

# INTERNATIONAL STANDARD

**ISO/IEC**  
**8651-4**

First edition  
1991-12-15

---

---

## **Information technology — Computer graphics — Graphical Kernel System (GKS) language bindings —**

### **Part 4: C**

*Technologies de l'information — Infographie — Système graphique de  
base (GKS) — Interface langage —*

*Partie 4: C*



## Contents

|        |                                                             |    |
|--------|-------------------------------------------------------------|----|
| 1      | Scope.....                                                  | 1  |
| 2      | Normative references.....                                   | 2  |
| 3      | The C Language Binding of GKS .....                         | 3  |
| 3.1    | Conformance .....                                           | 3  |
| 3.2    | Functions versus Macros .....                               | 3  |
| 3.3    | Character Strings .....                                     | 3  |
| 3.4    | Function Identifiers .....                                  | 3  |
| 3.5    | Registration.....                                           | 3  |
| 3.6    | Identifiers for Graphical Items.....                        | 4  |
| 3.7    | Return Values .....                                         | 4  |
| 3.8    | Header Files.....                                           | 4  |
| 3.9    | Memory Management.....                                      | 4  |
| 3.9.1  | Functions which Return Simple Lists.....                    | 5  |
| 3.9.2  | Functions which Return Complex Data Structures .....        | 5  |
| 3.10   | Error Handling.....                                         | 6  |
| 3.10.1 | Application Supplied Error Handlers .....                   | 6  |
| 3.10.2 | Error Codes.....                                            | 7  |
| 3.10.3 | C-specific GKS errors.....                                  | 7  |
| 3.11   | Colour Representations.....                                 | 7  |
| 3.12   | Storage of Multi-dimensional Arrays.....                    | 8  |
| 3.12.1 | Storage of 2*3 Matrices.....                                | 8  |
| 3.12.2 | Storage of Colour Arrays.....                               | 8  |
| 4      | Tables and Abbreviations .....                              | 9  |
| 4.1    | Abbreviation Policy in Construction of Identifiers.....     | 9  |
| 4.2    | Abbreviations Used .....                                    | 9  |
| 4.3    | Function Names .....                                        | 13 |
| 4.3.1  | List Ordered Alphabetically by Bound Name .....             | 13 |
| 4.3.2  | List Ordered Alphabetically by GKS Name.....                | 17 |
| 4.3.3  | List Ordered Alphabetically by Bound Name within Level..... | 21 |
| 5      | Type Definitions .....                                      | 27 |
| 5.1    | Mapping of GKS data types .....                             | 27 |
| 5.2    | Environmental Type Definitions.....                         | 27 |
| 5.3    | Implementation Dependent Type Definitions .....             | 27 |
| 5.4    | Implementation Independent Type Definitions .....           | 34 |
| 6      | Macro Definitions .....                                     | 49 |
| 6.1    | Function identifiers .....                                  | 49 |
| 6.2    | Error Codes.....                                            | 51 |
| 6.3    | Miscellaneous Macros .....                                  | 57 |
| 6.3.1  | Linetypes.....                                              | 57 |
| 6.3.2  | Marker Types.....                                           | 57 |
| 6.3.3  | Prompt and Echo Types.....                                  | 57 |
| 6.3.4  | Default Parameters of OPEN GKS.....                         | 57 |
| 7      | C GKS Function Interface .....                              | 58 |
| 7.1    | Notational Conventions .....                                | 58 |
| 7.2    | Control Functions .....                                     | 58 |
| 7.3    | Output Functions .....                                      | 60 |

© ISO/IEC 1991

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

|        |                                                                              |     |
|--------|------------------------------------------------------------------------------|-----|
| 7.4    | Output Attribute Functions .....                                             | 61  |
| 7.4.1  | Workstation Independent Primitive Attributes .....                           | 61  |
| 7.4.2  | Workstation Attributes .....                                                 | 64  |
| 7.5    | Transformation Functions .....                                               | 65  |
| 7.5.1  | Normalization Transformation .....                                           | 65  |
| 7.5.2  | Workstation transformation .....                                             | 66  |
| 7.6    | Segment Functions .....                                                      | 66  |
| 7.6.1  | Segment Manipulation Functions .....                                         | 66  |
| 7.6.2  | Segment Attribute Functions .....                                            | 68  |
| 7.7    | Input Functions .....                                                        | 68  |
| 7.7.1  | Initialization of Input Devices Functions .....                              | 68  |
| 7.7.2  | Setting the Mode of Input Devices Functions .....                            | 70  |
| 7.7.3  | Request Input Functions .....                                                | 71  |
| 7.7.4  | Sample Input Functions .....                                                 | 73  |
| 7.7.5  | Event Input Functions .....                                                  | 74  |
| 7.8    | Metafile Functions .....                                                     | 75  |
| 7.9    | Inquiry Functions .....                                                      | 76  |
| 7.9.1  | Inquiry Functions for Operating State Value .....                            | 76  |
| 7.9.2  | Inquiry Functions for GKS Description Table .....                            | 76  |
| 7.9.3  | Inquiry Functions for GKS State List .....                                   | 77  |
| 7.9.4  | Inquiry Functions for Workstation State List .....                           | 84  |
| 7.9.5  | Inquiry Functions for Workstation Description Table .....                    | 91  |
| 7.9.6  | Inquire functions for the Segment State List .....                           | 98  |
| 7.9.7  | Pixel Inquiries .....                                                        | 98  |
| 7.9.8  | Inquiry Functions for Error State List .....                                 | 99  |
| 7.10   | Utility Functions .....                                                      | 99  |
| 7.10.1 | Utility Functions in GKS .....                                               | 99  |
| 7.10.2 | Binding Specific Utilities .....                                             | 100 |
| 7.11   | Error Handling .....                                                         | 100 |
| A      | Compiled GKS/C Specification .....                                           | 102 |
| B      | Sample Programs .....                                                        | 145 |
| B.1    | STAR Program .....                                                           | 145 |
| B.2    | IRON Program .....                                                           | 147 |
| B.3    | MAP Program .....                                                            | 154 |
| B.4    | MANIPULATE Program .....                                                     | 156 |
| B.5    | SHOW LINE Program .....                                                      | 162 |
| C      | Metafile Items .....                                                         | 168 |
| D      | Short Function Identifiers .....                                             | 170 |
| E      | Memory Management .....                                                      | 175 |
| E.1    | Introduction .....                                                           | 175 |
| E.2    | Functions That Return Simple Lists .....                                     | 175 |
| E.2.1  | Operation of <code>ginq_list_line_inds</code> .....                          | 175 |
| E.3    | Functions That Return Structured Data .....                                  | 178 |
| E.3.1  | Operation of <code>gcreate_store</code> .....                                | 179 |
| E.3.2  | Operation of <code>ginq_stroke_st</code> and <code>ginq_pat_rep</code> ..... | 181 |
| E.3.3  | Operation of <code>gdel_store</code> .....                                   | 185 |
| F      | Function Lists .....                                                         | 188 |
| F.1    | Alphabetic by GKS Name .....                                                 | 188 |
| F.2    | Alphabetic by Binding Name .....                                             | 192 |

## Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 8651-4 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

ISO/IEC 8651 consists of the following parts, under the general title *Information technology — Computer graphics — Graphical Kernel System (GKS) language bindings*:

- Part 1: *FORTRAN 77*
- Part 2: *PASCAL*
- Part 3: *ADA*
- Part 4: *C*

Annexes A to F of this part of ISO/IEC 8651 are for information only.

## Introduction

The Graphical Kernel System (GKS) functional description is registered as ISO 7942 : 1985. As explained in the Scope and Field of Application of ISO 7942, that International Standard is specified in a language independent manner and needs to be embedded in language dependent layers (language bindings) for use with particular programming languages.

The purpose of this part of ISO/IEC 8651 is to define a standard binding for the C computer programming language.



# **Information technology – Computer graphics – Graphical Kernel System (GKS) language bindings -**

## **Part 4:**

### **C**

#### **1 Scope**

The Graphical Kernel System (GKS), ISO 7942 : 1985 , specifies a language independent nucleus of a graphics system. For integration into a programming language, GKS is embedded in a language dependent layer obeying the particular conventions of that language. This part of ISO/IEC 8651 specifies such a language dependent layer for the C language.

## 2 Normative references

The following standards contain provisions which, through reference in this text, constitute provision of this part of ISO/IEC 8651. At the time of publication, the editions indicated were valid. All standards are subject to revisions, and parties to agreements based on this part of ISO/IEC 8651 are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO 7942:1985, *Information processing systems – Computer graphics – Graphical Kernel System (GKS) functional description*.

ISO/IEC 8651-1:1988, *Information processing systems – Computer graphics – Graphical Kernel System (GKS) - language bindings - Part 1 : FORTRAN*.

ISO/IEC 8806-4:1991, *Information technology – Computer graphics – Graphical Kernel System for Three Dimensions (GKS-3D) language bindings - Part 4 : C*.

ISO/IEC 9899:1990, *Programming languages - C*.

ISO/IEC TR 9973:1988, *Information processing – Procedures for registration of graphical items*.

### 3 The C Language Binding of GKS

The C language binding of GKS shall be as described in clauses 3, 4, 5 and 6.

#### 3.1 Conformance

This part of ISO/IEC 8651 incorporates the rules of conformance defined in the GKS Standard (ISO 7942) for GKS implementations, with those additional requirements specifically defined for C bindings in GKS.

The following criteria shall determine conformance of an implementation to this part of ISO/IEC 8651:

In order to conform, an implementation of the C binding of GKS shall implement a specific level of GKS as specified in ISO 7942. It shall make visible all of the declarations in the C binding specified in this part of ISO/IEC 8651 for that same level of GKS and all lower levels and for a specific level of C.

Thus, for example, the syntax of the function names shall be precisely as specified in the binding and parameters shall be of the data types stated in the binding.

#### 3.2 Functions versus Macros

An implementation may substitute macros for functions. However, the macros shall be designed so that side-effects work properly. In general, a macro cannot be used to replace the error handling function `gerr_hand`. See also 3.10.

#### 3.3 Character Strings

The C language represents character strings as an array of characters terminated by the null character (i.e. `'\0'`). This means that the null character is not usable as a printable character.

