td>939536640</td>
<td>Set window</td>
</tr>
<tr>
<td>GTXS</td>
<td>38000E00</td>
<td>939527680</td>
<td>Text</td>
</tr>
<tr>
<td>GURECS</td>
<td>38006C00</td>
<td>939551744</td>
<td>Unpack data record</td>
</tr>
<tr>
<td>GUNK</td>
<td>38000E00</td>
<td>939526144</td>
<td>Update workstation</td>
</tr>
<tr>
<td>GWAIT</td>
<td>38005D00</td>
<td>939547904</td>
<td>Await event</td>
</tr>
<tr>
<td>GWITM</td>
<td>38006500</td>
<td>939549952</td>
<td>Write item to GKSM</td>
</tr>
</tbody>
</table>
### RCP codes ordered by code value

<table>
<thead>
<tr>
<th>RCP Code (Hex.)</th>
<th>RCP Code (Decimal)</th>
<th>Call Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>38000000</td>
<td>939524096</td>
<td>GDPKS</td>
<td>Open GKS</td>
</tr>
<tr>
<td>38000100</td>
<td>939524352</td>
<td>GCLKS</td>
<td>Close GKS</td>
</tr>
<tr>
<td>38000200</td>
<td>939524608</td>
<td>GOPWK</td>
<td>Open workstation</td>
</tr>
<tr>
<td>38000300</td>
<td>939524864</td>
<td>GCLKWK</td>
<td>Close workstation</td>
</tr>
<tr>
<td>38000400</td>
<td>939525120</td>
<td>GACWK</td>
<td>Activate workstation</td>
</tr>
<tr>
<td>38000500</td>
<td>939525376</td>
<td>GDAWK</td>
<td>Deactivate workstation</td>
</tr>
<tr>
<td>38000600</td>
<td>939525632</td>
<td>GCLRWK</td>
<td>Clear workstation</td>
</tr>
<tr>
<td>38000700</td>
<td>939525888</td>
<td>GRSGWK</td>
<td>Redraw all segments on workstation</td>
</tr>
<tr>
<td>38000800</td>
<td>939526144</td>
<td>GUNK</td>
<td>Update workstation</td>
</tr>
<tr>
<td>38000900</td>
<td>939526400</td>
<td>GSDS</td>
<td>Set deferral state</td>
</tr>
<tr>
<td>38000A00</td>
<td>939526656</td>
<td>GMSGS</td>
<td>Message</td>
</tr>
<tr>
<td>38000B00</td>
<td>939526912</td>
<td>GESC</td>
<td>Escape</td>
</tr>
<tr>
<td>38000C00</td>
<td>939527168</td>
<td>GPL</td>
<td>Polyline</td>
</tr>
<tr>
<td>38000D00</td>
<td>939527424</td>
<td>GPM</td>
<td>Polymarker</td>
</tr>
<tr>
<td>38000E00</td>
<td>939527680</td>
<td>GTXS</td>
<td>Text</td>
</tr>
<tr>
<td>38000F00</td>
<td>939527936</td>
<td>GFA</td>
<td>Fill area</td>
</tr>
<tr>
<td>38001000</td>
<td>939528192</td>
<td>GCA</td>
<td>Cell array</td>
</tr>
<tr>
<td>38001100</td>
<td>939528448</td>
<td>GGDP</td>
<td>Generalized drawing primitive</td>
</tr>
<tr>
<td>38001200</td>
<td>939528704</td>
<td>GSPLI</td>
<td>Set polyline index</td>
</tr>
<tr>
<td>38001300</td>
<td>939528960</td>
<td>GSLN</td>
<td>Set linetype</td>
</tr>
<tr>
<td>38001400</td>
<td>939529216</td>
<td>GSLWSC</td>
<td>Set linewidth scale factor</td>
</tr>
<tr>
<td>38001500</td>
<td>939529472</td>
<td>GSPLCI</td>
<td>Set polyline color index</td>
</tr>
<tr>
<td>38001600</td>
<td>939529728</td>
<td>GSPMI</td>
<td>Set polymarker index</td>
</tr>
<tr>
<td>38001700</td>
<td>939529984</td>
<td>GSMK</td>
<td>Set marker type</td>
</tr>
<tr>
<td>38001800</td>
<td>939530240</td>
<td>GSMKSC</td>
<td>Set marker size scale factor</td>
</tr>
<tr>
<td>38001900</td>
<td>939530496</td>
<td>GSPMCI</td>
<td>Set polymarker color index</td>
</tr>
<tr>
<td>38001A00</td>
<td>939530752</td>
<td>GSTXI</td>
<td>Set text index</td>
</tr>
<tr>
<td>38001B00</td>
<td>939531008</td>
<td>GSTXFP</td>
<td>Set text font and precision</td>
</tr>
<tr>
<td>38001C00</td>
<td>939531264</td>
<td>GSCHXP</td>
<td>Set character expansion factor</td>
</tr>
<tr>
<td>38001D00</td>
<td>939531520</td>
<td>GSCHSP</td>
<td>Set character spacing</td>
</tr>
<tr>
<td>38001E00</td>
<td>939531776</td>
<td>GSTXCI</td>
<td>Set text color index</td>
</tr>
<tr>
<td>38001F00</td>
<td>939532032</td>
<td>GSCHH</td>
<td>Set character height</td>
</tr>
<tr>
<td>38002000</td>
<td>939532288</td>
<td>GSCHUP</td>
<td>Set character up vector</td>
</tr>
<tr>
<td>38002100</td>
<td>939532544</td>
<td>GSTXP</td>
<td>Set text path</td>
</tr>
<tr>
<td>38002200</td>
<td>939532800</td>
<td>GSTXAL</td>
<td>Set text alignment</td>
</tr>
<tr>
<td>38002300</td>
<td>939533056</td>
<td>GSFAI</td>
<td>Set fill area index</td>
</tr>
<tr>
<td>38002400</td>
<td>939533312</td>
<td>GSFAIS</td>
<td>Set fill area interior style</td>
</tr>
<tr>
<td>38002500</td>
<td>939533568</td>
<td>GSFASI</td>
<td>Set fill area style index</td>
</tr>
<tr>
<td>38002600</td>
<td>939533824</td>
<td>GSFACI</td>
<td>Set fill area color index</td>
</tr>
<tr>
<td>38002700</td>
<td>939534080</td>
<td>GSPA</td>
<td>Set pattern size</td>
</tr>
<tr>
<td>38002800</td>
<td>939534336</td>
<td>GSPARF</td>
<td>Set pattern reference point</td>
</tr>
<tr>
<td>38002900</td>
<td>939534592</td>
<td>GSASF</td>
<td>Set aspect source flags</td>
</tr>
<tr>
<td>38002A00</td>
<td>939534848</td>
<td>GSPKID</td>
<td>Set pick identifier</td>
</tr>
<tr>
<td>38002B00</td>
<td>939535104</td>
<td>GSPLR</td>
<td>Set polyline representation</td>
</tr>
<tr>
<td>38002C00</td>
<td>939535360</td>
<td>GSPMR</td>
<td>Set polymarker representation</td>
</tr>
<tr>
<td>38002D00</td>
<td>939535616</td>
<td>GSTXR</td>
<td>Set text representation</td>
</tr>
<tr>
<td>38002E00</td>
<td>939535872</td>
<td>GSFA</td>
<td>Set fill area representation</td>
</tr>
<tr>
<td>38002F00</td>
<td>939536128</td>
<td>GSPAR</td>
<td>Set pattern representation</td>
</tr>
<tr>
<td>38003000</td>
<td>939536384</td>
<td>GSCR</td>
<td>Set color representation</td>
</tr>
<tr>
<td>38003100</td>
<td>939536640</td>
<td>GSNW</td>
<td>Set window</td>
</tr>
<tr>
<td>38003200</td>
<td>939536896</td>
<td>GSVP</td>
<td>Set viewport</td>
</tr>
<tr>
<td>38003300</td>
<td>939537152</td>
<td>GSVPIP</td>
<td>Set viewport input priority</td>
</tr>
<tr>
<td>38003400</td>
<td>939537408</td>
<td>GSELTNT</td>
<td>Select normalization transformation</td>
</tr>
<tr>
<td>RCP Code (Hex.)</td>
<td>RCP Code (Decimal)</td>
<td>Call Name</td>
<td>Function</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>38003500</td>
<td>939537664</td>
<td>GSCLIP</td>
<td>Set clipping indicator</td>
</tr>
<tr>
<td>38003600</td>
<td>939537920</td>
<td>GSWKWN</td>
<td>Set workstation window</td>
</tr>
<tr>
<td>38003700</td>
<td>939538176</td>
<td>GSWKVP</td>
<td>Set workstation viewport</td>
</tr>
<tr>
<td>38003800</td>
<td>939538432</td>
<td>GCRSG</td>
<td>Create segment</td>
</tr>
<tr>
<td>38003900</td>
<td>939538688</td>
<td>GCLSG</td>
<td>Close segment</td>
</tr>
<tr>
<td>38003A00</td>
<td>939539444</td>
<td>GRENSG</td>
<td>Rename segment</td>
</tr>
<tr>
<td>38003B00</td>
<td>939539956</td>
<td>GSMGWK</td>
<td>Delete segment from workstation</td>
</tr>
<tr>
<td>38003C00</td>
<td>939539712</td>
<td>GAGSK</td>
<td>Associate segment with workstation</td>
</tr>
<tr>
<td>38003D00</td>
<td>939539368</td>
<td>GSMGWK</td>
<td>Delete segment</td>
</tr>
<tr>
<td>38003E00</td>
<td>939539640</td>
<td>GCSGWK</td>
<td>Copy segment to workstation</td>
</tr>
<tr>
<td>38003F00</td>
<td>939540224</td>
<td>GSINSG</td>
<td>Insert segment</td>
</tr>
<tr>
<td>38004000</td>
<td>939540480</td>
<td>GSSGT</td>
<td>Set segment transformation</td>
</tr>
<tr>
<td>38004100</td>
<td>939540736</td>
<td>GSVIS</td>
<td>Set visibility</td>
</tr>
<tr>
<td>38004200</td>
<td>939540992</td>
<td>GSHLIT</td>
<td>Set highlighting</td>
</tr>
<tr>
<td>38004300</td>
<td>939541248</td>
<td>GSGP</td>
<td>Set segment priority</td>
</tr>
<tr>
<td>38004400</td>
<td>939541504</td>
<td>GSDTEC</td>
<td>Set detectability</td>
</tr>
<tr>
<td>38004500</td>
<td>939541760</td>
<td>GINLC</td>
<td>Initialize locator</td>
</tr>
<tr>
<td>38004600</td>
<td>939542016</td>
<td>GINSK</td>
<td>Initialize stroke</td>
</tr>
<tr>
<td>38004700</td>
<td>939542272</td>
<td>GINVL</td>
<td>Initialize valuator</td>
</tr>
<tr>
<td>38004800</td>
<td>939542528</td>
<td>GINCH</td>
<td>Initialize choice</td>
</tr>
<tr>
<td>38004900</td>
<td>939542784</td>
<td>GINPK</td>
<td>Initialize pick</td>
</tr>
<tr>
<td>38004A00</td>
<td>939543040</td>
<td>GINSTS</td>
<td>Initialize string</td>
</tr>
<tr>
<td>38004B00</td>
<td>939543296</td>
<td>GSMLC</td>
<td>Set locator mode</td>
</tr>
<tr>
<td>38004C00</td>
<td>939543552</td>
<td>GSMSK</td>
<td>Set stroke mode</td>
</tr>
<tr>
<td>38004D00</td>
<td>939543808</td>
<td>GSVM</td>
<td>Set valuator mode</td>
</tr>
<tr>
<td>38004E00</td>
<td>939544064</td>
<td>GSCHM</td>
<td>Set choice mode</td>
</tr>
<tr>
<td>38004F00</td>
<td>939544320</td>
<td>GSPKM</td>
<td>Set pick mode</td>
</tr>
<tr>
<td>38005000</td>
<td>939544576</td>
<td>GSSTM</td>
<td>Set string mode</td>
</tr>
<tr>
<td>38005100</td>
<td>939544832</td>
<td>GRQLC</td>
<td>Request locator</td>
</tr>
<tr>
<td>38005200</td>
<td>939545088</td>
<td>GRQSK</td>
<td>Request stroke</td>
</tr>
<tr>
<td>38005300</td>
<td>939545344</td>
<td>GRQVL</td>
<td>Request valuator</td>
</tr>
<tr>
<td>38005400</td>
<td>939545600</td>
<td>GRQCH</td>
<td>Request choice</td>
</tr>
<tr>
<td>38005500</td>
<td>939545856</td>
<td>GRQPK</td>
<td>Request pick</td>
</tr>
<tr>
<td>38005600</td>
<td>939546112</td>
<td>GRQSTS</td>
<td>Request string</td>
</tr>
<tr>
<td>38005700</td>
<td>939546368</td>
<td>GSMLC</td>
<td>Sample locator</td>
</tr>
<tr>
<td>38005800</td>
<td>939546624</td>
<td>GSMSK</td>
<td>Sample stroke</td>
</tr>
<tr>
<td>38005900</td>
<td>939546880</td>
<td>GSML</td>
<td>Sample valuator</td>
</tr>
<tr>
<td>38005A00</td>
<td>939547136</td>
<td>GSCH</td>
<td>Sample choice</td>
</tr>
<tr>
<td>38005B00</td>
<td>939547392</td>
<td>GSPKM</td>
<td>Sample pick</td>
</tr>
<tr>
<td>38005C00</td>
<td>939547648</td>
<td>GSSTSM</td>
<td>Sample string</td>
</tr>
<tr>
<td>38005D00</td>
<td>939547904</td>
<td>GWAIT</td>
<td>Await event</td>
</tr>
<tr>
<td>38005E00</td>
<td>939548160</td>
<td>GFLUSH</td>
<td>Flush device events</td>
</tr>
<tr>
<td>38005F00</td>
<td>939548416</td>
<td>GTLC</td>
<td>Get locator</td>
</tr>
<tr>
<td>38006000</td>
<td>939548672</td>
<td>GTSK</td>
<td>Get stroke</td>
</tr>
<tr>
<td>38006100</td>
<td>939548928</td>
<td>GTVL</td>
<td>Get valuator</td>
</tr>
<tr>
<td>38006200</td>
<td>939549184</td>
<td>GTCH</td>
<td>Get choice</td>
</tr>
<tr>
<td>38006300</td>
<td>939549444</td>
<td>GTPK</td>
<td>Get pick</td>
</tr>
<tr>
<td>38006400</td>
<td>939549696</td>
<td>GGTST</td>
<td>Get string</td>
</tr>
<tr>
<td>38006500</td>
<td>939549952</td>
<td>GWITM</td>
<td>Write item to GKSM</td>
</tr>
<tr>
<td>38006600</td>
<td>939550208</td>
<td>GWTITM</td>
<td>Get item type from GKSM</td>
</tr>
<tr>
<td>38006700</td>
<td>939550464</td>
<td>GRTITM</td>
<td>Read item from GKSM</td>
</tr>
<tr>
<td>38006800</td>
<td>939550720</td>
<td>GIITM</td>
<td>Interpret item</td>
</tr>
<tr>
<td>38006900</td>
<td>939550976</td>
<td>GEVTM</td>
<td>Evaluate transformation matrix</td>
</tr>
<tr>
<td>38006A00</td>
<td>939551232</td>
<td>GACTM</td>
<td>Accumulate transformation matrix</td>
</tr>
<tr>
<td>38006B00</td>
<td>939551488</td>
<td>GPRECS</td>
<td>Pack data record</td>
</tr>
<tr>
<td>38006C00</td>
<td>939551744</td>
<td>GURECS</td>
<td>Unpack data record</td>
</tr>
<tr>
<td>RCP Code (Hex.)</td>
<td>RCP Code (Decimal)</td>
<td>Call Name</td>
<td>Function</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>38080000</td>
<td>389556864</td>
<td>GQOPS</td>