#### 3.4 Function Identifiers

The function names of GKS are all mapped to C functions which begin with the letter `g`. Words and phrases used in the GKS function names are often abbreviated in the representation and are always separated with the underscore character `"_"`. The set of such abbreviations is given in 4.2, and the resulting C function names are listed in 4.3. For example, the abbreviation for the GKS function DELETE SEGMENT FROM WORKSTATION is `gdel_seg_ws`. `del`, `seg`, and `ws` are abbreviations for DELETE, SEGMENT, and WORKSTATION. The conjunctive FROM is mapped to the null string.

The C standard (ISO/IEC 9899) requires that compilers recognize internal identifiers which are distinct in at least 31 characters. That standard also requires that external identifiers (i.e. those seen by the linker) be recognized to a minimum of six characters, independent of case.

Implementations which run in environments where two distinct C internal identifiers would be equivalent, if they were both external identifiers, shall include a set of `#defines` in the header file which equate the long names to a set of short names. A possible set of short names for a compiler that accepts only eight characters for external definitions may be found in annex D.

#### 3.5 Registration

ISO 7942 reserves certain value ranges for registration<sup>1</sup> as graphical items. The registered graphical items will be bound to the C programming language (and other programming languages). The registered item binding will be consistent with the binding presented in this part of ISO/IEC 8651.

<sup>1</sup>) For the purpose of this part of ISO/IEC 8651 and according to the rules for the designation and operation of registration authorities in the ISO/IEC Directives, the ISO and IEC councils have designated the National Institute of Standards and Technology (Institute of Computer Sciences and Technology), A-266 Technology Building, Gaithersburg, MD 20899, USA to act as registration authority.

## Identifiers for Graphical Items

## The C Language Binding of GKS

## 3.6 Identifiers for Graphical Items

Generalized Drawing Primitives and Escape functions are referenced via identifiers. This part of ISO/IEC 8651 specifies the format of the identifiers but it does not specify the registration of the identifiers. The identifiers are used as arguments to the functions `ggdp` and `gescape`.

An implementation may also represent GDPs and Escapes as separate functions, but this is not required.

There are two formats for these identifiers. One format is for registered GDPs and Escapes and the other format is for unregistered GDPs and Escapes.

The format for registered GDP identifiers is:

```
#define    GGDP_Rn      (n)    /* 'n' is the registered GDP id. */
```

The format for unregistered GDP identifiers is:

```
#define    GGDP_Un      (-n)   /* 'n' is implementation dependent */
```

The format for registered Escape function identifiers is:

```
#define    GESCAPE_Rn   (n)    /* 'n' is the registered Escape id. */
```

The format for unregistered Escape function identifiers is:

```
#define    GESCAPE_Un   (-n)   /* 'n' is implementation dependent */
```

## 3.7 Return Values

All GKS/C functions return `void`.

## 3.8 Header Files

C provides a mechanism to allow external files to be included in a compilation. Clause 5 of this part of ISO/IEC 8651 describes the data types that shall be defined in the file `gks.h` which should be included in any application program that intends to use GKS via the C binding.

This part of ISO/IEC 8651 uses the data type `size_t` (as a field in the data type `Gdata`). The type `size_t` is environment-dependent (e.g. `int`, `long int`, `unsigned int`) and is defined in the file `<stddef.h>`. Therefore the file `gks.h` shall also include the file `<stddef.h>`.

Additional implementation-dependent items may be placed in this file if needed. These items should start with the sentinel "G" or "g", as far as applicable.

The file `gks.h` shall also contain external declarations for all GKS/C functions because they return `void`. For example, the declaration for the function `gopen_gks` would look like this:

```
extern    void    gopen_gks(char *err_file, size_t mem_units);
```

## 3.9 Memory Management

The application shall allocate the memory needed for the data returned by the implementation. In general, the application will allocate a C structure and pass a pointer to that structure to an inquiry routine, which will then place information into the structure. However, a number of inquiry functions return variable length data, the length of which is not known *a priori* by the application.

These functions fall into two classes. One class of functions returns a simple, homogeneous, list of items. For example, the function INQUIRE SET OF SEGMENT NAMES returns a list of the segment names in use. The other class returns complex, heterogeneous data structures. For example, the function INQUIRE LOCATOR DEVICE STATE returns the device state which includes a locator data record; the data record can contain arbitrarily complex implementation-defined data structures. The binding of these two classes of functions is described in detail below. Subclause 3.10 describes the errors that can be invoked during execution of functions which use the memory management policy.

## The C Language Binding of GKS

## Memory Management

## 3.9.1 Functions which Return Simple Lists

Inquiry functions which return a list of items are bound such that the application can inquire about a portion of the list. This list is a subset of the implementation's internal list and is called the application's list. This allows the application to process the implementation's list in a piecewise manner rather than all at once.

The application allocates the memory for a list and passes that list to the implementation. The implementation places the results of the inquiry into the list. In order to support this policy of memory management, three additional parameters have been added to functions which return lists:

- a) `num_elems_appl_list`: An integer input parameter which is the length of the application's list. The value of `num_elems_appl_list` indicates the number of items (i.e. list elements) which will fit into the application list. A value of 0 is valid and allows the application to determine the size of the implementation's list (which is returned via `num_elems_impl_list`) without having the implementation return any of the elements of its list. If `num_elems_appl_list` is negative, `GE_APPL_LIST_LENGTH_LT_ZERO` is returned as the value of the error indicator parameter.
- b) `start_ind`: An integer input parameter which is an index into the implementation's list. (Index 0 is the first element of both the implementation's and application's list.) `start_ind` indicates the first item in the implementation's list that is copied into index 0 of the application's list. Items are copied sequentially from the implementation's list into the application's list until the application's list is full or there are no more items in the implementation's list. If `start_ind` is out of range, error `GE_START_IND_INVAL` is returned as the value of the error indicator parameter.
- c) `num_elems_impl_list`: An output parameter which is a pointer to an integer. The implementation stores into this parameter the number of items that are in the implementation's list.

In annex E, a possible underlying mechanism is described.

## 3.9.2 Functions which Return Complex Data Structures

The data returned by the `ESCAPE` function and the functions which return input device data records or pattern tables can be complex in structure. They cannot be represented by a simple list of items. It would be an onerous task for the application to have to allocate and prepare data structures for these routines. In order to facilitate this task of using these inquiry functions, the binding defines a new resource, called a *Store*, to manage the memory for these functions.

The *Store* resource is opaque to the application. The application does not know the structure of the *Store* or how it is implemented. The *Store* is defined as a `void *`. This part of ISO/IEC 8651 defines two new functions which create (in `CREATE STORE`, bound as `gcreate_store`) and delete (in `DELETE STORE`, bound as `del_store`) a *Store*.

A *Store* is used by the implementation to manage the memory needed by the functions which return complex data structures. Without specifying an implementation of a *Store*, it is safe to say that it will contain and control memory needed to hold the data returned by these functions and also contain some bookkeeping information about the contents and size of the memory.

The semantics of the *Store* resource provide two levels of memory management. The implementation is responsible for managing the memory at a low level because it uses, reuses, allocates and deallocates memory from the system in order to return information to the application. But the application is ultimately responsible for managing the memory at a high level because it creates and deletes *Stores*.

A *Store* is passed as a parameter to a function returning complex data structures. Another parameter to this function is a pointer to a pointer to a structure which defines the format of the returned data. The *Store* contains memory for the structure and any additional memory referenced by fields within the structure. The application accesses the returned data through its pointer to the structure. It does not use the *Store* to access the data.

A *Store* continues to hold the information from the function until the *Store* is deleted by the `DELETE STORE` function or until the *Store* is used as an argument to a subsequent function, which returns complex

**Memory Management****The C Language Binding of GKS**

data structures. At that time, the old information is replaced with the new. Thus multiple calls to functions overwrite the contents of a *Store*. A *Store* only contains the results of the last function.

This part of ISO/IEC 8651 defines two new errors that can occur when using or creating a *Store*; these errors are described in 3.10.3. For most functions using a *Store*, these and other errors are returned via the "error indicator" parameter. However, the function ESCAPE does not have an error indicator parameter. For this function, the error reporting mechanism is used when an error is encountered. For this function, the implementation shall, in addition to reporting the error, set the pointer to the returned data to NULL when an error occurs. See the binding of these functions for more information.

The definitions for the functions CREATE STORE and DELETE STORE follow:

---

|                     |                               |            |
|---------------------|-------------------------------|------------|
| <b>CREATE STORE</b> | <b>GKOP, WSOP, WSAC, SGOP</b> | <b>L0a</b> |
| <i>Parameters:</i>  |                               |            |

|     |                 |       |
|-----|-----------------|-------|
| Out | error indicator | I     |
| Out | store           | STORE |

**Effect:** Creates a *Store* and returns a handle to it in the output parameter *store*. If the *Store* cannot be created, the *store* parameter is set to NULL and the error indicator is set to one of the following error values:

|      |                                                                                     |
|------|-------------------------------------------------------------------------------------|
| 8    | GKS not in proper state; GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP |
| 2203 | Error while allocating Store                                                        |

**Errors:** None.

---

|                     |                               |            |
|---------------------|-------------------------------|------------|
| <b>DELETE STORE</b> | <b>GKOP, WSOP, WSAC, SGOP</b> | <b>L0a</b> |
| <i>Parameters:</i>  |                               |            |

|        |                 |       |
|--------|-----------------|-------|
| Out    | error indicator | I     |
| In/Out | store           | STORE |

**Effect:** Deletes the *Store* and all internal resources associated with it. If there is not an error, the parameter *store* will be set to NULL to signify that it is no longer valid. If an error is detected, the error indicator is set to one of the following values:

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

**Errors:** None.

In 7.10.2, the C specification of these functions is given. In annex E, a possible underlying mechanism is illustrated.

### 3.10 Error Handling

#### 3.10.1 Application Supplied Error Handlers

User-defined error handlers shall accept the same arguments as the standard error handler. The user-defined error handler is specified by the utility function (see also 7.10.2)

**SET ERROR HANDLER****GKCL, GKOP, WSOP, WSAC, SGOP****L0a***Parameters:*

|     |                             |          |
|-----|-----------------------------|----------|
| In  | New error handling function | Function |
| Out | Old error handling function | Function |

**Effect:** Sets the GKS error handling function to *New error handling function* and returns the current error handling function to *Old error handling function*.

**Errors:** None.

Application defined error handling functions accept the same arguments as the standard error handler. They may invoke the standard error logging function ERROR LOGGING.

ISO 7942 defines the initial error handling function to be ERROR HANDLING, that is, the value of the parameter *Old error handling function* points to ERROR HANDLING, when SET ERROR HANDLER is called for the first time.