<td>Inquire operating state value</td>
</tr>
<tr>
<td>38081000</td>
<td>389557120</td>
<td>GOLVKS</td>
<td>Inquire level of GKS</td>
</tr>
<tr>
<td>38082000</td>
<td>389557376</td>
<td>GQEWK</td>
<td>Inquire list element of available workstation types</td>
</tr>
<tr>
<td>38083000</td>
<td>389557632</td>
<td>GQWKM</td>
<td>Inquire workstation maximum numbers</td>
</tr>
<tr>
<td>38084000</td>
<td>389557888</td>
<td>GQMTN</td>
<td>Inquire maximum normalization transformation number</td>
</tr>
<tr>
<td>38085000</td>
<td>389558144</td>
<td>GQOPWK</td>
<td>Inquire set member of open workstations</td>
</tr>
<tr>
<td>38086000</td>
<td>389558400</td>
<td>GQACWK</td>
<td>Inquire set member of active workstations</td>
</tr>
<tr>
<td>38087000</td>
<td>389558656</td>
<td>GQPLI</td>
<td>Inquire polyline index</td>
</tr>
<tr>
<td>38088000</td>
<td>389558912</td>
<td>GQPMI</td>
<td>Inquire polymarker index</td>
</tr>
<tr>
<td>38089000</td>
<td>389559168</td>
<td>GQTXI</td>
<td>Inquire text index</td>
</tr>
<tr>
<td>3808A000</td>
<td>389559424</td>
<td>GQCHH</td>
<td>Inquire character height</td>
</tr>
<tr>
<td>3808B000</td>
<td>389559680</td>
<td>GQCHUP</td>
<td>Inquire character up vector</td>
</tr>
<tr>
<td>3808C000</td>
<td>389559936</td>
<td>GQCHW</td>
<td>Inquire character width</td>
</tr>
<tr>
<td>3808D000</td>
<td>389560192</td>
<td>GQCHB</td>
<td>Inquire character base vector</td>
</tr>
<tr>
<td>3808E000</td>
<td>389560448</td>
<td>GQTXP</td>
<td>Inquire text path</td>
</tr>
<tr>
<td>3808F000</td>
<td>389560704</td>
<td>GQTXAL</td>
<td>Inquire text alignment</td>
</tr>
<tr>
<td>38090000</td>
<td>389560960</td>
<td>GQFAI</td>
<td>Inquire fill area index</td>
</tr>
<tr>
<td>38091000</td>
<td>389561216</td>
<td>GQPA</td>
<td>Inquire pattern size</td>
</tr>
<tr>
<td>38092000</td>
<td>389561472</td>
<td>GQPARF</td>
<td>Inquire pattern reference point</td>
</tr>
<tr>
<td>38093000</td>
<td>389561728</td>
<td>GPKID</td>
<td>Inquire pick identifier</td>
</tr>
<tr>
<td>38094000</td>
<td>389561984</td>
<td>GQLN</td>
<td>Inquire linetype</td>
</tr>
<tr>
<td>38095000</td>
<td>389562240</td>
<td>GQLWSC</td>
<td>Inquire linewidth scale factor</td>
</tr>
<tr>
<td>38096000</td>
<td>389562496</td>
<td>GQPLCI</td>
<td>Inquire polyline color index</td>
</tr>
<tr>
<td>38097000</td>
<td>389562752</td>
<td>GQMK</td>
<td>Inquire marker type</td>
</tr>
<tr>
<td>38098000</td>
<td>389563008</td>
<td>GQMKSC</td>
<td>Inquire marker size scale factor</td>
</tr>
<tr>
<td>38099000</td>
<td>389563264</td>
<td>GQPMCI</td>
<td>Inquire polymarker color index</td>
</tr>
<tr>
<td>3809A000</td>
<td>389563520</td>
<td>GQTXFP</td>
<td>Inquire text font and precision</td>
</tr>
<tr>
<td>3809B000</td>
<td>389563776</td>
<td>GQCHXP</td>
<td>Inquire character expansion factor</td>
</tr>
<tr>
<td>3809C000</td>
<td>389564032</td>
<td>GQCHSP</td>
<td>Inquire character spacing</td>
</tr>
<tr>
<td>3809D000</td>
<td>389564288</td>
<td>GQTXCI</td>
<td>Inquire text color index</td>
</tr>
<tr>
<td>3809E000</td>
<td>389564544</td>
<td>GQPAIS</td>
<td>Inquire fill area interior style</td>
</tr>
<tr>
<td>3809F000</td>
<td>389564800</td>
<td>GQFAXI</td>
<td>Inquire fill area style index</td>
</tr>
<tr>
<td>380A0000</td>
<td>389565056</td>
<td>GQFACI</td>
<td>Inquire fill area color index</td>
</tr>
<tr>
<td>380A1000</td>
<td>389565312</td>
<td>GQASF</td>
<td>Inquire aspect source flags</td>
</tr>
<tr>
<td>380A2000</td>
<td>389565568</td>
<td>GQCNTN</td>
<td>Inquire current normalization transformation number</td>
</tr>
<tr>
<td>380A3000</td>
<td>389565824</td>
<td>GQENTN</td>
<td>Inquire list element of normalization transformation numbers</td>
</tr>
<tr>
<td>380A4000</td>
<td>389566080</td>
<td>GQNT</td>
<td>Inquire normalization transformation</td>
</tr>
<tr>
<td>380A5000</td>
<td>389566336</td>
<td>GQCLIP</td>
<td>Inquire clipping indicator</td>
</tr>
<tr>
<td>380A6000</td>
<td>389566592</td>
<td>GQPSG</td>
<td>Inquire name of open segment</td>
</tr>
<tr>
<td>380A7000</td>
<td>389566848</td>
<td>GQSGUS</td>
<td>Inquire set member of segment names in use</td>
</tr>
<tr>
<td>380A8000</td>
<td>389567104</td>
<td>GQSIM</td>
<td>Inquire more simultaneous events</td>
</tr>
<tr>
<td>380A9000</td>
<td>389567360</td>
<td>GQWKC</td>
<td>Inquire workstation connection and type</td>
</tr>
<tr>
<td>380AA000</td>
<td>389567616</td>
<td>GQWKS</td>
<td>Inquire workstation state</td>
</tr>
<tr>
<td>380AB000</td>
<td>389567872</td>
<td>GQKWOD</td>
<td>Inquire workstation deferral and update states</td>
</tr>
<tr>
<td>380AC000</td>
<td>389568128</td>
<td>GQSEPLI</td>
<td>Inquire list element of polyline indexes</td>
</tr>
<tr>
<td>380AD000</td>
<td>389568384</td>
<td>GQPLR</td>
<td>Inquire polyline representation</td>
</tr>
<tr>
<td>380AE000</td>
<td>389568640</td>
<td>GQEPMI</td>
<td>Inquire list element of polymarker indexes</td>
</tr>
<tr>
<td>380AF000</td>
<td>389568896</td>
<td>GQPMR</td>
<td>Inquire polymarker representation</td>
</tr>
<tr>
<td>380B0000</td>
<td>389569152</td>
<td>GQETXI</td>
<td>Inquire list element of text indexes</td>
</tr>
<tr>
<td>380B1000</td>
<td>389569408</td>
<td>GQTXR</td>
<td>Inquire text representation</td>
</tr>
<tr>
<td>380B2000</td>
<td>389569664</td>
<td>GQTXXS</td>
<td>Inquire text extent</td>
</tr>
<tr>
<td>380B3000</td>
<td>389569920</td>
<td>GQFIAI</td>
<td>Inquire list element of fill area indexes</td>
</tr>
<tr>
<td>380B4000</td>
<td>389570176</td>
<td>GQFAR</td>
<td>Inquire fill area representation</td>
</tr>
<tr>
<td>380B5000</td>
<td>389570432</td>
<td>GQEPAI</td>
<td>Inquire list element of pattern indexes</td>
</tr>
<tr>
<td>380B6000</td>
<td>389570688</td>
<td>GQPAR</td>
<td>Inquire pattern representation</td>
</tr>
<tr>
<td>380B7000</td>
<td>389570944</td>
<td>GQECI</td>
<td>Inquire list element of color indexes</td>
</tr>
<tr>
<td>RCP Code (Hex.)</td>
<td>RCP Code (Decimal)</td>
<td>Call Name</td>
<td>Function</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>38008000</td>
<td>939571200</td>
<td>GQCR</td>
<td>Inquire color representation</td>
</tr>
<tr>
<td>38008090</td>
<td>939571456</td>
<td>GQKKT</td>
<td>Inquire workstation transformation</td>
</tr>
<tr>
<td>38008A00</td>
<td>939571712</td>
<td>GQSGWK</td>
<td>Inquire set member of segment names on workstation</td>
</tr>
<tr>
<td>38008B00</td>
<td>939571968</td>
<td>GQLCS</td>
<td>Inquire locator device state</td>
</tr>
<tr>
<td>38008C00</td>
<td>939572224</td>
<td>GQSKS</td>
<td>Inquire stroke device state</td>
</tr>
<tr>
<td>38008D00</td>
<td>939572480</td>
<td>GQVLS</td>
<td>Inquire valuator device state</td>
</tr>
<tr>
<td>38008E00</td>
<td>939572736</td>
<td>GQCHS</td>
<td>Inquire choice device state</td>
</tr>
<tr>
<td>38008F00</td>
<td>939572992</td>
<td>GQPKS</td>
<td>Inquire pick device state</td>
</tr>
<tr>
<td>3800C000</td>
<td>939573248</td>
<td>GQSTSS</td>
<td>Inquire string device state</td>
</tr>
<tr>
<td>3800C100</td>
<td>939573504</td>
<td>GQWKCA</td>
<td>Inquire workstation category</td>
</tr>
<tr>
<td>3800C200</td>
<td>939573760</td>
<td>GQWKCL</td>
<td>Inquire workstation classification</td>
</tr>
<tr>
<td>3800C300</td>
<td>939574016</td>
<td>GQDSP</td>
<td>Inquire display space size</td>
</tr>
<tr>
<td>3800C400</td>
<td>939574272</td>
<td>GQDWKA</td>
<td>Inquire dynamic modification of workstation attributes</td>
</tr>
<tr>
<td>3800C500</td>
<td>939574528</td>
<td>GQDDS</td>
<td>Inquire default deferral state values</td>
</tr>
<tr>
<td>3800C600</td>
<td>939574784</td>
<td>GQPFL</td>
<td>Inquire polyline facilities</td>
</tr>
<tr>
<td>3800C700</td>
<td>939575040</td>
<td>GQPPLR</td>
<td>Inquire predefined polyline representation</td>
</tr>
<tr>
<td>3800C800</td>
<td>939575296</td>
<td>GQPMF</td>
<td>Inquire polymarker facilities</td>
</tr>
<tr>
<td>3800C900</td>
<td>939575552</td>
<td>GQPPMR</td>
<td>Inquire predefined polymarker representation</td>
</tr>
<tr>
<td>3800CA00</td>
<td>939575808</td>
<td>GQTXF</td>
<td>Inquire text facilities</td>
</tr>
<tr>
<td>3800CB00</td>
<td>939576064</td>
<td>GQTXR</td>
<td>Inquire predefined text representation</td>
</tr>
<tr>
<td>3800CC00</td>
<td>939576320</td>
<td>GQFAF</td>
<td>Inquire fill area facilities</td>
</tr>
<tr>
<td>3800CD00</td>
<td>939576576</td>
<td>GQPFAR</td>
<td>Inquire predefined fill area representation</td>
</tr>
<tr>
<td>3800CE00</td>
<td>939576832</td>
<td>GQPFAR</td>
<td>Inquire pattern facilities</td>
</tr>
<tr>
<td>3800CF00</td>
<td>939577088</td>
<td>GQPPAR</td>
<td>Inquire predefined pattern representation</td>
</tr>
<tr>
<td>3800D000</td>
<td>939577344</td>
<td>GQCF</td>
<td>Inquire color facilities</td>
</tr>
<tr>
<td>3800D100</td>
<td>939577600</td>
<td>GQPCR</td>
<td>Inquire predefined color representation</td>
</tr>
<tr>
<td>3800D200</td>
<td>939577856</td>
<td>GQEGDP</td>
<td>Inquire list element of available generalized drawing primitives</td>
</tr>
<tr>
<td>3800D300</td>
<td>939578112</td>
<td>GQGDGP</td>
<td>Inquire generalized drawing primitive</td>
</tr>
<tr>
<td>3800D400</td>
<td>939578368</td>
<td>GQLKW</td>
<td>Inquire maximum length of workstation state tables</td>
</tr>
<tr>
<td>3800D500</td>
<td>939578624</td>
<td>GQSGP</td>
<td>Inquire number of segment priorities supported</td>
</tr>
<tr>
<td>3800D600</td>
<td>939578880</td>
<td>GQSDGA</td>
<td>Inquire dynamic modification of segment attributes</td>
</tr>
<tr>
<td>3800D700</td>
<td>939579136</td>
<td>GQLI</td>
<td>Inquire number of available logical input devices</td>
</tr>
<tr>
<td>3800D800</td>
<td>939579392</td>
<td>GQDLC</td>
<td>Inquire default locator device data</td>
</tr>
<tr>
<td>3800D900</td>
<td>939579648</td>
<td>GQDSK</td>
<td>Inquire default stroke device data</td>
</tr>
<tr>
<td>3800DA00</td>
<td>939579904</td>
<td>GQDVL</td>
<td>Inquire default valuator device data</td>
</tr>
<tr>
<td>3800DB00</td>
<td>939580160</td>
<td>GQDCH</td>
<td>Inquire default choice device data</td>
</tr>
<tr>
<td>3800DC00</td>
<td>939580416</td>
<td>GQDPK</td>
<td>Inquire default pick device data</td>
</tr>
<tr>
<td>3800DD00</td>
<td>939580672</td>
<td>GQDST</td>
<td>Inquire default string device data</td>
</tr>
<tr>
<td>3800DE00</td>
<td>939580928</td>
<td>GQASWK</td>
<td>Inquire set member of associated workstations</td>
</tr>
<tr>
<td>3800DF00</td>
<td>939581184</td>
<td>GQSGA</td>
<td>Inquire segment attributes</td>
</tr>
<tr>
<td>3800E000</td>
<td>939581440</td>
<td>GQPXAD</td>
<td>Inquire pixel array dimensions</td>
</tr>
<tr>
<td>3800E100</td>
<td>939581696</td>
<td>GQPXA</td>
<td>Inquire pixel array</td>
</tr>
<tr>
<td>3800E200</td>
<td>939581952</td>
<td>GQPX</td>
<td>Inquire pixel</td>
</tr>
<tr>
<td>3800E300</td>
<td>939582208</td>
<td>GQIQOV</td>
<td>Inquire input queue overflow</td>
</tr>
<tr>
<td>3800F000</td>
<td>939585536</td>
<td>GECKLS</td>
<td>Emergency close GKS</td>
</tr>
<tr>
<td>3800F100</td>
<td>939585792</td>
<td>GERLOG</td>
<td>Error logging</td>
</tr>
</tbody>
</table>
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

<table>
<thead>
<tr>
<th>Call Name</th>
<th>APL Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GACTM</td>
<td>1418</td>
<td>Accumulate transformation matrix</td>
</tr>
<tr>
<td>GACWK</td>
<td>1307</td>
<td>Activate workstation</td>
</tr>
<tr>
<td>GASGWK</td>
<td>1362</td>
<td>Associate segment with workstation</td>
</tr>
<tr>
<td>GCA</td>
<td>1316</td>
<td>Cell array</td>
</tr>
<tr>
<td>GCLKS</td>
<td>1301</td>
<td>Close GKS</td>
</tr>
<tr>
<td>GCLRWK</td>
<td>1309</td>
<td>Clear workstation</td>
</tr>
<tr>
<td>GCLSG</td>
<td>1358</td>
<td>Close segment</td>
</tr>
<tr>
<td>GCLWK</td>
<td>1303</td>
<td>Close workstation</td>
</tr>
<tr>
<td>GCRSG</td>
<td>1357</td>
<td>Create segment</td>
</tr>
<tr>
<td>GCSEGWK</td>
<td>1363</td>
<td>Copy segment to workstation</td>
</tr>
<tr>
<td>GDAWK</td>
<td>1308</td>
<td>Deactivate workstation</td>
</tr>
<tr>
<td>GDSG</td>
<td>1360</td>
<td>Delete segment</td>
</tr>
<tr>
<td>GDSGWK</td>
<td>1361</td>
<td>Delete segment from workstation</td>
</tr>
<tr>
<td>GECLS</td>
<td>1304</td>
<td>Emergency close GKS</td>
</tr>
<tr>
<td>GERLOG</td>
<td>1306</td>
<td>Error logging</td>
</tr>
<tr>
<td>GESC</td>
<td>1491</td>
<td>Escape</td>
</tr>
<tr>
<td>GEVTM</td>
<td>1416</td>
<td>Evaluate transformation matrix</td>
</tr>
<tr>
<td>GFA</td>
<td>1315</td>
<td>Fill area</td>
</tr>
<tr>
<td>GFLUSH</td>
<td>1392</td>
<td>Flush device events</td>
</tr>
<tr>
<td>GGD</td>
<td>1493</td>
<td>Generalized drawing primitive</td>
</tr>
<tr>
<td>GGTCH</td>
<td>1398</td>
<td>Get choice</td>
</tr>
<tr>
<td>GGTITM</td>
<td>1410</td>
<td>Get item type from GKSM</td>
</tr>
<tr>
<td>GGTLC</td>
<td>1393</td>
<td>Get locator</td>
</tr>
<tr>
<td>GGTPK</td>
<td>1399</td>
<td>Get pick</td>
</tr>
<tr>
<td>GGTSRK</td>
<td>1395</td>
<td>Get stroke</td>
</tr>
<tr>
<td>GGTSTS</td>
<td>1401</td>
<td>Get string</td>
</tr>
<tr>
<td>GGVLL</td>
<td>1396</td>
<td>Get valuator</td>
</tr>
<tr>
<td>GIITM</td>
<td>1412</td>
<td>Interpret item</td>
</tr>
<tr>
<td>GINCH</td>
<td>1496</td>
<td>Initialize choice</td>
</tr>
<tr>
<td>GINLC</td>
<td>1370</td>
<td>Initialize locator</td>
</tr>
<tr>
<td>GINPK</td>
<td>1497</td>
<td>Initialize pick</td>
</tr>
<tr>
<td>GINS</td>
<td>1364</td>
<td>Insert segment</td>
</tr>
<tr>
<td>GINSK</td>
<td>1494</td>
<td>Initialize stroke</td>
</tr>
<tr>
<td>GINST</td>
<td>1498</td>
<td>Initialize string</td>
</tr>
<tr>
<td>GINVL</td>
<td>1495</td>
<td>Initialize valuator</td>
</tr>
<tr>
<td>GMSGS</td>
<td>1490</td>
<td>Message</td>
</tr>
<tr>
<td>GOPKS</td>
<td>1300</td>
<td>Open GKS</td>
</tr>
<tr>
<td>GOPWK</td>
<td>1302</td>
<td>Open workstation</td>
</tr>
<tr>
<td>GPL</td>
<td>1313</td>
<td>Polyline</td>
</tr>
<tr>
<td>GPM</td>
<td>1314</td>
<td>Polymarker</td>
</tr>
<tr>
<td>GPRECS</td>
<td>1513</td>
<td>Pack data record</td>
</tr>
<tr>
<td>GQAACK</td>
<td>1422</td>
<td>Inquire set member of active workstations</td>
</tr>
<tr>
<td>GQASF</td>
<td>1460</td>
<td>Inquire aspect source flags</td>
</tr>
<tr>
<td>GQASWK</td>
<td>1429</td>
<td>Inquire set member of associated workstations</td>
</tr>
<tr>
<td>GQCF</td>
<td>1415</td>
<td>Inquire color facilities</td>
</tr>
<tr>
<td>Call Name</td>
<td>APL Code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>GQCHB</td>
<td>1439</td>
<td>Inquire character base vector</td>
</tr>
<tr>
<td>GQCHH</td>
<td>1431</td>
<td>Inquire character height</td>
</tr>
<tr>
<td>GQCHS</td>
<td>1487</td>
<td>Inquire choice device state</td>
</tr>
<tr>
<td>GQCHSP</td>
<td>1455</td>
<td>Inquire character spacing</td>
</tr>
<tr>
<td>GQCHUP</td>
<td>1433</td>
<td>Inquire character up vector</td>
</tr>
<tr>
<td>GQCHW</td>
<td>1438</td>
<td>Inquire character width</td>
</tr>
<tr>
<td>GQCHXP</td>
<td>1454</td>
<td>Inquire character expansion factor</td>
</tr>
<tr>
<td>GQCLIP</td>
<td>1464</td>
<td>Inquire clipping indicator</td>
</tr>
<tr>
<td>GQCNNTN</td>
<td>1461</td>
<td>Inquire current normalization transformation number</td>
</tr>
<tr>
<td>GQCR</td>
<td>1482</td>
<td>Inquire color representation</td>
</tr>
<tr>
<td>GQDCH</td>
<td>1503</td>
<td>Inquire default choice device data</td>
</tr>
<tr>
<td>GQDDS</td>
<td>1394</td>
<td>Inquire default deferral state values</td>
</tr>
<tr>
<td>GQDLC</td>
<td>1512</td>
<td>Inquire default locator device data</td>
</tr>
<tr>
<td>GQDPK</td>
<td>1505</td>
<td>Inquire default pick device data</td>
</tr>
<tr>
<td>GQDSGA</td>
<td>1425</td>
<td>Inquire dynamic modification of segment attributes</td>
</tr>
<tr>
<td>GQDSK</td>
<td>1501</td>
<td>Inquire default stroke device data</td>
</tr>
<tr>
<td>GQDSP</td>
<td>1504</td>
<td>Inquire default string device data</td>
</tr>
<tr>
<td>GQDV</td>
<td>1502</td>
<td>Inquire default valuator device data</td>
</tr>
<tr>
<td>GQDHKA</td>
<td>1390</td>
<td>Inquire dynamic modification of workstation attributes</td>
</tr>
<tr>
<td>GQECI</td>
<td>1481</td>
<td>Inquire list element of color indexes</td>
</tr>
<tr>
<td>GQEFAI</td>
<td>1477</td>
<td>Inquire list element of fill area indexes</td>
</tr>
<tr>
<td>GQEGDP</td>
<td>1420</td>
<td>Inquire list element of available generalized drawing primitives</td>
</tr>
<tr>
<td>GQENTN</td>
<td>1462</td>
<td>Inquire list element of normalization transformation numbers</td>
</tr>
<tr>
<td>GQEPAI</td>
<td>1479</td>
<td>Inquire list element of pattern indexes</td>
</tr>
<tr>
<td>GQERI</td>
<td>1470</td>
<td>Inquire list element of polyline indexes</td>
</tr>
<tr>
<td>GQFMI</td>
<td>1472</td>
<td>Inquire list element of polymarker indexes</td>
</tr>
<tr>
<td>GQFXI</td>
<td>1474</td>
<td>Inquire list element of text indexes</td>
</tr>
<tr>
<td>GQEWK</td>
<td>1508</td>
<td>Inquire list element of available workstation types</td>
</tr>
<tr>
<td>GFACI</td>
<td>1459</td>
<td>Inquire fill area color index</td>
</tr>
<tr>
<td>GFCAF</td>
<td>1408</td>
<td>Inquire fill area facilities</td>
</tr>
<tr>
<td>GFAR</td>
<td>1443</td>
<td>Inquire fill area index</td>
</tr>
<tr>
<td>GFARIS</td>
<td>1457</td>
<td>Inquire fill area interior style</td>
</tr>
<tr>
<td>GFAR</td>
<td>1478</td>
<td>Inquire fill area representation</td>
</tr>
<tr>
<td>GFASI</td>
<td>1458</td>
<td>Inquire fill area style index</td>
</tr>
<tr>
<td>GQGDP</td>
<td>1421</td>
<td>Inquire generalized drawing primitive</td>
</tr>
<tr>
<td>GQIQOV</td>
<td>1437</td>
<td>Inquire input queue overflow</td>
</tr>
<tr>
<td>GQLCS</td>
<td>1500</td>
<td>Inquire locator device state</td>
</tr>
<tr>
<td>GQLI</td>
<td>1426</td>
<td>Inquire number of available logical input devices</td>
</tr>
<tr>
<td>GQLN</td>
<td>1447</td>
<td>Inquire linetype</td>
</tr>
<tr>
<td>GQLVKS</td>
<td>1499</td>
<td>Inquire level of GKS</td>
</tr>
<tr>
<td>GQLWK</td>
<td>1423</td>
<td>Inquire maximum length of workstation state tables</td>
</tr>
<tr>
<td>GQLWCS</td>
<td>1448</td>
<td>Inquire linewidth scale factor</td>
</tr>
<tr>
<td>GQMK</td>
<td>1450</td>
<td>Inquire marker type</td>
</tr>
<tr>
<td>GQMKSC</td>
<td>1451</td>
<td>Inquire marker size scale factor</td>
</tr>
<tr>
<td>GQMNNTN</td>
<td>1506</td>
<td>Inquire maximum normalization transformation number</td>
</tr>
<tr>
<td>GQNT</td>
<td>1463</td>
<td>Inquire normalization transformation</td>
</tr>
<tr>
<td>GQOPS</td>
<td>1305</td>
<td>Inquire operating state value</td>
</tr>
<tr>
<td>GQPSG</td>
<td>1465</td>
<td>Inquire name of open segment</td>
</tr>
<tr>
<td>GQPWK</td>
<td>1419</td>
<td>Inquire set member of open workstations</td>
</tr>
<tr>
<td>GQP</td>
<td>1444</td>
<td>Inquire pattern size</td>
</tr>
<tr>
<td>GQPAF</td>
<td>1413</td>
<td>Inquire pattern facilities</td>
</tr>
<tr>
<td>GQPAL</td>
<td>1480</td>
<td>Inquire pattern representation</td>
</tr>
<tr>
<td>GQPURL</td>
<td>1445</td>
<td>Inquire pattern reference point</td>
</tr>
<tr>
<td>GQPCR</td>
<td>1417</td>
<td>Inquire predefined color representation</td>
</tr>
<tr>
<td>GQPFAR</td>
<td>1409</td>
<td>Inquire predefined fill area representation</td>
</tr>
<tr>
<td>Call Name</td>
<td>APL Code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>GQPKID</td>
<td>1446</td>
<td>Inquire pick identifier</td>
</tr>
<tr>
<td>GQPKS</td>
<td>1488</td>
<td>Inquire pick device state</td>
</tr>
<tr>
<td>GQPLCI</td>
<td>1449</td>
<td>Inquire polyline color index</td>
</tr>
<tr>
<td>GQPLF</td>
<td>1397</td>
<td>Inquire polyline facilities</td>
</tr>
<tr>
<td>GQPLI</td>
<td>1427</td>
<td>Inquire polyline index</td>
</tr>
<tr>
<td>GQPLR</td>
<td>1471</td>
<td>Inquire polyline representation</td>
</tr>
<tr>
<td>GQPMCI</td>
<td>1452</td>
<td>Inquire polymarker color index</td>
</tr>
<tr>
<td>GQPMF</td>
<td>1402</td>
<td>Inquire polymarker facilities</td>
</tr>
<tr>
<td>GQPMI</td>
<td>1428</td>
<td>Inquire polymarker index</td>
</tr>
<tr>
<td>GQPMR</td>
<td>1473</td>
<td>Inquire polymarker representation</td>
</tr>
<tr>
<td>GQPPAR</td>
<td>1414</td>
<td>Inquire predefined pattern representation</td>
</tr>
<tr>
<td>GQPPLR</td>
<td>1400</td>
<td>Inquire predefined polyline representation</td>
</tr>
<tr>
<td>GQPPMR</td>
<td>1404</td>
<td>Inquire predefined polymarker representation</td>
</tr>
<tr>
<td>GQPTXR</td>
<td>1406</td>
<td>Inquire predefined text representation</td>
</tr>
<tr>
<td>GQPX</td>
<td>1436</td>
<td>Inquire pixel</td>
</tr>
<tr>
<td>GQPA</td>
<td>1435</td>
<td>Inquire pixel array</td>
</tr>
<tr>
<td>GQPAAD</td>
<td>1434</td>
<td>Inquire pixel array dimensions</td>
</tr>
<tr>
<td>GQSGA</td>
<td>1432</td>
<td>Inquire segment attributes</td>
</tr>
<tr>
<td>GQSGP</td>
<td>1424</td>
<td>Inquire number of segment priorities supported</td>
</tr>
<tr>
<td>GQSGUS</td>
<td>1509</td>
<td>Inquire set member of segment names in use</td>
</tr>
<tr>
<td>GQSGWK</td>
<td>1484</td>
<td>Inquire set member of segment names on workstation</td>
</tr>
<tr>
<td>GQSIM</td>
<td>1466</td>
<td>Inquire more simultaneous events</td>
</tr>
<tr>
<td>GQSKS</td>
<td>1485</td>
<td>Inquire stroke device state</td>
</tr>
<tr>
<td>GQSTSS</td>
<td>1489</td>
<td>Inquire string device state</td>
</tr>
<tr>
<td>GQTXAL</td>
<td>1442</td>
<td>Inquire text alignment</td>
</tr>
<tr>
<td>GQTXCI</td>
<td>1456</td>
<td>Inquire text color index</td>
</tr>
<tr>
<td>GQTXF</td>
<td>1405</td>
<td>Inquire text facilities</td>
</tr>
<tr>
<td>GQTXFP</td>
<td>1453</td>
<td>Inquire text font and precision</td>
</tr>
<tr>
<td>GQTXI</td>
<td>1430</td>
<td>Inquire text index</td>
</tr>
<tr>
<td>GQTXP</td>
<td>1511</td>
<td>Inquire text path</td>
</tr>
<tr>
<td>GQTXR</td>
<td>1475</td>
<td>Inquire text representation</td>
</tr>
<tr>
<td>GQTXXS</td>
<td>1476</td>
<td>Inquire text extent</td>
</tr>
<tr>
<td>GQVLS</td>
<td>1486</td>
<td>Inquire valuator device state</td>
</tr>
<tr>
<td>GQWKC</td>
<td>1467</td>
<td>Inquire workstation connection and type</td>
</tr>
<tr>
<td>GQWKCA</td>
<td>1386</td>
<td>Inquire workstation category</td>
</tr>
<tr>
<td>GQWKCL</td>
<td>1387</td>
<td>Inquire workstation classification</td>
</tr>
<tr>
<td>GQWKDU</td>
<td>1469</td>
<td>Inquire workstation deferral and update states</td>
</tr>
<tr>
<td>GQWKMK</td>
<td>1507</td>
<td>Inquire workstation maximum numbers</td>
</tr>
<tr>
<td>GQWKS</td>
<td>1468</td>
<td>Inquire workstation state</td>
</tr>
<tr>
<td>GQWKT</td>
<td>1483</td>
<td>Inquire workstation transformation</td>
</tr>
<tr>
<td>GRDTIM</td>
<td>1411</td>
<td>Read item from GKSM</td>
</tr>
<tr>
<td>GRENSG</td>
<td>1359</td>
<td>Rename segment</td>
</tr>
<tr>
<td>GROCH</td>
<td>1380</td>
<td>Request choice</td>
</tr>
<tr>
<td>GROLC</td>
<td>1377</td>
<td>Request locator</td>
</tr>
<tr>
<td>GROPK</td>
<td>1381</td>
<td>Request pick</td>
</tr>
<tr>
<td>GROSK</td>
<td>1378</td>
<td>Request stroke</td>
</tr>
<tr>
<td>GROSTS</td>
<td>1382</td>
<td>Request string</td>
</tr>
<tr>
<td>GROVL</td>
<td>1379</td>
<td>Request valuator</td>
</tr>
<tr>
<td>GRSiWK</td>
<td>1310</td>
<td>Redraw all segments on workstation</td>
</tr>
<tr>
<td>GSA5F</td>
<td>1341</td>
<td>Set aspect source flags</td>
</tr>
<tr>
<td>GSCIH</td>
<td>1331</td>
<td>Set character height</td>
</tr>
<tr>
<td>GSCIM</td>
<td>1374</td>
<td>Set choice mode</td>
</tr>
<tr>
<td>GSCISP</td>
<td>1329</td>
<td>Set character spacing</td>
</tr>
<tr>
<td>GSCUHP</td>
<td>1332</td>
<td>Set character up vector</td>
</tr>
<tr>
<td>GSCXP</td>
<td>1328</td>
<td>Set character expansion factor</td>
</tr>
<tr>
<td>GSCLIP</td>
<td>1354</td>
<td>Set clipping indicator</td>
</tr>
<tr>
<td>Call Name</td>
<td>APL Code</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>GSCR</td>
<td>1349</td>
<td>Set color representation</td>
</tr>
<tr>
<td>GSDS</td>
<td>1312</td>
<td>Set deferral state</td>
</tr>
<tr>
<td>GSDTEC</td>
<td>1369</td>
<td>Set detectability</td>
</tr>
<tr>
<td>GSELNT</td>
<td>1353</td>
<td>Select normalization transformation</td>
</tr>
<tr>
<td>GSFACI</td>
<td>1338</td>
<td>Set fill area color index</td>
</tr>
<tr>
<td>GSFAI</td>
<td>1335</td>
<td>Set fill area index</td>
</tr>
<tr>
<td>GSFAINS</td>
<td>1336</td>
<td>Set fill area interior style</td>
</tr>
<tr>
<td>GSFAR</td>
<td>1347</td>
<td>Set fill area representation</td>
</tr>
<tr>
<td>GSFASI</td>
<td>1337</td>
<td>Set fill area style index</td>
</tr>
<tr>
<td>GSHLIT</td>
<td>1367</td>
<td>Set highlighting</td>
</tr>
<tr>
<td>GSLCM</td>
<td>1371</td>
<td>Set locator mode</td>
</tr>
<tr>
<td>GSLN</td>
<td>1318</td>
<td>Set linetype</td>
</tr>
<tr>
<td>GSLWSC</td>
<td>1319</td>
<td>Set linewidth scale factor</td>
</tr>
<tr>
<td>GSMCH</td>
<td>1385</td>
<td>Sample choice</td>
</tr>
<tr>
<td>GSMK</td>
<td>1322</td>
<td>Set marker type</td>
</tr>
<tr>
<td>GSMKSC</td>
<td>1323</td>
<td>Set marker size scale factor</td>
</tr>
<tr>
<td>GSMLC</td>
<td>1510</td>
<td>Sample locator</td>
</tr>
<tr>
<td>GSMPK</td>
<td>1403</td>
<td>Sample pick</td>
</tr>
<tr>
<td>GSMSK</td>
<td>1383</td>
<td>Sample stroke</td>
</tr>
<tr>
<td>GSMSTS</td>
<td>1389</td>
<td>Sample string</td>
</tr>
<tr>
<td>GSMVL</td>
<td>1384</td>
<td>Sample valuator</td>
</tr>
<tr>
<td>GSPA</td>
<td>1339</td>
<td>Set pattern size</td>
</tr>
<tr>
<td>GSPAR</td>
<td>1348</td>
<td>Set pattern representation</td>
</tr>
<tr>
<td>GSPARF</td>
<td>1340</td>
<td>Set pattern reference point</td>
</tr>
<tr>
<td>GSPKID</td>
<td>1342</td>
<td>Set pick identifier</td>
</tr>
<tr>
<td>GSPKM</td>
<td>1375</td>
<td>Set pick mode</td>
</tr>
<tr>
<td>GSPLCI</td>
<td>1320</td>
<td>Set polyline color index</td>
</tr>
<tr>
<td>GSPILI</td>
<td>1317</td>
<td>Set polyline index</td>
</tr>
<tr>
<td>GSPPLR</td>
<td>1344</td>
<td>Set polyline representation</td>
</tr>
<tr>
<td>GSPMCI</td>
<td>1324</td>
<td>Set polymarker color index</td>
</tr>
<tr>
<td>GSPMI</td>
<td>1321</td>
<td>Set polymarker index</td>
</tr>
<tr>
<td>GSPMR</td>
<td>1345</td>
<td>Set polymarker representation</td>
</tr>
<tr>
<td>GSGP</td>
<td>1368</td>
<td>Set segment priority</td>
</tr>
<tr>
<td>GSSGT</td>
<td>1365</td>
<td>Set segment transformation</td>
</tr>
<tr>
<td>GSSKM</td>
<td>1372</td>
<td>Set stroke mode</td>
</tr>
<tr>
<td>GSTM</td>
<td>1376</td>
<td>Set string mode</td>
</tr>
<tr>
<td>GSTXAL</td>
<td>1334</td>
<td>Set text alignment</td>
</tr>
<tr>
<td>GSTXCI</td>
<td>1330</td>
<td>Set text color index</td>
</tr>
<tr>
<td>GSTXFP</td>
<td>1326</td>
<td>Set text font and precision</td>
</tr>
<tr>
<td>GSTXI</td>
<td>1325</td>
<td>Set text index</td>
</tr>
<tr>
<td>GSTXP</td>
<td>1333</td>
<td>Set text path</td>
</tr>
<tr>
<td>GSTXR</td>
<td>1346</td>
<td>Set text representation</td>
</tr>
<tr>
<td>GSVIS</td>
<td>1366</td>
<td>Set visibility</td>
</tr>
<tr>
<td>GSVLM</td>
<td>1373</td>
<td>Set valuator mode</td>
</tr>
<tr>
<td>GSVP</td>
<td>1351</td>
<td>Set viewport</td>
</tr>
<tr>
<td>GSVPIP</td>
<td>1352</td>
<td>Set viewport input priority</td>
</tr>
<tr>
<td>GSWKVP</td>
<td>1356</td>
<td>Set workstation viewport</td>
</tr>
<tr>
<td>GSWKWN</td>
<td>1355</td>
<td>Set workstation window</td>
</tr>
<tr>
<td>GSWN</td>
<td>1350</td>
<td>Set window</td>
</tr>
<tr>
<td>GTXS</td>
<td>1492</td>
<td>Text</td>
</tr>
<tr>
<td>GURECS</td>
<td>1514</td>
<td>Unpack data record</td>
</tr>
<tr>
<td>GUWK</td>
<td>1311</td>
<td>Update workstation</td>
</tr>
<tr>
<td>GWAIT</td>
<td>1391</td>
<td>Await event</td>
</tr>
<tr>
<td>GWITM</td>
<td>1407</td>
<td>Write item to GKSM</td>
</tr>
</tbody>
</table>
## APL codes ordered by code value