When the application changes the error handling function, the implementation will invoke the new function when an error is detected. If the application calls the default error handling function ERROR HANDLING, ERROR HANDLING will always call the function ERROR LOGGING.

If *New error handler* is not a valid pointer, the error handling will automatically be done by the standard error handler ERROR HANDLING.

User-defined error handlers may invoke the standard error logging function ERROR LOGGING.

**3.10.2 Error Codes**

Hard coding numbers into a program decreases its maintainability. Therefore, this part of ISO/IEC 8651 defines a set of constants for the GKS error numbers. Each error constant begins with the characters **GE\_**. See also 6.2 for the error macros.

**3.10.3 C-specific GKS errors**

This part of ISO/IEC 8651 knows some specific error messages. This section gives the messages and the error macros.

| Value | Message                                                                                                                                     |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------|
| 2200  | Start index out of range<br>Is issued when the start index is less than zero or larger than the last element in the implementation's list   |
| 2201  | Length of application list is negative<br>Is issued when the length of the application's list is less than zero                             |
| 2202  | Enumeration type out of range<br>Is issued when a parameter value whose type is an enumeration is out range of the enumeration              |
| 2203  | Error while allocating Store<br>Is issued when an error is detected during CREATE STORE                                                     |
| 2204  | Error while allocating memory for Store<br>Is issued when a function using a <i>Store</i> is unable to allocate memory for the <i>Store</i> |

**3.11 Colour Representations**

A union type definition is used for colour bundles, guaranteeing upward compatibility with GKS-3D/C (ISO/IEC 8806-4), which supports at least two colour models (RGB and CIE L\*u\*v\* 1976).

## Storage of multi-dimensional arrays"

## The C Language Binding of GKS

## 3.12 Storage of Multi-dimensional Arrays

## 3.12.1 Storage of 2\*3 Matrices

The entries of `Gtran_matrix` data types shall be stored such that the segment transformation is defined by

$$\begin{aligned} \text{Tp.x} &= \text{mat}[0,0]*\text{p.x} + \text{mat}[0,1]*\text{p.y} + \text{mat}[0,2]; \\ \text{Tp.y} &= \text{mat}[1,0]*\text{p.x} + \text{mat}[1,1]*\text{p.y} + \text{mat}[1,2]; \end{aligned}$$

where `p` is a 2D point, `Tp` its transformation and `mat` is of type `Gtran_matrix`.

## 3.12.2 Storage of Colour Arrays

The entries of `Gpat_rep` data types shall be stored such that the colour index at the  $(i,j)$ -th entry is given by

$$\begin{aligned} \text{Colr\_ind}_{i,j} &= \text{colr\_rect.colr\_array}[i + \text{DX}*j]; \\ i &= 0, \dots, \text{DX} - 1; & j &= 0, \dots, \text{DY} - 1; \\ \text{DX} &= \text{colr\_rect.dims.size\_x}; & \text{DY} &= \text{colr\_rect.dims.size\_y}; \end{aligned}$$

where `colr_rect` is of type `Gpat_rep`.

## 4 Tables

### 4.1 Abbreviation Policy in Construction of Identifiers

In the construction of the several data types, function names, etc., the following policy is applied:

- 1) All identifiers in the C binding are abbreviated using the same abbreviations for every component and using underscores to denote blanks;
- 2) The plural of an expression is constructed by adding an "s" after its abbreviation; so, for example, "vector" is abbreviated to "vec" and "vectors" is abbreviated to "vecs"; if an expression is mapped to NULL, so will be its plural;
- 3) Digits are also preceded by underscores;
- 4) The words POLYLINE, POLYMARKER and FILL AREA are not abbreviated in the output primitive function names; in all other cases they are abbreviated using the list in 4.2;
- 5) Construction of GKS/C identifiers:
  - a) Function names:  
"g" (lower case) followed by abbreviated function name in lower case;
  - b) Data types:  
"G" (upper case) followed by abbreviated data type in lower case;
  - c) Fields of data types; the following refinements are used: "redundant" (words in the field name that are identical to those in the structure name) parts are omitted, if the context allows this; thus the colour index in the field of Gxxx\_bundle is abbreviated to colr\_ind, because the context makes clear which colour index is used;
  - d) Function macros:  
"Gfn\_" followed by abbreviation of function name;
  - e) GKSM item types:  
"Gksm\_" followed by abbreviation of item name;
  - f) Error macros:  
"GE\_" followed by some abbreviated expression;
  - g) Fields of enumeration types:  
"G" (upper case) followed by a prefix followed by an abbreviation of the field name; this prefix is constant for each enumeration field; all the fields are in upper case;

### 4.2 Table of Abbreviations Used

In this table, only words which are abbreviated are listed. They are used for

function names;  
data types;  
fields of data types;  
error macros;  
GKSM item types.

The word "NULL" denotes those words which are deleted completely when forming function names or data types.

| Word or Phrase | Abbreviation |
|----------------|--------------|
| accepted       | NULL         |

## Table of Abbreviations Used

## Tables

|                                      |        |
|--------------------------------------|--------|
| accumulate                           | accum  |
| action                               | NULL   |
| actual                               | act    |
| addition                             | add    |
| alignment                            | align  |
| allocate                             | alloc  |
| and                                  | NULL   |
| application                          | appl   |
| arithmetic                           | arith  |
| as soon as possible                  | asap   |
| aspect source flag                   | asf    |
| associate(d)                         | assoc  |
| at some time                         | asti   |
| at                                   | NULL   |
| attribute                            | attr   |
| availability                         | avail  |
| available                            | avail  |
| before the next interaction globally | bnig   |
| before the next interaction locally  | bnil   |
| between                              | NULL   |
| buffer                               | buf    |
| cannot                               | cant   |
| category                             | cat    |
| centre                               | ctr    |
| character                            | char   |
| classification                       | class  |
| clipping                             | clip   |
| colour                               | colr   |
| concatenation                        | concat |
| conditional(ly)                      | cond   |
| connection                           | conn   |
| control                              | ctrl   |
| coordinate                           | coord  |
| corner                               | cnr    |
| current                              | cur    |
| dashed                               | dash   |
| default                              | def    |
| deferral                             | defer  |
| delete                               | del    |
| deletion                             | del    |
| detectability                        | det    |
| detectable                           | det    |
| device coordinate(s)                 | dc     |
| device(s)                            | NULL   |
| digital                              | digit  |
| dimension                            | dim    |
| display                              | disp   |
| dotted                               | dot    |
| duplicate                            | dup    |
| dynamic                              | dyn    |
| equal                                | eq     |
| error                                | err    |

## Tables

evaluate  
 expansion  
 facility  
 factor  
 fill area  
 from  
 function  
 generalized drawing primitive  
 gks closed  
 gks open  
 gksm  
 graphical  
 handling  
 height  
 highlighted  
 highlighting  
 horizontal  
 identifier  
 immediate(ly)  
 implicit regeneration  
 implicit  
 in use  
 in  
 index  
 indicator  
 individual  
 initial  
 initialize  
 input/output  
 input  
 inquire  
 integer  
 interior  
 invalid  
 invisible  
 length  
 less than or equal  
 less than  
 level x  
 library  
 locator  
 logical  
 mapping  
 maximum  
 memory  
 metafile input  
 metafile output  
 minimum  
 modification  
 monochrome  
 necessary  
 nominal

## Table of Abbreviations Used

eval  
 expan  
 fac  
 NULL  
 fill  
 NULL  
 func  
 gdp  
 gkcl  
 gkop  
 NULL  
 graph  
 hand  
 ht  
 highl  
 highl  
 hor  
 id  
 imm  
 irg  
 impl  
 NULL  
 NULL  
 ind  
 ind  
 indiv  
 init  
 init  
 io  
 in  
 inq  
 int  
 int  
 inval  
 invis  
 NULL  
 le  
 lt  
 lx (with x = 0a, 0b, ...)  
 lib  
 loc  
 NULL  
 map  
 max  
 mem  
 mi  
 mo  
 min  
 mod  
 monochr  
 nec  
 nom

## Table of Abbreviations Used

## Tables

|                                 |         |
|---------------------------------|---------|
| normal                          | norm    |
| normalization                   | norm    |
| normalized device coordinate(s) | ndc     |
| number                          | num     |
| of                              | NULL    |
| on                              | NULL    |
| operating                       | op      |
| output and input                | outin   |
| output                          | out     |
| overflow                        | overf   |
| parallel(ogram)                 | paral   |
| pattern                         | pat     |
| pending                         | pend    |
| pointer                         | ptr     |
| polyline                        | line    |
| polymarker                      | marker  |
| position                        | pos     |
| precision                       | prec    |
| predefined                      | pred    |
| presence                        | pres    |
| primitive                       | prim    |
| priority                        | pri     |
| prompt and echo type            | pet     |
| prompt                          | pr      |
| queue                           | NULL    |
| record                          | NULL    |
| rectangle                       | rect    |
| red green blue (colour model)   | rgb     |
| reference                       | ref     |
| regeneration                    | regen   |
| registered                      | r       |
| relative                        | rel     |
| representation                  | rep     |
| request(ed)                     | req     |
| rubber                          | rub     |
| scale factor                    | NULL    |
| segment open                    | sgop    |
| segment                         | seg     |
| select                          | sel     |
| simultaneous(ly)                | simult  |
| spacing                         | space   |
| specification                   | specif  |
| specified                       | specif  |
| state                           | st      |
| storage                         | store   |
| supported                       | NULL    |
| suppressed                      | suppr   |
| to                              | NULL    |
| tracking                        | track   |
| transform(ation)                | tran    |
| unavailable                     | unavail |
| undefined                       | undef   |

## Tables

## Table of Abbreviations Used

|                                       |       |
|---------------------------------------|-------|
| undetectable                          | undet |
| unregistered                          | u     |
| update                                | upd   |
| used                                  | NULL  |
| valuator                              | val   |
| value                                 | NULL  |
| vector                                | vec   |
| vertical                              | vert  |
| viewport                              | vp    |
| visibility                            | vis   |
| visible                               | vis   |
| which                                 | NULL  |
| window                                | win   |
| with                                  | NULL  |
| workstation active                    | wsac  |
| workstation independent segment store | wiss  |
| workstation open                      | wsop  |
| workstation                           | ws    |
| world coordinate                      | wc    |