<table>
<thead>
<tr>
<th>APL Code</th>
<th>Call Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1300</td>
<td>GOPKS</td>
<td>Open GKS</td>
</tr>
<tr>
<td>1301</td>
<td>GCLKS</td>
<td>Close GKS</td>
</tr>
<tr>
<td>1302</td>
<td>GOPWK</td>
<td>Open workstation</td>
</tr>
<tr>
<td>1303</td>
<td>GCLWK</td>
<td>Close workstation</td>
</tr>
<tr>
<td>1304</td>
<td>GECLKS</td>
<td>Emergency close GKS</td>
</tr>
<tr>
<td>1305</td>
<td>GQOPS</td>
<td>Inquire operating state value</td>
</tr>
<tr>
<td>1306</td>
<td>GERLOG</td>
<td>Error logging</td>
</tr>
<tr>
<td>1307</td>
<td>GACWK</td>
<td>Activate workstation</td>
</tr>
<tr>
<td>1308</td>
<td>GDAWK</td>
<td>Deactivate workstation</td>
</tr>
<tr>
<td>1309</td>
<td>GCLRWK</td>
<td>Clear workstation</td>
</tr>
<tr>
<td>1310</td>
<td>GRSGWK</td>
<td>Redraw all segments on workstation</td>
</tr>
<tr>
<td>1311</td>
<td>GUWK</td>
<td>Update workstation</td>
</tr>
<tr>
<td>1312</td>
<td>GSWS</td>
<td>Set deferral state</td>
</tr>
<tr>
<td>1313</td>
<td>GPL</td>
<td>Polyline</td>
</tr>
<tr>
<td>1314</td>
<td>GPM</td>
<td>Polymarker</td>
</tr>
<tr>
<td>1315</td>
<td>GFA</td>
<td>Fill area</td>
</tr>
<tr>
<td>1316</td>
<td>GCA</td>
<td>Cell array</td>
</tr>
<tr>
<td>1317</td>
<td>GSPLI</td>
<td>Set polyline index</td>
</tr>
<tr>
<td>1318</td>
<td>GSLN</td>
<td>Set linetype</td>
</tr>
<tr>
<td>1319</td>
<td>GSLWSC</td>
<td>Set linewidth scale factor</td>
</tr>
<tr>
<td>1320</td>
<td>GSPLCI</td>
<td>Set polyline color index</td>
</tr>
<tr>
<td>1321</td>
<td>GSPMI</td>
<td>Set polymarker index</td>
</tr>
<tr>
<td>1322</td>
<td>GSMK</td>
<td>Set marker type</td>
</tr>
<tr>
<td>1323</td>
<td>GSMKSC</td>
<td>Set marker size scale factor</td>
</tr>
<tr>
<td>1324</td>
<td>GSPMCI</td>
<td>Set polymarker color index</td>
</tr>
<tr>
<td>1325</td>
<td>GSTXI</td>
<td>Set text index</td>
</tr>
<tr>
<td>1326</td>
<td>GSTXFP</td>
<td>Set text font and precision</td>
</tr>
<tr>
<td>1328</td>
<td>GSCHXP</td>
<td>Set character expansion factor</td>
</tr>
<tr>
<td>1329</td>
<td>GCHSP</td>
<td>Set character spacing</td>
</tr>
<tr>
<td>1330</td>
<td>GSTXCI</td>
<td>Set text color index</td>
</tr>
<tr>
<td>1331</td>
<td>GCHH</td>
<td>Set character height</td>
</tr>
<tr>
<td>1332</td>
<td>GCHUP</td>
<td>Set character up vector</td>
</tr>
<tr>
<td>1333</td>
<td>GSTXP</td>
<td>Set text path</td>
</tr>
<tr>
<td>1334</td>
<td>GSTXAL</td>
<td>Set text alignment</td>
</tr>
<tr>
<td>1335</td>
<td>GSFAI</td>
<td>Set fill area index</td>
</tr>
<tr>
<td>1336</td>
<td>GSFAIS</td>
<td>Set fill area interior style</td>
</tr>
<tr>
<td>1337</td>
<td>GSFASI</td>
<td>Set fill area style index</td>
</tr>
<tr>
<td>1338</td>
<td>GSFAICI</td>
<td>Set fill area color index</td>
</tr>
<tr>
<td>1339</td>
<td>GSPA</td>
<td>Set pattern size</td>
</tr>
<tr>
<td>1340</td>
<td>GSPARF</td>
<td>Set pattern reference point</td>
</tr>
<tr>
<td>1341</td>
<td>GSASF</td>
<td>Set aspect source flags</td>
</tr>
<tr>
<td>1342</td>
<td>GSPKID</td>
<td>Set pick identifier</td>
</tr>
<tr>
<td>1344</td>
<td>GSPLR</td>
<td>Set polyline representation</td>
</tr>
<tr>
<td>1345</td>
<td>GSPHR</td>
<td>Set polymarker representation</td>
</tr>
<tr>
<td>1346</td>
<td>GSTXR</td>
<td>Set text representation</td>
</tr>
<tr>
<td>1347</td>
<td>GSFAI</td>
<td>Set fill area representation</td>
</tr>
<tr>
<td>1348</td>
<td>GSPAR</td>
<td>Set pattern representation</td>
</tr>
<tr>
<td>1349</td>
<td>GSCR</td>
<td>Set color representation</td>
</tr>
<tr>
<td>1350</td>
<td>GSWH</td>
<td>Set window</td>
</tr>
<tr>
<td>1351</td>
<td>GSVP</td>
<td>Set viewport</td>
</tr>
<tr>
<td>1352</td>
<td>GSVPUP</td>
<td>Set viewport input priority</td>
</tr>
<tr>
<td>1353</td>
<td>GSELNT</td>
<td>Select normalization transformation</td>
</tr>
<tr>
<td>1354</td>
<td>GSCLIP</td>
<td>Set clipping indicator</td>
</tr>
<tr>
<td>APL Code</td>
<td>Call Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>1355</td>
<td>GSWKWN</td>
<td>Set workstation window</td>
</tr>
<tr>
<td>1356</td>
<td>GSWKVP</td>
<td>Set workstation viewport</td>
</tr>
<tr>
<td>1357</td>
<td>GCRSG</td>
<td>Create segment</td>
</tr>
<tr>
<td>1358</td>
<td>GCLSG</td>
<td>Close segment</td>
</tr>
<tr>
<td>1359</td>
<td>GREN SG</td>
<td>Rename segment</td>
</tr>
<tr>
<td>1360</td>
<td>GDSG</td>
<td>Delete segment</td>
</tr>
<tr>
<td>1361</td>
<td>GDSGWK</td>
<td>Delete segment from workstation</td>
</tr>
<tr>
<td>1362</td>
<td>GAGSWK</td>
<td>Associate segment with workstation</td>
</tr>
<tr>
<td>1363</td>
<td>GCAGWK</td>
<td>Copy segment to workstation</td>
</tr>
<tr>
<td>1364</td>
<td>GINS G</td>
<td>Insert segment</td>
</tr>
<tr>
<td>1365</td>
<td>GSSTG</td>
<td>Set segment transformation</td>
</tr>
<tr>
<td>1366</td>
<td>GSVIS</td>
<td>Set visibility</td>
</tr>
<tr>
<td>1367</td>
<td>GSH LIT</td>
<td>Set highlighting</td>
</tr>
<tr>
<td>1368</td>
<td>GSSTP</td>
<td>Set segment priority</td>
</tr>
<tr>
<td>1369</td>
<td>GSDTEC</td>
<td>Set detectability</td>
</tr>
<tr>
<td>1370</td>
<td>GINL C</td>
<td>Initialize locator</td>
</tr>
<tr>
<td>1371</td>
<td>GSLCM</td>
<td>Set locator mode</td>
</tr>
<tr>
<td>1372</td>
<td>GSSCM</td>
<td>Set stroke mode</td>
</tr>
<tr>
<td>1373</td>
<td>GSVLM</td>
<td>Set valuator mode</td>
</tr>
<tr>
<td>1374</td>
<td>GSC HM</td>
<td>Set choice mode</td>
</tr>
<tr>
<td>1375</td>
<td>GSPKM</td>
<td>Set pick mode</td>
</tr>
<tr>
<td>1376</td>
<td>GSS TM</td>
<td>Set string mode</td>
</tr>
<tr>
<td>1377</td>
<td>GRO LC</td>
<td>Request locator</td>
</tr>
<tr>
<td>1378</td>
<td>GRO SK</td>
<td>Request stroke</td>
</tr>
<tr>
<td>1379</td>
<td>GRO VL</td>
<td>Request valuator</td>
</tr>
<tr>
<td>1380</td>
<td>GRO CH</td>
<td>Request choice</td>
</tr>
<tr>
<td>1381</td>
<td>GROP K</td>
<td>Request pick</td>
</tr>
<tr>
<td>1382</td>
<td>GRO STS</td>
<td>Request string</td>
</tr>
<tr>
<td>1383</td>
<td>GSMS K</td>
<td>Sample stroke</td>
</tr>
<tr>
<td>1384</td>
<td>G SVM L</td>
<td>Sample valuator</td>
</tr>
<tr>
<td>1385</td>
<td>G SM CH</td>
<td>Sample choice</td>
</tr>
<tr>
<td>1386</td>
<td>GQWK C A</td>
<td>Inquire workstation category</td>
</tr>
<tr>
<td>1387</td>
<td>GQWK C L</td>
<td>Inquire workstation classification</td>
</tr>
<tr>
<td>1388</td>
<td>GQO SP</td>
<td>Inquire display space size</td>
</tr>
<tr>
<td>1389</td>
<td>GSM ST S</td>
<td>Sample string</td>
</tr>
<tr>
<td>1390</td>
<td>GQDKWA</td>
<td>Inquire dynamic modification of workstation attributes</td>
</tr>
<tr>
<td>1391</td>
<td>GN AIT</td>
<td>Await event</td>
</tr>
<tr>
<td>1392</td>
<td>GFLUSH</td>
<td>Flush device events</td>
</tr>
<tr>
<td>1393</td>
<td>GG TL C</td>
<td>Get locator</td>
</tr>
<tr>
<td>1394</td>
<td>GQ D D S</td>
<td>Inquire default deferral state values</td>
</tr>
<tr>
<td>1395</td>
<td>GGSK</td>
<td>Get stroke</td>
</tr>
<tr>
<td>1396</td>
<td>GGT VL</td>
<td>Get valuator</td>
</tr>
<tr>
<td>1397</td>
<td>GQPLF</td>
<td>Inquire polyline facilities</td>
</tr>
<tr>
<td>1398</td>
<td>GGT CH</td>
<td>Get choice</td>
</tr>
<tr>
<td>1399</td>
<td>GGT PK</td>
<td>Get pick</td>
</tr>
<tr>
<td>1400</td>
<td>GQPL R</td>
<td>Inquire predefined polyline representation</td>
</tr>
<tr>
<td>1401</td>
<td>GGT ST S</td>
<td>Get string</td>
</tr>
<tr>
<td>1402</td>
<td>GQPM F</td>
<td>Inquire polymarker facilities</td>
</tr>
<tr>
<td>1403</td>
<td>GSM PK</td>
<td>Sample pick</td>
</tr>
<tr>
<td>1404</td>
<td>GQPM MR</td>
<td>Inquire predefined polymarker representation</td>
</tr>
<tr>
<td>1405</td>
<td>GQTX F</td>
<td>Inquire text facilities</td>
</tr>
<tr>
<td>1406</td>
<td>GQTX XR</td>
<td>Inquire predefined text representation</td>
</tr>
<tr>
<td>1407</td>
<td>GW IT M</td>
<td>Write item to GKSM</td>
</tr>
<tr>
<td>1408</td>
<td>GQ FA F</td>
<td>Inquire fill area facilities</td>
</tr>
<tr>
<td>1409</td>
<td>GQPF AR</td>
<td>Inquire predefined fill area representation</td>
</tr>
<tr>
<td>1410</td>
<td>GGT IT M</td>
<td>Get item type from GKSM</td>
</tr>
<tr>
<td>APL Code</td>
<td>Call Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>1411</td>
<td>GRDITM</td>
<td>Read item from GKSM</td>
</tr>
<tr>
<td>1412</td>
<td>GIITM</td>
<td>Interpret item</td>
</tr>
<tr>
<td>1413</td>
<td>GQPAF</td>
<td>Inquire pattern facilities</td>
</tr>
<tr>
<td>1414</td>
<td>GQPPAR</td>
<td>Inquire predefined pattern representation</td>
</tr>
<tr>
<td>1415</td>
<td>GQCF</td>
<td>Inquire color facilities</td>
</tr>
<tr>
<td>1416</td>
<td>GEVTM</td>
<td>Evaluate transformation matrix</td>
</tr>
<tr>
<td>1417</td>
<td>GQPCR</td>
<td>Inquire predefined color representation</td>
</tr>
<tr>
<td>1418</td>
<td>GACTM</td>
<td>Accumulate transformation matrix</td>
</tr>
<tr>
<td>1419</td>
<td>GQQPKW</td>
<td>Inquire set member of open workstations</td>
</tr>
<tr>
<td>1420</td>
<td>GQEDGP</td>
<td>Inquire list element of available generalized drawing primitives</td>
</tr>
<tr>
<td>1421</td>
<td>GQGDGP</td>
<td>Inquire generalized drawing primitive</td>
</tr>
<tr>
<td>1422</td>
<td>GQACKW</td>
<td>Inquire set member of active workstations</td>
</tr>
<tr>
<td>1423</td>
<td>GQLWK</td>
<td>Inquire maximum length of workstation state tables</td>
</tr>
<tr>
<td>1424</td>
<td>GQSGP</td>
<td>Inquire number of segment priorities supported</td>
</tr>
<tr>
<td>1425</td>
<td>GQSGSA</td>
<td>Inquire dynamic modification of segment attributes</td>
</tr>
<tr>
<td>1426</td>
<td>GQLI</td>
<td>Inquire number of available logical input devices</td>
</tr>
<tr>
<td>1427</td>
<td>GQPLI</td>
<td>Inquire polyline index</td>
</tr>
<tr>
<td>1428</td>
<td>GQPMI</td>
<td>Inquire polymarker index</td>
</tr>
<tr>
<td>1429</td>
<td>GQASWK</td>
<td>Inquire set member of associated workstations</td>
</tr>
<tr>
<td>1430</td>
<td>GQTXI</td>
<td>Inquire text index</td>
</tr>
<tr>
<td>1431</td>
<td>GQCHH</td>
<td>Inquire character height</td>
</tr>
<tr>
<td>1432</td>
<td>GQSGA</td>
<td>Inquire segment attributes</td>
</tr>
<tr>
<td>1433</td>
<td>GQCHUP</td>
<td>Inquire character up vector</td>
</tr>
<tr>
<td>1434</td>
<td>GQPXAD</td>
<td>Inquire pixel array dimensions</td>
</tr>
<tr>
<td>1435</td>
<td>GQPXA</td>
<td>Inquire pixel array</td>
</tr>
<tr>
<td>1436</td>
<td>GQPX</td>
<td>Inquire pixel</td>
</tr>
<tr>
<td>1437</td>
<td>GQIQOV</td>
<td>Inquire input queue overflow</td>
</tr>
<tr>
<td>1438</td>
<td>GQCHW</td>
<td>Inquire character width</td>
</tr>
<tr>
<td>1439</td>
<td>GQCHB</td>
<td>Inquire character base vector</td>
</tr>
<tr>
<td>1442</td>
<td>GQTXAL</td>
<td>Inquire text alignment</td>
</tr>
<tr>
<td>1443</td>
<td>GQFAI</td>
<td>Inquire fill area index</td>
</tr>
<tr>
<td>1444</td>
<td>GQPA</td>
<td>Inquire pattern size</td>
</tr>
<tr>
<td>1445</td>
<td>GQPARF</td>
<td>Inquire pattern reference point</td>
</tr>
<tr>
<td>1446</td>
<td>GPKID</td>
<td>Inquire pick identifier</td>
</tr>
<tr>
<td>1447</td>
<td>GQLN</td>
<td>Inquire linetype</td>
</tr>
<tr>
<td>1448</td>
<td>GQLWSC</td>
<td>Inquire linewidth scale factor</td>
</tr>
<tr>
<td>1449</td>
<td>GQPLCI</td>
<td>Inquire polyline color index</td>
</tr>
<tr>
<td>1450</td>
<td>GQMK</td>
<td>Inquire marker type</td>
</tr>
<tr>
<td>1451</td>
<td>GQMKSC</td>
<td>Inquire marker size scale factor</td>
</tr>
<tr>
<td>1452</td>
<td>GQPMCI</td>
<td>Inquire polymarker color index</td>
</tr>
<tr>
<td>1453</td>
<td>GQTXFP</td>
<td>Inquire text font and precision</td>
</tr>
<tr>
<td>1454</td>
<td>GCHXPM</td>
<td>Inquire character expansion factor</td>
</tr>
<tr>
<td>1455</td>
<td>GCHSP</td>
<td>Inquire character spacing</td>
</tr>
<tr>
<td>1456</td>
<td>GQTXCI</td>
<td>Inquire text color index</td>
</tr>
<tr>
<td>1457</td>
<td>GQFAIS</td>
<td>Inquire fill area interior style</td>
</tr>
<tr>
<td>1458</td>
<td>GQFASI</td>
<td>Inquire fill area style index</td>
</tr>
<tr>
<td>1459</td>
<td>GQFACI</td>
<td>Inquire fill area color index</td>
</tr>
<tr>
<td>1460</td>
<td>GQASF</td>
<td>Inquire aspect source flags</td>
</tr>
<tr>
<td>1461</td>
<td>GQCNTN</td>
<td>Inquire current normalization transformation number</td>
</tr>
<tr>
<td>1462</td>
<td>GQENTN</td>
<td>Inquire list element of normalization transformation numbers</td>
</tr>
<tr>
<td>1463</td>
<td>GQNT</td>
<td>Inquire normalization transformation</td>
</tr>
<tr>
<td>1464</td>
<td>GQCLIP</td>
<td>Inquire clipping indicator</td>
</tr>
<tr>
<td>1465</td>
<td>GQPSG</td>
<td>Inquire name of open segment</td>
</tr>
<tr>
<td>1466</td>
<td>GQSIM</td>
<td>Inquire more simultaneous events</td>
</tr>
<tr>
<td>1467</td>
<td>GQWKC</td>
<td>Inquire workstation connection and type</td>
</tr>
<tr>
<td>1468</td>
<td>GQWKS</td>
<td>Inquire workstation state</td>
</tr>
</tbody>
</table>
### APL codes