## 4.3 Function Names

## 4.3.1 List Ordered Alphabetically by Bound Name

|                      |                                    |
|----------------------|------------------------------------|
| gaccum_tran_matrix   | ACCUMULATE TRANSFORMATION MATRIX   |
| gactivate_ws         | ACTIVATE WORKSTATION               |
| gassoc_seg_ws        | ASSOCIATE SEGMENT WITH WORKSTATION |
| gawait_event         | AWAIT EVENT                        |
| gcell_array          | CELL ARRAY                         |
| gclear_ws            | CLEAR WORKSTATION                  |
| gclose_gks           | CLOSE GKS                          |
| gclose_seg           | CLOSE SEGMENT                      |
| gclose_ws            | CLOSE WORKSTATION                  |
| gcopy_seg_ws         | COPY SEGMENT FROM WORKSTATION      |
| gcreate_seg          | CREATE SEGMENT                     |
| gcreate_store        | CREATE STORE (GKS/C only)          |
| gdeactivate_ws       | DEACTIVATE WORKSTATION             |
| gdel_seg             | DELETE SEGMENT                     |
| gdel_seg_ws          | DELETE SEGMENT FROM WORKSTATION    |
| gdel_store           | DELETE STORE (GKS/C only)          |
| gemergency_close_gks | EMERGENCY CLOSE GKS                |
| gerr_hand            | ERROR HANDLING                     |
| gerr_log             | ERROR LOGGING                      |
| gescape              | ESCAPE                             |
| geval_tran_matrix    | EVALUATE TRANSFORMATION MATRIX     |
| gfill_area           | FILL AREA                          |
| gflush_events        | FLUSH DEVICE EVENTS                |
| ggdp                 | GENERALIZED DRAWING PRIMITIVE      |
| gget_choice          | GET CHOICE                         |
| gget_item_type       | GET ITEM TYPE FROM GKSM            |
| gget_loc             | GET LOCATOR                        |
| gget_pick            | GET PICK                           |

## Function Names

## Tables

|                          |                                                          |
|--------------------------|----------------------------------------------------------|
| gget_string              | GET STRING                                               |
| gget_stroke              | GET STROKE                                               |
| gget_val                 | GET VALUATOR                                             |
| ginit_choice             | INITIALISE CHOICE                                        |
| ginit_loc                | INITIALISE LOCATOR                                       |
| ginit_pick               | INITIALISE PICK                                          |
| ginit_string             | INITIALISE STRING                                        |
| ginit_stroke             | INITIALISE STROKE                                        |
| ginit_val                | INITIALISE VALUATOR                                      |
| ginq_asfs                | INQUIRE ASPECT SOURCE FLAGS                              |
| ginq_char_base_vec       | INQUIRE CHARACTER BASE VECTOR                            |
| ginq_char_expan          | INQUIRE CHARACTER EXPANSION FACTOR                       |
| ginq_char_ht             | INQUIRE CHARACTER HEIGHT                                 |
| ginq_char_space          | INQUIRE CHARACTER SPACING                                |
| ginq_char_up_vec         | INQUIRE CHARACTER UP VECTOR                              |
| ginq_char_width          | INQUIRE CHARACTER WIDTH                                  |
| ginq_choice_st           | INQUIRE CHOICE DEVICE STATE                              |
| ginq_clip                | INQUIRE CLIPPING                                         |
| ginq_colr_fac            | INQUIRE COLOUR FACILITIES                                |
| ginq_colr_rep            | INQUIRE COLOUR REPRESENTATION                            |
| ginq_cur_indiv_attrs     | INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES              |
| ginq_cur_norm_tran_num   | INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER      |
| ginq_cur_pick_id         | INQUIRE CURRENT PICK IDENTIFIER VALUE                    |
| ginq_cur_prim_attrs      | INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES               |
| ginq_def_choice_data     | INQUIRE DEFAULT CHOICE DEVICE DATA                       |
| ginq_def_defer_sts       | INQUIRE DEFAULT DEFERRAL STATE VALUES                    |
| ginq_def_loc_data        | INQUIRE DEFAULT LOCATOR DEVICE DATA                      |
| ginq_def_pick_data       | INQUIRE DEFAULT PICK DEVICE DATA                         |
| ginq_def_string_data     | INQUIRE DEFAULT STRING DEVICE DATA                       |
| ginq_def_stroke_data     | INQUIRE DEFAULT STROKE DEVICE DATA                       |
| ginq_def_val_data        | INQUIRE DEFAULT VALUATOR DEVICE DATA                     |
| ginq_disp_space_size     | INQUIRE DISPLAY SPACE SIZE                               |
| ginq_dyn_mod_seg_attrs   | INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES       |
| ginq_dyn_mod_ws_attrs    | INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES   |
| ginq_fill_colr_ind       | INQUIRE FILL AREA COLOUR INDEX                           |
| ginq_fill_fac            | INQUIRE FILL AREA FACILITIES                             |
| ginq_fill_ind            | INQUIRE FILL AREA INDEX                                  |
| ginq_fill_int_style      | INQUIRE FILL AREA INTERIOR STYLE                         |
| ginq_fill_rep            | INQUIRE FILL AREA REPRESENTATION                         |
| ginq_fill_style_ind      | INQUIRE FILL AREA STYLE INDEX                            |
| ginq_gdp                 | INQUIRE GENERALIZED DRAWING PRIMITIVE                    |
| ginq_in_overf            | INQUIRE INPUT QUEUE OVERFLOW                             |
| ginq_level_gks           | INQUIRE LEVEL OF GKS                                     |
| ginq_line_colr_ind       | INQUIRE POLYLINE COLOUR INDEX                            |
| ginq_line_fac            | INQUIRE POLYLINE FACILITIES                              |
| ginq_line_ind            | INQUIRE POLYLINE INDEX                                   |
| ginq_line_rep            | INQUIRE POLYLINE REPRESENTATION                          |
| ginq_linetype            | INQUIRE LINETYPE                                         |
| ginq_linewidth           | INQUIRE LINEWIDTH SCALE FACTOR                           |
| ginq_list_avail_gdps     | INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES |
| ginq_list_avail_ws_types | INQUIRE LIST OF AVAILABLE WORKSTATION TYPES              |
| ginq_list_colr_inds      | INQUIRE LIST OF COLOUR INDICES                           |

## Tables

## Function Names

|                          |                                                      |
|--------------------------|------------------------------------------------------|
| ginq_list_fill_inds      | INQUIRE LIST OF FILL AREA INDICES                    |
| ginq_list_line_inds      | INQUIRE LIST OF POLYLINE INDICES                     |
| ginq_list_marker_inds    | INQUIRE LIST OF POLYMARKER INDICES                   |
| ginq_list_norm_tran_nums | INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS |
| ginq_list_pat_inds       | INQUIRE LIST OF PATTERN INDICES                      |
| ginq_list_text_inds      | INQUIRE LIST OF TEXT INDICES                         |
| ginq_loc_st              | INQUIRE LOCATOR DEVICE STATE                         |
| ginq_marker_colr_ind     | INQUIRE POLYMARKER COLOUR INDEX                      |
| ginq_marker_fac          | INQUIRE POLYMARKER FACILITIES                        |
| ginq_marker_ind          | INQUIRE POLYMARKER INDEX                             |
| ginq_marker_rep          | INQUIRE POLYMARKER REPRESENTATION                    |
| ginq_marker_size         | INQUIRE MARKER SIZE SCALE FACTOR                     |
| ginq_marker_type         | INQUIRE MARKER TYPE                                  |
| ginq_max_norm_tran_num   | INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER  |
| ginq_max_ws_st_tables    | INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES   |
| ginq_more_simult_events  | INQUIRE MORE SIMULTANEOUS EVENTS                     |
| ginq_name_open_seg       | INQUIRE NAME OF OPEN SEGMENT                         |
| ginq_norm_tran           | INQUIRE NORMALIZATION TRANSFORMATION                 |
| ginq_num_avail_in        | INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES    |
| ginq_num_seg_pris        | INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED       |
| ginq_op_st               | INQUIRE OPERATING STATE VALUE                        |
| ginq_pat_fac             | INQUIRE PATTERN FACILITIES                           |
| ginq_pat_ht_vec          | INQUIRE PATTERN HEIGHT VECTOR                        |
| ginq_pat_ref_point       | INQUIRE PATTERN REFERENCE POINT                      |
| ginq_pat_rep             | INQUIRE PATTERN REPRESENTATION                       |
| ginq_pat_width_vec       | INQUIRE PATTERN WIDTH VECTOR                         |
| ginq_pick_st             | INQUIRE PICK DEVICE STATE                            |
| ginq_pixel               | INQUIRE PIXEL                                        |
| ginq_pixel_array         | INQUIRE PIXEL ARRAY                                  |
| ginq_pixel_array_dims    | INQUIRE PIXEL ARRAY DIMENSIONS                       |
| ginq_pred_colr_rep       | INQUIRE PREDEFINED COLOUR REPRESENTATION             |
| ginq_pred_fill_rep       | INQUIRE PREDEFINED FILL AREA REPRESENTATION          |
| ginq_pred_line_rep       | INQUIRE PREDEFINED POLYLINE REPRESENTATION           |
| ginq_pred_marker_rep     | INQUIRE PREDEFINED POLYMARKER REPRESENTATION         |
| ginq_pred_pat_rep        | INQUIRE PREDEFINED PATTERN REPRESENTATION            |
| ginq_pred_text_rep       | INQUIRE PREDEFINED TEXT REPRESENTATION               |
| ginq_seg_attrs           | INQUIRE SEGMENT ATTRIBUTES                           |
| ginq_set_active_wss      | INQUIRE SET OF ACTIVE WORKSTATIONS                   |
| ginq_set_assoc_wss       | INQUIRE SET OF ASSOCIATED WORKSTATIONS               |
| ginq_set_open_wss        | INQUIRE SET OF OPEN WORKSTATIONS                     |
| ginq_set_seg_names       | INQUIRE SET OF SEGMENT NAMES IN USE                  |
| ginq_set_seg_names_ws    | INQUIRE SET OF SEGMENT NAMES ON WORKSTATION          |
| ginq_string_st           | INQUIRE STRING DEVICE STATE                          |
| ginq_stroke_st           | INQUIRE STROKE DEVICE STATE                          |
| ginq_text_align          | INQUIRE TEXT ALIGNMENT                               |
| ginq_text_colr_ind       | INQUIRE TEXT COLOUR INDEX                            |
| ginq_text_extent         | INQUIRE TEXT EXTENT                                  |
| ginq_text_fac            | INQUIRE TEXT FACILITIES                              |
| ginq_text_font_prec      | INQUIRE TEXT FONT AND PRECISION                      |
| ginq_text_ind            | INQUIRE TEXT INDEX                                   |
| ginq_text_path           | INQUIRE TEXT PATH                                    |
| ginq_text_rep            | INQUIRE TEXT REPRESENTATION                          |

## Function Names

## Tables

|                       |                                                |
|-----------------------|------------------------------------------------|
| ginq_val_st           | INQUIRE VALUATOR DEVICE STATE                  |
| ginq_ws_cat           | INQUIRE WORKSTATION CATEGORY                   |
| ginq_ws_class         | INQUIRE WORKSTATION CLASSIFICATION             |
| ginq_ws_conn_type     | INQUIRE WORKSTATION CONNECTION AND TYPE        |
| ginq_ws_defer_upd_sts | INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES |
| ginq_ws_max_nums      | INQUIRE WORKSTATION MAXIMUM NUMBERS            |
| ginq_ws_st            | INQUIRE WORKSTATION STATE                      |
| ginq_ws_tran          | INQUIRE WORKSTATION TRANSFORMATION             |
| ginsert_seg           | INSERT SEGMENT                                 |
| ginterpret_item       | INTERPRET ITEM                                 |
| gmessage              | MESSAGE                                        |
| gopen_gks             | OPEN GKS                                       |
| gopen_ws              | OPEN WORKSTATION                               |
| gpolyline             | POLYLINE                                       |
| gpolymarker           | POLYMARKER                                     |
| gread_item            | READ ITEM FROM GKSM                            |
| gredraw_all_segs_ws   | REDRAW ALL SEGMENTS ON WORKSTATION             |
| grename_seg           | RENAME SEGMENT                                 |
| greq_choice           | REQUEST CHOICE                                 |
| greq_loc              | REQUEST LOCATOR                                |
| greq_pick             | REQUEST PICK                                   |
| greq_string           | REQUEST STRING                                 |
| greq_stroke           | REQUEST STROKE                                 |
| greq_val              | REQUEST VALUATOR                               |
| gsample_choice        | SAMPLE CHOICE                                  |
| gsample_loc           | SAMPLE LOCATOR                                 |
| gsample_pick          | SAMPLE PICK                                    |
| gsample_string        | SAMPLE STRING                                  |
| gsample_stroke        | SAMPLE STROKE                                  |
| gsample_val           | SAMPLE VALUATOR                                |
| gsel_norm_tran        | SELECT NORMALIZATION TRANSFORMATION            |
| gset_asfs             | SET ASPECT SOURCE FLAGS                        |
| gset_char_expan       | SET CHARACTER EXPANSION FACTOR                 |
| gset_char_ht          | SET CHARACTER HEIGHT                           |
| gset_char_space       | SET CHARACTER SPACING                          |
| gset_char_up_vec      | SET CHARACTER UP VECTOR                        |
| gset_choice_mode      | SET CHOICE MODE                                |
| gset_clip_ind         | SET CLIPPING INDICATOR                         |
| gset_colr_rep         | SET COLOUR REPRESENTATION                      |
| gset_defer_st         | SET DEFERRAL STATE                             |
| gset_det              | SET DETECTABILITY                              |
| gset_err_hand         | SET ERROR HANDLER (GKS/C only)                 |
| gset_fill_colr_ind    | SET FILL AREA COLOUR INDEX                     |
| gset_fill_ind         | SET FILL AREA INDEX                            |
| gset_fill_int_style   | SET FILL AREA INTERIOR STYLE                   |
| gset_fill_rep         | SET FILL AREA REPRESENTATION                   |
| gset_fill_style_ind   | SET FILL AREA STYLE INDEX                      |
| gset_highl            | SET HIGHLIGHTING                               |
| gset_line_colr_ind    | SET POLYLINE COLOUR INDEX                      |
| gset_line_ind         | SET POLYLINE INDEX                             |
| gset_line_rep         | SET POLYLINE REPRESENTATION                    |
| gset_linetype         | SET LINETYPE                                   |

## Tables

## Function Names

|                      |                               |
|----------------------|-------------------------------|
| gset_linewidth       | SET LINEWIDTH SCALE FACTOR    |
| gset_loc_mode        | SET LOCATOR MODE              |
| gset_marker_colr_ind | SET POLYMARKER COLOUR INDEX   |
| gset_marker_ind      | SET POLYMARKER INDEX          |
| gset_marker_rep      | SET POLYMARKER REPRESENTATION |
| gset_marker_size     | SET MARKER SIZE SCALE FACTOR  |
| gset_marker_type     | SET MARKER TYPE               |
| gset_pat_ref_point   | SET PATTERN REFERENCE POINT   |
| gset_pat_rep         | SET PATTERN REPRESENTATION    |
| gset_pat_size        | SET PATTERN SIZE              |
| gset_pick_id         | SET PICK IDENTIFIER           |
| gset_pick_mode       | SET PICK MODE                 |
| gset_seg_pri         | SET SEGMENT PRIORITY          |
| gset_seg_tran        | SET SEGMENT TRANSFORMATION    |
| gset_string_mode     | SET STRING MODE               |
| gset_stroke_mode     | SET STROKE MODE               |
| gset_text_align      | SET TEXT ALIGNMENT            |
| gset_text_colr_ind   | SET TEXT COLOUR INDEX         |
| gset_text_font_prec  | SET TEXT FONT AND PRECISION   |
| gset_text_ind        | SET TEXT INDEX                |
| gset_text_path       | SET TEXT PATH                 |
| gset_text_rep        | SET TEXT REPRESENTATION       |
| gset_val_mode        | SET VALUATOR MODE             |
| gset_vis             | SET VISIBILITY                |
| gset_vp              | SET VIEWPORT                  |
| gset_vp_in_pri       | SET VIEWPORT INPUT PRIORITY   |
| gset_win             | SET WINDOW                    |
| gset_ws_vp           | SET WORKSTATION VIEWPORT      |
| gset_ws_win          | SET WORKSTATION WINDOW        |
| gtext                | TEXT                          |
| gupd_ws              | UPDATE WORKSTATION            |
| gwrite_item          | WRITE ITEM TO GKSM            |

## 4.3.2 List Ordered Alphabetically by GKS Name

|                                    |                      |
|------------------------------------|----------------------|
| ACCUMULATE TRANSFORMATION MATRIX   | gaccum_tran_matrix   |
| ACTIVATE WORKSTATION               | gactivate_ws         |
| ASSOCIATE SEGMENT WITH WORKSTATION | gassoc_seg_ws        |
| AWAIT EVENT                        | gawait_event         |
| CELL ARRAY                         | gcell_array          |
| CLEAR WORKSTATION                  | gclear_ws            |
| CLOSE GKS                          | gclose_gks           |
| CLOSE SEGMENT                      | gclose_seg           |
| CLOSE WORKSTATION                  | gclose_ws            |
| COPY SEGMENT FROM WORKSTATION      | gcopy_seg_ws         |
| CREATE SEGMENT                     | gcreate_seg          |
| CREATE STORE (GKS/C only)          | gcreate_store        |
| DEACTIVATE WORKSTATION             | gdeactivate_ws       |
| DELETE SEGMENT                     | gdel_seg             |
| DELETE SEGMENT FROM WORKSTATION    | gdel_seg_ws          |
| DELETE STORE (GKS/C only)          | gdel_store           |
| EMERGENCY CLOSE GKS                | gemergency_close_gks |

## Function Names

## Tables

|                                                        |                                     |
|--------------------------------------------------------|-------------------------------------|
| ERROR HANDLING                                         | <code>gerr_hand</code>              |
| ERROR LOGGING                                          | <code>gerr_log</code>               |
| ESCAPE                                                 | <code>gescape</code>                |
| EVALUATE TRANSFORMATION MATRIX                         | <code>geval_tran_matrix</code>      |
| FILL AREA                                              | <code>gfill_area</code>             |
| FLUSH DEVICE EVENTS                                    | <code>gflush_events</code>          |
| GENERALIZED DRAWING PRIMITIVE                          | <code>ggdp</code>                   |
| GET CHOICE                                             | <code>gget_choice</code>            |
| GET ITEM TYPE FROM GKSM                                | <code>gget_item_type</code>         |
| GET LOCATOR                                            | <code>gget_loc</code>               |
| GET PICK                                               | <code>gget_pick</code>              |
| GET STRING                                             | <code>gget_string</code>            |
| GET STROKE                                             | <code>gget_stroke</code>            |
| GET VALUATOR                                           | <code>gget_val</code>               |
| INITIALISE CHOICE                                      | <code>ginit_choice</code>           |
| INITIALISE LOCATOR                                     | <code>ginit_loc</code>              |
| INITIALISE PICK                                        | <code>ginit_pick</code>             |
| INITIALISE STRING                                      | <code>ginit_string</code>           |
| INITIALISE STROKE                                      | <code>ginit_stroke</code>           |
| INITIALISE VALUATOR                                    | <code>ginit_val</code>              |
| INQUIRE ASPECT SOURCE FLAGS                            | <code>ginq_asfs</code>              |
| INQUIRE CHARACTER BASE VECTOR                          | <code>ginq_char_base_vec</code>     |
| INQUIRE CHARACTER EXPANSION FACTOR                     | <code>ginq_char_expan</code>        |
| INQUIRE CHARACTER HEIGHT                               | <code>ginq_char_ht</code>           |
| INQUIRE CHARACTER SPACING                              | <code>ginq_char_space</code>        |
| INQUIRE CHARACTER UP VECTOR                            | <code>ginq_char_up_vec</code>       |
| INQUIRE CHARACTER WIDTH                                | <code>ginq_char_width</code>        |
| INQUIRE CHOICE DEVICE STATE                            | <code>ginq_choice_st</code>         |
| INQUIRE CLIPPING                                       | <code>ginq_clip</code>              |
| INQUIRE COLOUR FACILITIES                              | <code>ginq_colr_fac</code>          |
| INQUIRE COLOUR REPRESENTATION                          | <code>ginq_colr_rep</code>          |
| INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES            | <code>ginq_cur_indiv_attr</code>    |
| INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER    | <code>ginq_cur_norm_tran_num</code> |
| INQUIRE CURRENT PICK IDENTIFIER VALUE                  | <code>ginq_cur_pick_id</code>       |
| INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES             | <code>ginq_cur_prim_attr</code>     |
| INQUIRE DEFAULT CHOICE DEVICE DATA                     | <code>ginq_def_choice_data</code>   |
| INQUIRE DEFAULT DEFERRAL STATE VALUES                  | <code>ginq_def_defer_sts</code>     |
| INQUIRE DEFAULT LOCATOR DEVICE DATA                    | <code>ginq_def_loc_data</code>      |
| INQUIRE DEFAULT PICK DEVICE DATA                       | <code>ginq_def_pick_data</code>     |
| INQUIRE DEFAULT STRING DEVICE DATA                     | <code>ginq_def_string_data</code>   |
| INQUIRE DEFAULT STROKE DEVICE DATA                     | <code>ginq_def_stroke_data</code>   |
| INQUIRE DEFAULT VALUATOR DEVICE DATA                   | <code>ginq_def_val_data</code>      |
| INQUIRE DISPLAY SPACE SIZE                             | <code>ginq_disp_space_size</code>   |
| INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES     | <code>ginq_dyn_mod_seg_attr</code>  |
| INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES | <code>ginq_dyn_mod_ws_attr</code>   |
| INQUIRE FILL AREA COLOUR INDEX                         | <code>ginq_fill_colr_ind</code>     |
| INQUIRE FILL AREA FACILITIES                           | <code>ginq_fill_fac</code>          |
| INQUIRE FILL AREA INDEX                                | <code>ginq_fill_ind</code>          |
| INQUIRE FILL AREA INTERIOR STYLE                       | <code>ginq_fill_int_style</code>    |
| INQUIRE FILL AREA REPRESENTATION                       | <code>ginq_fill_rep</code>          |
| INQUIRE FILL AREA STYLE INDEX                          | <code>ginq_fill_style_ind</code>    |
| INQUIRE GENERALIZED DRAWING PRIMITIVE                  | <code>ginq_gdp</code>               |