<table>
<thead>
<tr>
<th>APL Code</th>
<th>Call Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1469</td>
<td>GQWKDU</td>
<td>Inquire workstation deferral and update states</td>
</tr>
<tr>
<td>1470</td>
<td>GQEP LI</td>
<td>Inquire list element of polyline indexes</td>
</tr>
<tr>
<td>1471</td>
<td>GQPLR</td>
<td>Inquire polyline representation</td>
</tr>
<tr>
<td>1472</td>
<td>GQREM I</td>
<td>Inquire list element of polymarker indexes</td>
</tr>
<tr>
<td>1473</td>
<td>GQPMR</td>
<td>Inquire polymarker representation</td>
</tr>
<tr>
<td>1474</td>
<td>GQET XI</td>
<td>Inquire list element of text indexes</td>
</tr>
<tr>
<td>1475</td>
<td>GQTXR</td>
<td>Inquire text representation</td>
</tr>
<tr>
<td>1476</td>
<td>GQTXXS</td>
<td>Inquire text extent</td>
</tr>
<tr>
<td>1477</td>
<td>GQF AI</td>
<td>Inquire list element of fill area indexes</td>
</tr>
<tr>
<td>1478</td>
<td>GQFAR</td>
<td>Inquire fill area representation</td>
</tr>
<tr>
<td>1479</td>
<td>GQEP AI</td>
<td>Inquire list element of pattern indexes</td>
</tr>
<tr>
<td>1480</td>
<td>GQPAR</td>
<td>Inquire pattern representation</td>
</tr>
<tr>
<td>1481</td>
<td>GQECI</td>
<td>Inquire list element of color indexes</td>
</tr>
<tr>
<td>1482</td>
<td>GQCR</td>
<td>Inquire color representation</td>
</tr>
<tr>
<td>1483</td>
<td>GQWT</td>
<td>Inquire workstation transformation</td>
</tr>
<tr>
<td>1484</td>
<td>GQS GWK</td>
<td>Inquire set member of segment names on workstation</td>
</tr>
<tr>
<td>1485</td>
<td>GQSKS</td>
<td>Inquire stroke device state</td>
</tr>
<tr>
<td>1486</td>
<td>GQVLS</td>
<td>Inquire valuator device state</td>
</tr>
<tr>
<td>1487</td>
<td>GQC HS</td>
<td>Inquire choice device state</td>
</tr>
<tr>
<td>1488</td>
<td>GQP KS</td>
<td>Inquire pick device state</td>
</tr>
<tr>
<td>1489</td>
<td>GQSTSS</td>
<td>Inquire string device state</td>
</tr>
<tr>
<td>1490</td>
<td>GMSGS</td>
<td>Message</td>
</tr>
<tr>
<td>1491</td>
<td>GESC</td>
<td>Escape</td>
</tr>
<tr>
<td>1492</td>
<td>GTXS</td>
<td>Text</td>
</tr>
<tr>
<td>1493</td>
<td>GGD P</td>
<td>Generalized drawing primitive</td>
</tr>
<tr>
<td>1494</td>
<td>GINS K</td>
<td>Initialize stroke</td>
</tr>
<tr>
<td>1495</td>
<td>GIN VL</td>
<td>Initialize valuator</td>
</tr>
<tr>
<td>1496</td>
<td>GINCH</td>
<td>Initialize choice</td>
</tr>
<tr>
<td>1497</td>
<td>GIN PK</td>
<td>Initialize pick</td>
</tr>
<tr>
<td>1498</td>
<td>GINST S</td>
<td>Initialize string</td>
</tr>
<tr>
<td>1499</td>
<td>GQLVK S</td>
<td>Inquire level of GKS</td>
</tr>
<tr>
<td>1500</td>
<td>GQ LCS</td>
<td>Inquire locator device state</td>
</tr>
<tr>
<td>1501</td>
<td>GQDS K</td>
<td>Inquire default stroke device data</td>
</tr>
<tr>
<td>1502</td>
<td>GQDV L</td>
<td>Inquire default valuator device data</td>
</tr>
<tr>
<td>1503</td>
<td>GQDCH</td>
<td>Inquire default choice device data</td>
</tr>
<tr>
<td>1504</td>
<td>GQDST</td>
<td>Inquire default string device data</td>
</tr>
<tr>
<td>1505</td>
<td>GQDP K</td>
<td>Inquire default pick device data</td>
</tr>
<tr>
<td>1506</td>
<td>GQMN T N</td>
<td>Inquire maximum normalization transformation number</td>
</tr>
<tr>
<td>1507</td>
<td>GQWM K</td>
<td>Inquire workstation maximum numbers</td>
</tr>
<tr>
<td>1508</td>
<td>GQEW K</td>
<td>Inquire list element of available workstation types</td>
</tr>
<tr>
<td>1509</td>
<td>GQSG US</td>
<td>Inquire set member of segment names in use</td>
</tr>
<tr>
<td>1510</td>
<td>GSM LC</td>
<td>Sample locator</td>
</tr>
<tr>
<td>1511</td>
<td>GQTX P</td>
<td>Inquire text path</td>
</tr>
<tr>
<td>1512</td>
<td>GQDL C</td>
<td>Inquire default locator device data</td>
</tr>
<tr>
<td>1513</td>
<td>GPRECS</td>
<td>Pack data record</td>
</tr>
<tr>
<td>1514</td>
<td>GURECS</td>
<td>Unpack data record</td>
</tr>
</tbody>
</table>
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 (alphanumeric 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.
**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.

**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.

**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.

**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.

**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
3179-G 52
3270-PC/G and /GX ranges 52
3278 52
3279 52
5080 52

A
AAB (application anchor block) 44
Accumulate transformation matrix (GACTM) 78
Activate workstation (GACWK) 80
ADMASP (SPI interface entry point) 45
ADMJROOM 1
See also ROOM program
alignment, text 40
anchor pointer 44
APL
codes 63
ordered by code value 431
ordered by function name 427
language considerations for calls 46
using with GDDM 48
application anchor block (AAB) 44
application-defined error handling 58
argument conventions 27
aspect 9
source flag 10
assembler language
language considerations for calls 46
linkage conventions 47
Associate segment with workstation (GASGWK) 81
attribute functions 9
Set aspect source flags (GSASF) 275
Set character expansion factor (GSCHXP) 280
Set character height (GSCHH) 276
Set character spacing (GSCHSP) 278
Set character up vector (GSCHUP) 279
Set color representation (GSCR) 282
Set fill area color index (GSFACI) 288
Set fill area index (GSFAI) 289
Set fill area interior style (GSFAIS) 290
Set fill area representation (GSFAR) 291
Set fill area style index (GSFASI) 293
Set linetype (GSLN) 297
Setlinewidth scale factor (GSLWSC) 298
Set marker size scale factor (GSMKSC) 302
Set marker type (GSMK) 301
Set pattern reference point (GSPARF) 314
Set pattern representation (GSPAR) 312
Set pattern size (GSPA) 311
attribute functions (continued)
Set pick identifier (GSPKID) 315
Set polyline color index (GSPLCI) 317
Set polyline index (GSPLI) 318
Set polyline representation (GSPLR) 319
Set polymarker color index (GSPMCI) 321
Set polymarker index (GSPMI) 322
Set polymarker representation (GSPMR) 323
Set text alignment (GSTXAL) 331
Set text color index (GSTXCI) 333
Set text font and precision (GSTXPath) 334
Set text index (GSTXI) 337
Set text path (GSTXP) 338
Set text representation (GSTXR) 339
attributes 9
Await event (GWAIT) 356

B
BASIC (IBM)
interface to GDDM 48
language considerations for calls 46
bundles 10

call statements, GDDM-GKS, syntax conventions 63
cell array 16
Cell array (GCA) 83
clear workstation (GCLRWK) 85
Close GKS (GCLKS) 85
Close segment (GCLSG) 87
Close workstation (GCLWK) 88

COBOL
format of calls 47
language considerations for calls 46
codes
APL 63, 427
RCP 63, 419
concepts of GKS 5
control functions 9
Activate workstation (GACWK) 80
Clear workstation (GCLRWK) 85
Close GKS (GCLKS) 85
Close segment (GCLSG) 87
Close workstation (GCLWK) 88
Deactivate workstation (GDAWK) 91
Escape (GESC) 97
Message (GMSGS) 134

© Copyright IBM Corp. 1984, 1987
439
control functions (continued)
Message (VS FORTRAN only) (GMSG) 133
Open GKS (GOPKS) 135
Open workstation (GOPWK) 136
Redraw all segments on workstation (GRSGWK) 274
Set deferral state (GSDS) 284
Update workstation (GUWK) 355
conventions, syntax for calls 63
cordinate systems 6
Copy segment to workstation (GCSGWK) 90
Create segment (GCRSG) 89

D
data structures, GKS 359
data types for function parameters 64
Deactivate workstation (GDAWK) 91
Delete segment (GDSG) 92
Delete segment from workstation (GDSGWK) 93
description table, GKS 8, 360
detectability 21
device coordinates 6
device tokens 51
devices, physical 50
differences between GDDM-GKS and the GKS standards 39

E
echo 25
Emergency close GKS (GECLKS) 94
don GKS
See Close GKS (GCLKS)
enumeration type parameters 65
enumeration types 379
error file information 59
error handling 26, 56
application-defined 58
error file information 59
error messages 59
standard 57
when mixing with GDDM Base 50
Error handling (GERHND) 95
Error logging (GERLOG) 96
error messages 59, 66
error state list, GKS 377
error-handling functions
Emergency close GKS (GECLKS) 94
Error handling (GERHND) 95
Error logging (GERLOG) 96
ing errors, GKS 66
Escape (GESC) 97
Evaluate transformation matrix (GEVTM) 98
example programs 405