## Tables

## Function Names

|                                                          |                          |
|----------------------------------------------------------|--------------------------|
| INQUIRE INPUT QUEUE OVERFLOW                             | ginq_in_overf            |
| INQUIRE LEVEL OF GKS                                     | ginq_level_gks           |
| INQUIRE LINETYPE                                         | ginq_linetype            |
| INQUIRE LINEWIDTH SCALE FACTOR                           | ginq_linewidth           |
| INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES | ginq_list_avail_gdps     |
| INQUIRE LIST OF AVAILABLE WORKSTATION TYPES              | ginq_list_avail_ws_types |
| INQUIRE LIST OF COLOUR INDICES                           | ginq_list_colr_inds      |
| INQUIRE LIST OF FILL AREA INDICES                        | ginq_list_fill_inds      |
| INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS     | ginq_list_norm_tran_nums |
| INQUIRE LIST OF PATTERN INDICES                          | ginq_list_pat_inds       |
| INQUIRE LIST OF POLYLINE INDICES                         | ginq_list_line_inds      |
| INQUIRE LIST OF POLYMARKER INDICES                       | ginq_list_marker_inds    |
| INQUIRE LIST OF TEXT INDICES                             | ginq_list_text_inds      |
| INQUIRE LOCATOR DEVICE STATE                             | ginq_loc_st              |
| INQUIRE MARKER SIZE SCALE FACTOR                         | ginq_marker_size         |
| INQUIRE MARKER TYPE                                      | ginq_marker_type         |
| INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES       | ginq_max_ws_st_tables    |
| INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER      | ginq_max_norm_tran_num   |
| INQUIRE MORE SIMULTANEOUS EVENTS                         | ginq_more_simult_events  |
| INQUIRE NAME OF OPEN SEGMENT                             | ginq_name_open_seg       |
| INQUIRE NORMALIZATION TRANSFORMATION                     | ginq_norm_tran           |
| INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES        | ginq_num_avail_in        |
| INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED           | ginq_num_seg_pris        |
| INQUIRE OPERATING STATE VALUE                            | ginq_op_st               |
| INQUIRE PATTERN FACILITIES                               | ginq_pat_fac             |
| INQUIRE PATTERN HEIGHT VECTOR                            | ginq_pat_ht_vec          |
| INQUIRE PATTERN REFERENCE POINT                          | ginq_pat_ref_point       |
| INQUIRE PATTERN REPRESENTATION                           | ginq_pat_rep             |
| INQUIRE PATTERN WIDTH VECTOR                             | ginq_pat_width_vec       |
| INQUIRE PICK DEVICE STATE                                | ginq_pick_st             |
| INQUIRE PIXEL                                            | ginq_pixel               |
| INQUIRE PIXEL ARRAY                                      | ginq_pixel_array         |
| INQUIRE PIXEL ARRAY DIMENSIONS                           | ginq_pixel_array_dims    |
| INQUIRE POLYLINE COLOUR INDEX                            | ginq_line_colr_ind       |
| INQUIRE POLYLINE FACILITIES                              | ginq_line_fac            |
| INQUIRE POLYLINE INDEX                                   | ginq_line_ind            |
| INQUIRE POLYLINE REPRESENTATION                          | ginq_line_rep            |
| INQUIRE POLYMARKER COLOUR INDEX                          | ginq_marker_colr_ind     |
| INQUIRE POLYMARKER FACILITIES                            | ginq_marker_fac          |
| INQUIRE POLYMARKER INDEX                                 | ginq_marker_ind          |
| INQUIRE POLYMARKER REPRESENTATION                        | ginq_marker_rep          |
| INQUIRE PREDEFINED COLOUR REPRESENTATION                 | ginq_pred_colr_rep       |
| INQUIRE PREDEFINED FILL AREA REPRESENTATION              | ginq_pred_fill_rep       |
| INQUIRE PREDEFINED PATTERN REPRESENTATION                | ginq_pred_pat_rep        |
| INQUIRE PREDEFINED POLYLINE REPRESENTATION               | ginq_pred_line_rep       |
| INQUIRE PREDEFINED POLYMARKER REPRESENTATION             | ginq_pred_marker_rep     |
| INQUIRE PREDEFINED TEXT REPRESENTATION                   | ginq_pred_text_rep       |
| INQUIRE SEGMENT ATTRIBUTES                               | ginq_seg_attrs           |
| INQUIRE SET OF ACTIVE WORKSTATIONS                       | ginq_set_active_wss      |
| INQUIRE SET OF ASSOCIATED WORKSTATIONS                   | ginq_set_assoc_wss       |
| INQUIRE SET OF OPEN WORKSTATIONS                         | ginq_set_open_wss        |
| INQUIRE SET OF SEGMENT NAMES IN USE                      | ginq_set_seg_names       |

## Function Names

## Tables

INQUIRE SET OF SEGMENT NAMES ON WORKSTATION  
 INQUIRE STRING DEVICE STATE  
 INQUIRE STROKE DEVICE STATE  
 INQUIRE TEXT ALIGNMENT  
 INQUIRE TEXT COLOUR INDEX  
 INQUIRE TEXT EXTENT  
 INQUIRE TEXT FACILITIES  
 INQUIRE TEXT FONT AND PRECISION  
 INQUIRE TEXT INDEX  
 INQUIRE TEXT PATH  
 INQUIRE TEXT REPRESENTATION  
 INQUIRE VALUATOR DEVICE STATE  
 INQUIRE WORKSTATION CATEGORY  
 INQUIRE WORKSTATION CLASSIFICATION  
 INQUIRE WORKSTATION CONNECTION AND TYPE  
 INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES  
 INQUIRE WORKSTATION MAXIMUM NUMBERS  
 INQUIRE WORKSTATION STATE  
 INQUIRE WORKSTATION TRANSFORMATION  
 INSERT SEGMENT  
 INTERPRET ITEM  
 MESSAGE  
 OPEN GKS  
 OPEN WORKSTATION  
 POLYLINE  
 POLYMARKER  
 READ ITEM FROM GKSM  
 REDRAW ALL SEGMENTS ON WORKSTATION  
 RENAME SEGMENT  
 REQUEST CHOICE  
 REQUEST LOCATOR  
 REQUEST PICK  
 REQUEST STRING  
 REQUEST STROKE  
 REQUEST VALUATOR  
 SAMPLE CHOICE  
 SAMPLE LOCATOR  
 SAMPLE PICK  
 SAMPLE STRING  
 SAMPLE STROKE  
 SAMPLE VALUATOR  
 SELECT NORMALIZATION TRANSFORMATION  
 SET ASPECT SOURCE FLAGS  
 SET CHARACTER EXPANSION FACTOR  
 SET CHARACTER HEIGHT  
 SET CHARACTER SPACING  
 SET CHARACTER UP VECTOR  
 SET CHOICE MODE  
 SET CLIPPING INDICATOR  
 SET COLOUR REPRESENTATION  
 SET DEFERRAL STATE  
 SET DETECTABILITY

ginq\_set\_seg\_names\_ws  
 ginq\_string\_st  
 ginq\_stroke\_st  
 ginq\_text\_align  
 ginq\_text\_colr\_ind  
 ginq\_text\_extent  
 ginq\_text\_fac  
 ginq\_text\_font\_prec  
 ginq\_text\_ind  
 ginq\_text\_path  
 ginq\_text\_rep  
 ginq\_val\_st  
 ginq\_ws\_cat  
 ginq\_ws\_class  
 ginq\_ws\_conn\_type  
 ginq\_ws\_defer\_upd\_sts  
 ginq\_ws\_max\_nums  
 ginq\_ws\_st  
 ginq\_ws\_tran  
 ginsert\_seg  
 ginterpret\_item  
 gmessage  
 gopen\_gks  
 gopen\_ws  
 gpolyline  
 gpolymarker  
 gread\_item  
 gredraw\_all\_segs\_ws  
 grename\_seg  
 greq\_choice  
 greq\_loc  
 greq\_pick  
 greq\_string  
 greq\_stroke  
 greq\_val  
 gsample\_choice  
 gsample\_loc  
 gsample\_pick  
 gsample\_string  
 gsample\_stroke  
 gsample\_val  
 gsel\_norm\_tran  
 gset\_asfs  
 gset\_char\_expan  
 gset\_char\_ht  
 gset\_char\_space  
 gset\_char\_up\_vec  
 gset\_choice\_mode  
 gset\_clip\_ind  
 gset\_colr\_rep  
 gset\_defer\_st  
 gset\_det

## Tables

## Function Names

SET ERROR HANDLER (GKS/C only)  
 SET FILL AREA COLOUR INDEX  
 SET FILL AREA INDEX  
 SET FILL AREA INTERIOR STYLE  
 SET FILL AREA REPRESENTATION  
 SET FILL AREA STYLE INDEX  
 SET HIGHLIGHTING  
 SET LINETYPE  
 SET LINEWIDTH SCALE FACTOR  
 SET LOCATOR MODE  
 SET MARKER SIZE SCALE FACTOR  
 SET MARKER TYPE  
 SET PATTERN REFERENCE POINT  
 SET PATTERN REPRESENTATION  
 SET PATTERN SIZE  
 SET PICK IDENTIFIER  
 SET PICK MODE  
 SET POLYLINE COLOUR INDEX  
 SET POLYLINE INDEX  
 SET POLYLINE REPRESENTATION  
 SET POLYMARKER COLOUR INDEX  
 SET POLYMARKER INDEX  
 SET POLYMARKER REPRESENTATION  
 SET SEGMENT PRIORITY  
 SET SEGMENT TRANSFORMATION  
 SET STRING MODE  
 SET STROKE MODE  
 SET TEXT ALIGNMENT  
 SET TEXT COLOUR INDEX  
 SET TEXT FONT AND PRECISION  
 SET TEXT INDEX  
 SET TEXT PATH  
 SET TEXT REPRESENTATION  
 SET VALUATOR MODE  
 SET VIEWPORT  
 SET VIEWPORT INPUT PRIORITY  
 SET VISIBILITY  
 SET WINDOW  
 SET WORKSTATION VIEWPORT  
 SET WORKSTATION WINDOW  
 TEXT  
 UPDATE WORKSTATION  
 WRITE ITEM TO GKSM

gset\_err\_hand  
 gset\_fill\_colr\_ind  
 gset\_fill\_ind  
 gset\_fill\_int\_style  
 gset\_fill\_rep  
 gset\_fill\_style\_ind  
 gset\_highl  
 gset\_linetype  
 gset\_linewidth  
 gset\_loc\_mode  
 gset\_marker\_size  
 gset\_marker\_type  
 gset\_pat\_ref\_point  
 gset\_pat\_rep  
 gset\_pat\_size  
 gset\_pick\_id  
 gset\_pick\_mode  
 gset\_line\_colr\_ind  
 gset\_line\_ind  
 gset\_line\_rep  
 gset\_marker\_colr\_ind  
 gset\_marker\_ind  
 gset\_marker\_rep  
 gset\_seg\_pri  
 gset\_seg\_tran  
 gset\_string\_mode  
 gset\_stroke\_mode  
 gset\_text\_align  
 gset\_text\_colr\_ind  
 gset\_text\_font\_prec  
 gset\_text\_ind  
 gset\_text\_path  
 gset\_text\_rep  
 gset\_val\_mode  
 gset\_vp  
 gset\_vp\_in\_pri  
 gset\_vis  
 gset\_win  
 gset\_ws\_vp  
 gset\_ws\_win  
 gtext  
 gupd\_ws  
 gwrite\_item

### 4.3.3 List Ordered Alphabetically by Bound Name within Level

#### 4.3.3.1 Level 0A

|              |                      |
|--------------|----------------------|
| gactivate_ws | ACTIVATE WORKSTATION |
| gcell_array  | CELL ARRAY           |
| gclear_ws    | CLEAR WORKSTATION    |
| gclose_gks   | CLOSE GKS            |

## Function Names

## Tables

|                          |                                                          |
|--------------------------|----------------------------------------------------------|
| gclose_ws                | CLOSE WORKSTATION                                        |
| gcreate_store            | CREATE STORE (GKS/C only)                                |
| gdeactivate_ws           | DEACTIVATE WORKSTATION                                   |
| gdel_store               | DELETE STORE (GKS/C only)                                |
| gemergency_close_gks     | EMERGENCY CLOSE GKS                                      |
| gerr_hand                | ERROR HANDLING                                           |
| gerr_log                 | ERROR LOGGING                                            |
| gescape                  | ESCAPE                                                   |
| gfill_area               | FILL AREA                                                |
| ggdp                     | GENERALIZED DRAWING PRIMITIVE                            |
| gget_item_type           | GET ITEM TYPE FROM GKSM                                  |
| ginq_asfs                | INQUIRE ASPECT SOURCE FLAGS                              |
| ginq_char_base_vec       | INQUIRE CHARACTER BASE VECTOR                            |
| ginq_char_expan          | INQUIRE CHARACTER EXPANSION FACTOR                       |
| ginq_char_ht             | INQUIRE CHARACTER HEIGHT                                 |
| ginq_char_space          | INQUIRE CHARACTER SPACING                                |
| ginq_char_up_vec         | INQUIRE CHARACTER UP VECTOR                              |
| ginq_char_width          | INQUIRE CHARACTER WIDTH                                  |
| ginq_clip                | INQUIRE CLIPPING                                         |
| ginq_colr_fac            | INQUIRE COLOUR FACILITIES                                |
| ginq_colr_rep            | INQUIRE COLOUR REPRESENTATION                            |
| ginq_cur_indiv_attr      | INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES              |
| ginq_cur_norm_tran_num   | INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER      |
| ginq_cur_prim_attr       | INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES               |
| ginq_disp_space_size     | INQUIRE DISPLAY SPACE SIZE                               |
| ginq_dyn_mod_ws_attr     | INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES   |
| ginq_fill_colr_ind       | INQUIRE FILL AREA COLOUR INDEX                           |
| ginq_fill_fac            | INQUIRE FILL AREA FACILITIES                             |
| ginq_fill_ind            | INQUIRE FILL AREA INDEX                                  |
| ginq_fill_int_style      | INQUIRE FILL AREA INTERIOR STYLE                         |
| ginq_fill_style_ind      | INQUIRE FILL AREA STYLE INDEX                            |
| ginq_gdp                 | INQUIRE GENERALIZED DRAWING PRIMITIVE                    |
| ginq_level_gks           | INQUIRE LEVEL OF GKS                                     |
| ginq_linetype            | INQUIRE LINETYPE                                         |
| ginq_linewidth           | INQUIRE LINEWIDTH SCALE FACTOR                           |
| ginq_list_avail_gdps     | INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES |
| ginq_list_avail_ws_types | INQUIRE LIST OF AVAILABLE WORKSTATION TYPES              |
| ginq_list_colr_inds      | INQUIRE LIST OF COLOUR INDICES                           |
| ginq_list_norm_tran_nums | INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS     |
| ginq_marker_size         | INQUIRE MARKER SIZE SCALE FACTOR                         |
| ginq_marker_type         | INQUIRE MARKER TYPE                                      |
| ginq_max_ws_st_tables    | INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES       |
| ginq_max_norm_tran_num   | INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER      |
| ginq_norm_tran           | INQUIRE NORMALIZATION TRANSFORMATION                     |
| ginq_op_st               | INQUIRE OPERATING STATE VALUE                            |
| ginq_pat_fac             | INQUIRE PATTERN FACILITIES                               |
| ginq_pat_ht_vec          | INQUIRE PATTERN HEIGHT VECTOR                            |
| ginq_pat_ref_point       | INQUIRE PATTERN REFERENCE POINT                          |
| ginq_pat_width_vec       | INQUIRE PATTERN WIDTH VECTOR                             |
| ginq_pixel               | INQUIRE PIXEL                                            |
| ginq_pixel_array         | INQUIRE PIXEL ARRAY                                      |
| ginq_pixel_array_dims    | INQUIRE PIXEL ARRAY DIMENSIONS                           |

## Tables

## Function Names

|                       |                                                |
|-----------------------|------------------------------------------------|
| ginq_line_colr_ind    | INQUIRE POLYLINE COLOUR INDEX                  |
| ginq_line_fac         | INQUIRE POLYLINE FACILITIES                    |
| ginq_line_ind         | INQUIRE POLYLINE INDEX                         |
| ginq_marker_colr_ind  | INQUIRE POLYMARKER COLOUR INDEX                |
| ginq_marker_fac       | INQUIRE POLYMARKER FACILITIES                  |
| ginq_marker_ind       | INQUIRE POLYMARKER INDEX                       |
| ginq_pred_colr_rep    | INQUIRE PREDEFINED COLOUR REPRESENTATION       |
| ginq_pred_fill_rep    | INQUIRE PREDEFINED FILL AREA REPRESENTATION    |
| ginq_pred_pat_rep     | INQUIRE PREDEFINED PATTERN REPRESENTATION      |
| ginq_pred_line_rep    | INQUIRE PREDEFINED POLYLINE REPRESENTATION     |
| ginq_pred_marker_rep  | INQUIRE PREDEFINED POLYMARKER REPRESENTATION   |
| ginq_pred_text_rep    | INQUIRE PREDEFINED TEXT REPRESENTATION         |
| ginq_set_active_wss   | INQUIRE SET OF ACTIVE WORKSTATIONS             |
| ginq_set_open_wss     | INQUIRE SET OF OPEN WORKSTATIONS               |
| ginq_text_align       | INQUIRE TEXT ALIGNMENT                         |
| ginq_text_colr_ind    | INQUIRE TEXT COLOUR INDEX                      |
| ginq_text_extent      | INQUIRE TEXT EXTENT                            |
| ginq_text_fac         | INQUIRE TEXT FACILITIES                        |
| ginq_text_font_prec   | INQUIRE TEXT FONT AND PRECISION                |
| ginq_text_ind         | INQUIRE TEXT INDEX                             |
| ginq_text_path        | INQUIRE TEXT PATH                              |
| ginq_ws_cat           | INQUIRE WORKSTATION CATEGORY                   |
| ginq_ws_class         | INQUIRE WORKSTATION CLASSIFICATION             |
| ginq_ws_conn_type     | INQUIRE WORKSTATION CONNECTION AND TYPE        |
| ginq_ws_defer_upd_sts | INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES |
| ginq_ws_max_nums      | INQUIRE WORKSTATION MAXIMUM NUMBERS            |
| ginq_ws_st            | INQUIRE WORKSTATION STATE                      |
| ginq_ws_tran          | INQUIRE WORKSTATION TRANSFORMATION             |
| ginterpret_item       | INTERPRET ITEM                                 |
| gopen_gks             | OPEN GKS                                       |
| gopen_ws              | OPEN WORKSTATION                               |
| gpolyline             | POLYLINE                                       |
| gpolymarker           | POLYMARKER                                     |
| gread_item            | READ ITEM FROM GKSM                            |
| gsel_norm_tran        | SELECT NORMALIZATION TRANSFORMATION            |
| gset_asfs             | SET ASPECT SOURCE FLAGS                        |
| gset_char_expan       | SET CHARACTER EXPANSION FACTOR                 |
| gset_char_ht          | SET CHARACTER HEIGHT                           |
| gset_char_space       | SET CHARACTER SPACING                          |
| gset_char_up_vec      | SET CHARACTER UP VECTOR                        |
| gset_clip_ind         | SET CLIPPING INDICATOR                         |
| gset_colr_rep         | SET COLOUR REPRESENTATION                      |
| gset_err_hand         | SET ERROR HANDLER (GKS/C only)                 |
| gset_fill_colr_ind    | SET FILL AREA COLOUR INDEX                     |
| gset_fill_ind         | SET FILL AREA INDEX                            |
| gset_fill_int_style   | SET FILL AREA INTERIOR STYLE                   |
| gset_fill_style_ind   | SET FILL AREA STYLE INDEX                      |
| gset_linetype         | SET LINETYPE                                   |
| gset_linewidth        | SET LINEWIDTH SCALE FACTOR                     |
| gset_marker_size      | SET MARKER SIZE SCALE FACTOR                   |
| gset_marker_type      | SET MARKER TYPE                                |
| gset_pat_ref_point    | SET PATTERN REFERENCE POINT                    |