G
GACTM (Accumulate transformation matrix) 78
GACWK (Activate workstation) 80
GASGWK (Associate segment with workstation) 81
GCA (Cell array) 83
GCLKS (Close GKS) 85
GCLRWK (Clear workstation) 85
GCLSG (Close segment) 87
GCLWK (Close workstation) 88
GCRSG (Create segment) 89
GCSGWK (Copy segment to workstation) 90
GDAWK (Deactivate workstation) 91
GDDM Base, relation of GDDM-GKS to 39
GDDM hierarchy 49, 56
GDDM Internet home page xi
GDDM-GKS 60
functions 63
restrictions 66
under various subsystems 60
GDF files 54
GDPs 16
GDSG (Delete segment) 92
GDSGWK (Delete segment from workstation) 93
GECLKS (Emergency close GKS) 94
Generalized drawing primitive (GGDP) 103
generalized drawing primitives 16
GERHND (Error handling) 95
GERLOG (Error logging) 96
GESC (Escape) 97
Get choice (GGTCH) 105
Get item type from GKSM (GGTITM) 106
Get locator (GGTLC) 107
Get pick (GGTPK) 108
Get string (FORTRAN only) (GGTST) 110
Get string (GGTSTS) 111
Get stroke (GGTSK) 109
Get valuator (GGTVL) 112
GEVTM (Evaluate transformation matrix) 98
GFA (Fill area) 100
GFLUSH (Flush device events) 102
GGDP (Generalized drawing primitive) 103
GGTCH (Get choice) 105
GGTITM (Get item type from GKSM) 106
GGTLC (Get locator) 107
GGTPK (Get pick) 108
GGTSK (Get stroke) 109
GGTST (Get string (FORTRAN only)) 110
GGTSTS (Get string) 111
GGTVL (Get valuator) 112
GIITM (Interpret item) 113
GINCH (Initialize choice) 114
GINLC (Initialize locator) 116
GINPK (Initialize pick) 120
GINSG (Insert segment) 122
GINSK (Initialize stroke) 123
GINST (Initialize string (FORTRAN only)) 127
GINSTS (Initialize string) 129
GINVL (Initialize valuator) 132
GKS
data structures 8, 359
description table 360
enumeration types 379
error state list 377
function numbers 67
functions 8, 78
and other GDDM functions, mixing 49
metatypes 54
structure 383
operating states 7, 360
segment state list 377
standard
differences from GDDM-GKS 39
relation of GDDM-GKS to 39
state lists 8, 361
workstation
description tables 368
state list 364
glossary 435
GMSG (Message (VS FORTRAN only)) 133
GMSGS (Message) 134
GOPKS (Open GKS) 135
GOPWK (Open workstation) 136
GPL (Polyline) 138
GPM (Polymarker) 139
GPREC (Pack data record (FORTRAN only)) 140
GPRECS (Pack data record) 141
GQACWK (Inquire set member of active workstations) 143
GQASF (Inquire aspect source flags) 144
GQASWK (Inquire set member of associated workstations) 145
GQCF (Inquire color facilities) 146
GQCHB (Inquire character base vector) 147
GQCHH (Inquire character height) 148
GQCHS (Inquire character spacing) 150
GQCHUP (Inquire character up vector) 151
GQCHW (Inquire character width) 152
GQCHXP (Inquire character expansion factor) 153
GQLCLP (Inquire clipping indicator) 154
GQCNNTN (Inquire current normalization transformation number) 155
GQCR (Inquire color representation) 155
GQDCH (Inquire default choice device data) 157
GQDDS (Inquire default deferral state values) 159
GQDLC (Inquire default locator device data) 160
GQDPK (Inquire default pick device data) 161
GQDSGA (Inquire dynamic modification of segment attributes) 163
GQDSDK (Inquire default stroke device data) 164
GQDSP (Inquire display space size) 166
GQDST (Inquire default string device data) 167
GQDLV (Inquire default valuator device data) 168
GQDWKA (Inquire dynamic modification of workstation attributes) 170
GQECI (Inquire list element of color indexes) 172
GQFFAI (Inquire list element of fill area indexes) 173
GQEGDP (Inquire list element of available generalized drawing primitives) 174
GQENTN (Inquire list element of normalization transformation numbers) 175
GQEPAL (Inquire list element of pattern indexes) 176
GQEPAL (Inquire list element of polylines indexes) 177
GQEP(A) (Inquire list element of polymarker indexes) 178
GQETXI (Inquire list element of text indexes) 180
GQEWW (Inquire list element of available workstation types) 181
GQFACI (Inquire fill area color index) 182
GQFAF (Inquire fill area facilities) 182
GQFAI (Inquire fill area index) 184
GQFAIS (Inquire fill area interior style) 185
GQFAR (Inquire fill area representation)  186
GQFASI (Inquire fill area style index) 187
GQGDP (Inquire generalized drawing primitive) 188
GQIQOV (Inquire input queue overflow) 189
GQLCS (Inquire locator device state) 191
GQLI (Inquire number of available logical input devices) 192
GQLN (Inquire linetype) 193
GQLWKS (Inquire level of GKS) 194
GQLWKC (Inquire maximum length of workstation state tables) 195
GQLWSC (Inquire linewidth scale factor) 196
GQMK (Inquire marker type) 197
GQMKSC (Inquire marker size scale factor) 198
GQMNTN (Inquire maximum normalization transformation number) 199
GQNT (Inquire normalization transformation) 200
GQOPS (Inquire operating state value) 201
GQOPSQ (Inquire name of open segment) 202
GQOPWK (Inquire set member of open workstations) 202
GQPA (Inquire pattern size) 203
GQPAF (Inquire pattern facilities) 204
GQPAR (Inquire pattern representation) 205
GQPARF (Inquire pattern reference point) 206
GQPCR (Inquire predefined color representation) 207
GQPFAR (Inquire predefined fill area representation) 208
GQPKID (Inquire pick identifier) 209
GQPKS (Inquire pick device state) 210
GQPLCI (Inquire polyline color index) 212
GQPLF (Inquire polyline facilities) 213
GQPLI (Inquire polyline index) 214
GQPLR (Inquire polyline representation) 215
GQPMCI (Inquire polymarker color index) 216
GQPMF (Inquire polymarker facilities) 217
GQPMI (Inquire polymarker index) 219
GQPMR (Inquire polymarker representation) 219
GQPPAR (Inquire predefined pattern representation) 221
GQPPARF (Inquire predefined pattern representation) 222
GQPPMR (Inquire predefined polymarker representation) 223
GQPTXR (Inquire predefined text representation) 224
GQPX (Inquire pixel) 226
GQPXIA (Inquire pixel array) 227
GQPXAD (Inquire pixel array dimensions) 228
GQSGA (Inquire segment attributes) 229
GQSGP (Inquire number of segment priorities supported) 231
GQSGUS (Inquire set member of segment names in use) 232
GQSGWK (Inquire set member of segment names on workstation) 233
GQSIM (Inquire more simultaneous events) 234
GQSKS (Inquire stroke device state) 235
GQSTS (Inquire string device state (FORTRAN only)) 236
GQSTSS (Inquire string device state) 238
GQTAL (Inquire text alignment) 240
GQTXCI (Inquire text color index) 241
GQTXF (Inquire text facilities) 241
GQTXFP (Inquire text font and precision) 243
GQTXI (Inquire text index) 244
GQTXP (Inquire text path) 245
GQTXR (Inquire text representation) 246
GQTXX (Inquire text text (VS FORTRAN only)) 247
GQTXXS (Inquire text text) 250
GQVLS (Inquire valuator device state) 252
GQWKC (Inquire workstation connection and type) 254
GQWKC (Inquire workstation category) 255
GQWKCL (Inquire workstation classification) 256
GQWKDU (Inquire workstation deferral and update states) 257
GQWK (Inquire workstation maximum numbers) 258
GQWK (Inquire workstation state) 259
GQK (Inquire workstation transformation) 260
GQPM (Inquire graphical input 6)
GQPM (Inquire graphical output 6)
GRDITM (Read item from GKS) 261
GRENSG (Rename segment) 262
GQRL (Request choice) 263
GRQCH (Request choice) 265
GRQPK (Request pick) 267
GRQS (Request stroke) 268
GQST (Request string (FORTRAN only)) 270
GQST (Request string) 271
GQVL (Request valuator) 272
GRSGWK (Redraw all segments on workstation) 274
GSASF (Set aspect source flags) 275
GSCHH (Set character height) 276
GSCHM (Set choice mode) 277
GSCHSP (Set character spacing) 278
GSCHUP (Set character up vector) 279
GSCHXP (Set character expansion factor) 280
GSCLIP (Set clipping indicator) 281
GSCOR (Set color representation) 282
GSDS (Set deferral state) 234
GSDTEC (Set detectability) 236
GSELNT (Select normalization transformation) 287
GSFAI (Set fill area color index) 288
GSF (Set fill area index) 289
GSFSAIS (Set fill area interior style) 290
GSFAR (Set fill area representation) 291
GSFASI (Set fill area style index) 293
GSHLIT (Set highlighting) 295
GSLCM (Set locator mode) 296
GSLN (Set linetype) 297
Index

bundle table 10
style 12
Initialize choice (GINCH) 114
Initialize GKS
    See Open GKS (GOPKS)
Initializelocator (GINLC) 116
Initialize pick (GINPK) 120
Initialize string (FORTRAN only) (GINST) 127
Initialize string (GINSTS) 129
Initialize stroke (GINSK) 132
Initialize valuator (GINVL) 132
input classes 6, 24
input functions 24
    Await event (GWAIT) 356
    Flush device events (GFLUSH) 102
    Get choice (GGTCH) 105
    Get locator (GGTLC) 107
    Get pick (GGTPK) 108
    Get string (FORTRAN only) (GGTST) 110
    Get string (GGTSTS) 111
    Get stroke (GGTSK) 109
    Get valuator (GGTVL) 112
    Initialize choice (GINCH) 114
    Initialize locator (GINLC) 116
    Initialize pick (GINPK) 120
    Initialize string (FORTRAN only) (GINST) 127
    Initialize string (GINSTS) 129
    Initialize stroke (GINSK) 132
    Initialize valuator (GINVL) 132
    Request choice (GRQCH) 263
    Request locator (GRQLC) 265
    Request pick (GRQPK) 267
    Request string (FORTRAN only) (GRQST) 270
    Request string (GROST) 271
    Request stroke (GROSK) 268
    Request valuator (GROVL) 272
    Sample choice (GSMCH) 299
    Sample locator (GSMLC) 304
    Sample pick (GSMPK) 305
    Sample string (FORTRAN only) (GSMST) 307
    Sample string (GSMSTS) 309
    Sample stroke (GSMSK) 306
    Sample valuator (GSMVL) 310
    Set choice mode (GSCHM) 277
    Set locator mode (GSLCM) 296
    Set pick mode (GSPKM) 316
    Set string mode (GSSTM) 329
    Set stroke mode (GSSKM) 328
    Set valuator mode (GSVLM) 342
input modes 6, 24
    Inquire aspect source flags (GQASF) 144
    Inquire character base vector (GQCHB) 147

hatch styles 12
hierarchy, GDDM 49, 56
highlighting 21
home page for GDDM xi

GSLWSC (Set linewidth scale factor) 298
GSMCH (Sample choice) 299
GSMK (Set marker type) 301
GSMKSC (Set marker size scale factor) 302
GSMLC (Sample locator) 304
GSMPK (Sample pick) 305
GSMST (Sample string (FORTRAN only)) 307
GSMT (Sample string) 309
GSMVL (Sample valuator) 310
GSPA (Set pattern size) 311
GSPAR (Set pattern representation) 312
GSPARF (Set pattern reference point) 314
GSPKID (Set pick identifier) 315
GSPKM (Set pick mode) 316
GSPLCI (Set polyline color index) 317
GSPLI (Set polyline index) 318
GSPLR (Set polyline representation) 319
GSPMCI (Set polymarker color index) 321
GSPMI (Set polymarker index) 322
GSPMR (Set polymarker representation) 323
GSSGP (Set segment priority) 325
GSSGT (Set segment transformation) 327
GSSKM (Set stroke mode) 328
GSSTM (Set string mode) 329
GSTXAL (Set text alignment) 331
GSTXCI (Set text color index) 333
GSTXFP (Set text font and precision) 334
GSTXI (Set text index) 337
GSTXP (Set text path) 338
GSTXR (Set text representation) 339
GSVIS (Set visibility) 341
GSVLM (Set valuator mode) 342
GSVP (Set viewport) 343
GSVPPIP (Set viewport input priority) 344
GSKPV (Set workstation viewport) 345
GSKWN (Set workstation window) 347
GSWN (Set window) 348
GTX (Text (VS FORTRAN only)) 349
GTXS (Text) 351
GUREC (Unpack data record (FORTRAN only)) 352
GURECS (Unpack data record) 354
GUWK (Update workstation) 355
GWAIT (Await event) 356
GWITM (Write item to GKSM) 358
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
Inquire choice device state (GQCHS) 149
Inquire clipping indicator (GQCLIP) 154
Inquire color facilities (GQCF) 146
Inquire color representation (GQCR) 155
Inquire current normalization transformation number (GQCNTN) 155
Inquire default choice device data (GQDCH) 157
Inquire default deferral state values (GQDDS) 159
Inquire default locator device data (GQDLC) 160
Inquire default pick device data (GQDPK) 161
Inquire default string device data (GQDST) 167
Inquire default stroke device data (GQDSDK) 164
Inquire default valuator device data (GQDVL) 168
Inquire display space size (GQDSP) 166
Inquire dynamic modification of segment attributes (GQDSGA) 163
Inquire dynamic modification of workstation attributes (GQDWKA) 170
Inquire fill area color index (GQFACI) 182
Inquire fill area facilities (GQFAF) 182
Inquire fill area index (GQFAI) 184
Inquire fill area interior style (GQFAIS) 185
Inquire fill area representation (GQFAR) 186
Inquire fill area style index (GQFASI) 187
Inquire generalized drawing primitive (GQQGP) 188
Inquire input queue overflow (GQIQOV) 189
Inquire level of GKS (GQLVS) 194
Inquire linetype (GQLN) 193
Inquire linewidth scale factor (GQLWSC) 196
Inquire list element of available generalized drawing primitives (GQEGDP) 174
Inquire list element of available workstation types (GQEWK) 181
Inquire list element of color indexes (GQECI) 172
Inquire list element of fill area indexes (GQEFAL) 173
Inquire list element of normalization transformation numbers (GQENTN) 175
Inquire list element of pattern indexes (GQEPAI) 176
Inquire list element of polyline indexes (GQEP) 177
Inquire list element of polymarker indexes (GQEPMI) 178
Inquire list element of text indexes (GQETY) 180
Inquire locator device state (GQLCS) 191
Inquire marker size scale factor (GQMSKSC) 198
Inquire marker type (GQMK) 197
Inquire maximum length of workstation state tables (GQLWK) 195
Inquire maximum normalization transformation number (GQMNTN) 199
Inquire name of open segment (GQOPSG) 202
Inquire normalization transformation (GQNT) 200
Inquire number of available logical input devices (GQLI) 192
Inquire number of segment priorities supported (GQSGP) 231
Inquire operating state value (GQOPS) 201
Inquire pattern facilities (GQPFAF) 204
Inquire pattern reference point (GQPARF) 206
Inquire pattern representation (GQPAL) 205
Inquire pattern size (GQP) 203
Inquire pick device state (GQPDK) 210
Inquire pick identifier (GQPKID) 209
Inquire pixel (GQPX) 226
Inquire pixel array (GQPX) 227
Inquire pixel array dimensions (GQXAD) 228
Inquire polyline color index (GQPLC) 212
Inquire polyline facilities (GQPFLF) 213
Inquire polyline index (GQPLI) 214
Inquire polyline representation (GQPLR) 215
Inquire polymarker color index (GQPMLC) 216
Inquire polymarker facilities (GQPMLF) 217
Inquire polymarker index (GQPML) 219
Inquire polymarker representation (GQPMP) 219
Inquire predefined color representation (GQPC) 207
Inquire predefined fill area representation (GQPFAR) 208
Inquire predefined pattern representation (GQPPAR) 221
Inquire predefined polyline representation (GQPPR) 222
Inquire predefined polymarker representation (GQPPLR) 223
Inquire predefined text representation (GQPTR) 224
Inquire segment attributes (GQSSA) 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 (GQTXA) 240
Inquire text color index (GQTXC) 241
Inquire text extent (GQTXXS) 250
Inquire text extent (VS FORTRAN only) (GQTXX) 247
Inquire text facilities (GQTXF) 241
Inquire text font and precision (GQTXFP) 243
Inquire text index (GQTXI) 244
Inquire text path (GOTXP) 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
Inquiry functions 26
  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
  Inquire choice device state (GQCHS) 149
  Inquire clipping indicator (GQCLIP) 154
  Inquire color facilities (GQCF) 146
  Inquire color representation (GQCR) 155
  Inquire current normalization transformation number (GQCNTN) 155
  Inquire default choice device data (GQDCH) 157
  Inquire default deferral state values (GQDDS) 159
  Inquire default locator device data (GQDLC) 160
  Inquire default pick device data (GQDPK) 161
  Inquire default string device data (GQDST) 167
  Inquire default stroke device data (GQDSK) 164
  Inquire default valuator device data (GQDVL) 168
  Inquire display space size (GQDSP) 166
  Inquire dynamic modification of segment attributes (GQDSGA) 163
  Inquire dynamic modification of workstation attributes (GQDWKA) 170
  Inquire fill area color index (GQFACI) 182
  Inquire fill area facilities (GQFAF) 182
  Inquire fill area index (GQFAI) 184
  Inquire fill area interior style (GQFAIS) 185
  Inquire fill area representation (GQFAR) 186
  Inquire fill area style index (GQFASI) 187
  Inquire generalized drawing primitive (GQGDP) 188
  Inquire input queue overflow (GQIQOV) 189
  Inquire level of GKS (GQLVKS) 194
  Inquire linetype (GQLN) 193
  Inquire linewidth scale factor (GQLWSC) 196
  Inquire list element of available generalized drawing primitives (GQEGDP) 174
  Inquire list element of available workstation types (GQEWK) 181
  Inquire list element of color indexes (GQECI) 172
  Inquire list element of fill area indexes (GQFAI) 173
Inquiry functions (continued)
  Inquire list element of normalization transformation numbers (GQENTN) 175
  Inquire list element of pattern indexes (GQEPAI) 176
  Inquire list element of polyline indexes (GQEPLI) 177
  Inquire list element of polymarker indexes (GQEPMI) 178
  Inquire list element of text indexes (GQETXI) 180
  Inquire locator device state (GQLCS) 191
  Inquire marker size scale factor (GQMKSC) 198
  Inquire marker type (GQMK) 197
  Inquire maximum length of workstation state tables (GQLWK) 195
  Inquire maximum normalization transformation number (GQMNNT) 199
  Inquire more simultaneous events (GQSIM) 234
  Inquire name of open segment (GQOPSG) 202
  Inquire normalization transformation (GQNT) 200
  Inquire number of available logical input devices (GQLI) 192
  Inquire number of segment priorities supported (GQSGP) 231
  Inquire operating state value (GQOPS) 201
  Inquire pattern facilities (GQPFAF) 204
  Inquire pattern reference point (GQPARF) 206
  Inquire pattern representation (GQPAR) 205
  Inquire pattern size (GQPA) 203
  Inquire pick device state (GQPKS) 210
  Inquire pick identifier (GQPKID) 209
  Inquire pixel (GQPX) 226
  Inquire pixel array (GQPXAD) 227
  Inquire pixel array dimensions (GQPDGA) 228
  Inquire polyline color index (GQPLCI) 212
  Inquire polyline facilities (GQPLFL) 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 (GQPFAF) 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 (GQOSA) 229
  Inquire set member of active workstations (GQACWK) 143
Index 445
inquiry functions (continued)
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 (GQTXXS) 250
Inquire text extent (VS FORTRAN only)
(GQTXX) 247
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
(GQWK) 254
Inquire workstation deferral and update states
(GQWKDU) 257
Inquire workstation maximum numbers
(GQWMN) 258
Inquire workstation state (GQWKS) 259
Inquire workstation transformation (GQWTX) 260
Insert segment (GINSG) 122
interfaces, external 43
nonreentrant 43, 44
reentrant 43, 44
system programmer 43, 45
interior style 12
Internet home page for GDDM xi
Interpret item (GIITM) 113
introduction
to GDDM-GKS 39
to GKS 1

L
language considerations for calls
FORTRAN 45
other languages 46
linkage, assembler-language rules 47
list of functions
alphabetical 67
by function type 71
locator devices 24
logical input devices 24

M
Message (GMSG) 134
Message (VS FORTRAN only) (GMSG) 133
messages, error 66
metafile functions 25
Get item type from GKSM (GGETTM) 106
Interpret item (GIITM) 113
Read item from GKSM (GRRITM) 261
Write item to GKSM (GWITM) 358
metafile structure 383
metafiles, GKS (GKSM) 7, 54
MI workstations 25
mixing GKS functions and other GDDM functions 49
MO workstations 25
modes, input 6
MVS 60
MVS/XA 60

N
names, external 41
nicknames 51, 52
nonreentrant interface 43, 44
normalization transformations 16
normalized device coordinates 6
numbers, GKS function 67

O
Open GKS (GOPKS) 135
Open workstation (GOPWK) 136
opening workstations 55
operating states 7, 360
output attributes 9
output functions 9
Cell array (GCA) 83
Fill area (GFA) 100
Generalized drawing primitive (GGDP) 103
Polyline (GPL) 138
Polymarker (GPM) 139
Text (GTXS) 351
Text (VS FORTRAN only) (GTX) 349
output primitives 6

P
Pack data record (FORTRAN only) (GPPREC) 140
Pack data record (GPPRECS) 141
packed data records 25
parameter
types 63
values 64
parameters, enumeration type 65
path, text 15
pattern reference point 39
pattern size 39
physical devices 50
pick
devices 24
input 21
pixel inquiry functions 40
PL/I
declarations of 47
language considerations for calls 46
plotters 53
polyline 10
attributes 10
types 10
Polyline (GPL) 138
polymarker 11
attributes 11
types 11
Polymarker (GPM) 139
primitives 6
printers 53
programs
dexample 405
sample (ROOM) 397
prompt and echo types 25

R
RCP
codes 63
ordered by code value 423
ordered by function name 419
in ADMASP call 45
introduction 45
Read item from GKSM (GRDITM) 261
Redraw all segments on workstation (GRSGWK) 274
reentrant interface 43, 44
related functions 66
Rename segment (GRENSG) 262
representations 10
Request choice (GRQCH) 263
request control parameter
See also RCP
introduction 45
Request locator (GRQLC) 265
request mode 24
Request pick (GRQPK) 267
Request string (FORTRAN only) (GRQST) 270
Request string (GRQSTS) 271
Request stroke (GRQSK) 268
Request valuator (GRQVL) 272
restrictions, GDDM-GKS 63, 66
differences from GKS standards 39
ROOM program 27
accumulating transformation matrixes 36
choice input 32
closing GKS 38
deleting a segment 37
Initializing GKS 29
locator input 33
opening workstations 29
pick identifiers 30
pick input 33
primitives and segments 30
program program (ROOM) 397
program outline 28
running the 1
segment transformations 34
source code 397
transformations 30
valuator input 36
rotation 22

S
Sample choice (GSMCH) 299
Sample locator (GSMLC) 304
Sample pick (GSMPK) 305
sample program (ROOM) 397
Sample string (FORTRAN only) (GSMST) 307
Sample string (GSMSTS) 309
Sample stroke (GSMSK) 306
Sample valuator (GSMVL) 310
scaling 22
segment attributes 21
segment functions 20
Associate segment with workstation (GASGWK) 81
Close segment (GCLSG) 87
Copy segment to workstation (GCSGWK) 90
Create segment (GCRSG) 89
Delete segment (GDSG) 92
Delete segment from workstation (GDSGWK) 93
Insert segment (GINSG) 122
Rename segment (GRENSG) 262
Set detectability (GSDTEC) 286
Set highlighting (GSHLIT) 295
Set segment priority (GSSGP) 325
Set segment transformation (GSSGT) 327
Set visibility (GSVIS) 341
segment priority 21
segment state list 8, 377
segment transformations 21
segments 6
Select normalization transformation (GSELNT) 287
Set aspect source flags (GSASF) 275
Set character expansion factor (GSCHXP) 280
Set character height (GSCHH) 276
Set character spacing (GSCHSP) 278
Set character up vector (GSCHUP) 279
Set choice mode (GSCHM) 277
Set clipping indicator (GSCLIP) 281
Set color representation (GSCR) 282
Set deferral state (GSDS) 284
Set detectability (GSDTEC) 286
Set fill area color index (GSFACI) 288
Set fill area index (GSFAI) 289
Set fill area interior style (GSFAIS) 290
Set fill area representation (GSFAR) 291
Set fill area style index (GSFASI) 293
Set highlighting (GSHLIT) 295
Set linetype (GSLN) 297
Set linewidth scale factor (GSLWSC) 298
Set locator mode (GSLCM) 296
Set marker size scale factor (GSMKSC) 302
Set marker type (GSMK) 301
Set pattern reference point (GSPARF) 314
Set pattern representation (GSPAR) 312
Set pattern size (GSPA) 311
Set pick identifier (GSPKID) 315
Set pick mode (GSPKM) 316
Set polyline color index (GSPLCI) 317
Set polyline index (GSPLICI) 318
Set polyline representation (GSPLR) 319
Set polymarker color index (GSPMCI) 321
Set polymarker index (GSPMII) 322
Set polymarker representation (GSPMR) 323
Set segment priority (GSSGP) 325
Set segment transformation (GSSGT) 327
Set string mode (GSSTM) 329
Set stroke mode (GSSKM) 328
Set text alignment (GSTXAL) 331
Set text color index (GSTXCI) 333
Set text font and precision (GSTXFP) 334
Set text index (GSTXI) 337
Set text path (GSTXP) 338
Set text representation (GSTXR) 339
Set valuator mode (GSVLM) 342
Set viewport (GSVP) 343
Set viewport input priority (GSVPIP) 344
Set visibility (GSVIS) 341
Set window (GSWN) 348
Set workstation viewport (GSWKVP) 345
Set workstation window (GSWKWN) 347
Severity code 44
Shifting 21
SPI (system programmer interface) 45
Start GKS

See Open GKS (GOPKS)

State list, GKS 361
String devices 24
Stroke devices 24
Style
index 12
interior 12

Subsystems, using GDDM-GKS under various syntax conventions 60
Assembler-language linkage 47
COBOL format 47
PL/I declarations 47
System programmer interface 43, 45

Terminate GKS

See Close GKS (GCLKS)

text 14
alignment 14, 40
attributes 14

Text (GTXS) 351

Text (VS FORTRAN only) (GTX) 349

Transformation functions 16

Select normalization transformation (GSELNT) 287
Set clipping indicator (GSCLIP) 281
Set viewport (GSVP) 343
Set viewport input priority (GSVPIP) 344
Set window (GSWN) 348
Set workstation viewport (GSWKVP) 345
Set workstation window (GSWKWN) 347

TSO 60

Unpack data record (FORTRAN only) (GUREC) 352
Unpack data record (GURECS) 354
Update workstation (GUWK) 355

Utility functions 25

Accumulate transformation matrix (GACTM) 78
Evaluate transformation matrix (GEVTM) 98
Pack data record (FORTRAN only) (GPREC) 140
Pack data record (GPRECS) 141
Unpack data record (FORTRAN only) (GUREC) 352
Unpack data record (GURECS) 354

Valuator devices 24
Values, parameter 64
Viewport 16
Workstation 17
Visibility 21
VM/CMS 61

Window
Workstation 17
World 16
Windowing with GDDM-GKS 50

WISS, workstation independent segment storage 54
workstation
  description tables  8, 368
  state list  8, 364
  transformations  17
  viewport  17
  window  17
workstation independent segment storage (WISS)  6, 54
workstations  5, 50
  how opened  55
world coordinates  6
world window  16
Write item to GKSM (GWITM)  358