## Function Names

## Tables

|                                   |                             |
|-----------------------------------|-----------------------------|
| <code>gset_pat_size</code>        | SET PATTERN SIZE            |
| <code>gset_line_colr_ind</code>   | SET POLYLINE COLOUR INDEX   |
| <code>gset_line_ind</code>        | SET POLYLINE INDEX          |
| <code>gset_marker_colr_ind</code> | SET POLYMARKER COLOUR INDEX |
| <code>gset_marker_ind</code>      | SET POLYMARKER INDEX        |
| <code>gset_text_align</code>      | SET TEXT ALIGNMENT          |
| <code>gset_text_colr_ind</code>   | SET TEXT COLOUR INDEX       |
| <code>gset_text_font_prec</code>  | SET TEXT FONT AND PRECISION |
| <code>gset_text_ind</code>        | SET TEXT INDEX              |
| <code>gset_text_path</code>       | SET TEXT PATH               |
| <code>gset_vp</code>              | SET VIEWPORT                |
| <code>gset_win</code>             | SET WINDOW                  |
| <code>gset_ws_vp</code>           | SET WORKSTATION VIEWPORT    |
| <code>gset_ws_win</code>          | SET WORKSTATION WINDOW      |
| <code>gtext</code>                | TEXT                        |
| <code>gupd_ws</code>              | UPDATE WORKSTATION          |
| <code>gwrite_item</code>          | WRITE ITEM TO GKSM          |

## 4.3.3.2 Level 0B

|                                   |                                                   |
|-----------------------------------|---------------------------------------------------|
| <code>ginit_choice</code>         | INITIALISE CHOICE                                 |
| <code>ginit_loc</code>            | INITIALISE LOCATOR                                |
| <code>ginit_string</code>         | INITIALISE STRING                                 |
| <code>ginit_stroke</code>         | INITIALISE STROKE                                 |
| <code>ginit_val</code>            | INITIALISE VALUATOR                               |
| <code>ginq_choice_st</code>       | INQUIRE CHOICE DEVICE STATE                       |
| <code>ginq_def_choice_data</code> | INQUIRE DEFAULT CHOICE DEVICE DATA                |
| <code>ginq_def_loc_data</code>    | INQUIRE DEFAULT LOCATOR DEVICE DATA               |
| <code>ginq_def_string_data</code> | INQUIRE DEFAULT STRING DEVICE DATA                |
| <code>ginq_def_stroke_data</code> | INQUIRE DEFAULT STROKE DEVICE DATA                |
| <code>ginq_def_val_data</code>    | INQUIRE DEFAULT VALUATOR DEVICE DATA              |
| <code>ginq_loc_st</code>          | INQUIRE LOCATOR DEVICE STATE                      |
| <code>ginq_num_avail_in</code>    | INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES |
| <code>ginq_string_st</code>       | INQUIRE STRING DEVICE STATE                       |
| <code>ginq_stroke_st</code>       | INQUIRE STROKE DEVICE STATE                       |
| <code>ginq_val_st</code>          | INQUIRE VALUATOR DEVICE STATE                     |
| <code>greq_choice</code>          | REQUEST CHOICE                                    |
| <code>greq_loc</code>             | REQUEST LOCATOR                                   |
| <code>greq_string</code>          | REQUEST STRING                                    |
| <code>greq_stroke</code>          | REQUEST STROKE                                    |
| <code>greq_val</code>             | REQUEST VALUATOR                                  |
| <code>gset_choice_mode</code>     | SET CHOICE MODE                                   |
| <code>gset_loc_mode</code>        | SET LOCATOR MODE                                  |
| <code>gset_string_mode</code>     | SET STRING MODE                                   |
| <code>gset_stroke_mode</code>     | SET STROKE MODE                                   |
| <code>gset_val_mode</code>        | SET VALUATOR MODE                                 |
| <code>gset_vp_in_pri</code>       | SET VIEWPORT INPUT PRIORITY                       |

## Tables

## Function Names

## 4.3.3.3 Level 0C

|                         |                                  |
|-------------------------|----------------------------------|
| gawait_event            | AWAIT EVENT                      |
| gflush_events           | FLUSH DEVICE EVENTS              |
| gget_choice             | GET CHOICE                       |
| gget_loc                | GET LOCATOR                      |
| gget_string             | GET STRING                       |
| gget_stroke             | GET STROKE                       |
| gget_val                | GET VALUATOR                     |
| ginq_in_overf           | INQUIRE INPUT QUEUE OVERFLOW     |
| ginq_more_simult_events | INQUIRE MORE SIMULTANEOUS EVENTS |
| gsample_choice          | SAMPLE CHOICE                    |
| gsample_loc             | SAMPLE LOCATOR                   |
| gsample_string          | SAMPLE STRING                    |
| gsample_stroke          | SAMPLE STROKE                    |
| gsample_val             | SAMPLE VALUATOR                  |

## 4.3.3.4 Level 1A

|                        |                                                    |
|------------------------|----------------------------------------------------|
| gaccum_tran_matrix     | ACCUMULATE TRANSFORMATION MATRIX                   |
| gclose_seg             | CLOSE SEGMENT                                      |
| gcreate_seg            | CREATE SEGMENT                                     |
| gdel_seg               | DELETE SEGMENT                                     |
| gdel_seg_ws            | DELETE SEGMENT FROM WORKSTATION                    |
| geval_tran_matrix      | EVALUATE TRANSFORMATION MATRIX                     |
| ginq_def_defer_sts     | INQUIRE DEFAULT DEFERRAL STATE VALUES              |
| ginq_dyn_mod_seg_attrs | INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES |
| ginq_fill_rep          | INQUIRE FILL AREA REPRESENTATION                   |
| ginq_list_fill_inds    | INQUIRE LIST OF FILL AREA INDICES                  |
| ginq_list_pat_inds     | INQUIRE LIST OF PATTERN INDICES                    |
| ginq_list_line_inds    | INQUIRE LIST OF POLYLINE INDICES                   |
| ginq_list_marker_inds  | INQUIRE LIST OF POLYMARKER INDICES                 |
| ginq_list_text_inds    | INQUIRE LIST OF TEXT INDICES                       |
| ginq_name_open_seg     | INQUIRE NAME OF OPEN SEGMENT                       |
| ginq_num_seg_pris      | INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED     |
| ginq_pat_rep           | INQUIRE PATTERN REPRESENTATION                     |
| ginq_line_rep          | INQUIRE POLYLINE REPRESENTATION                    |
| ginq_marker_rep        | INQUIRE POLYMARKER REPRESENTATION                  |
| ginq_seg_attrs         | INQUIRE SEGMENT ATTRIBUTES                         |
| ginq_set_assoc_wss     | INQUIRE SET OF ASSOCIATED WORKSTATIONS             |
| ginq_set_seg_names     | INQUIRE SET OF SEGMENT NAMES IN USE                |
| ginq_set_seg_names_ws  | INQUIRE SET OF SEGMENT NAMES ON WORKSTATION        |
| ginq_text_rep          | INQUIRE TEXT REPRESENTATION                        |
| gmessage               | MESSAGE                                            |
| gredraw_all_segs_ws    | REDRAW ALL SEGMENTS ON WORKSTATION                 |
| grename_seg            | RENAME SEGMENT                                     |
| gset_defer_st          | SET DEFERRAL STATE                                 |
| gset_fill_rep          | SET FILL AREA REPRESENTATION                       |
| gset_highl             | SET HIGHLIGHTING                                   |
| gset_pat_rep           | SET PATTERN REPRESENTATION                         |
| gset_line_rep          | SET POLYLINE REPRESENTATION                        |
| gset_marker_rep        | SET POLYMARKER REPRESENTATION                      |
| gset_seg_pri           | SET SEGMENT PRIORITY                               |

## Function Names

## Tables

gset\_seg\_tran  
gset\_text\_rep  
gset\_vis

SET SEGMENT TRANSFORMATION  
SET TEXT REPRESENTATION  
SET VISIBILITY

## 4.3.3.5 Level 1B

ginit\_pick  
ginq\_cur\_pick\_id  
ginq\_def\_pick\_data  
ginq\_pick\_st  
greq\_pick  
gset\_det  
gset\_pick\_id  
gset\_pick\_mode

INITIALISE PICK  
INQUIRE CURRENT PICK IDENTIFIER VALUE  
INQUIRE DEFAULT PICK DEVICE DATA  
INQUIRE PICK DEVICE STATE  
REQUEST PICK  
SET DETECTABILITY  
SET PICK IDENTIFIER  
SET PICK MODE

## 4.3.3.6 Level 1C

gget\_pick  
gsample\_pick

GET PICK  
SAMPLE PICK

## 4.3.3.7 Level 2A

gassoc\_seg\_ws  
gcopy\_seg\_ws  
ginsert\_seg

ASSOCIATE SEGMENT WITH WORKSTATION  
COPY SEGMENT FROM WORKSTATION  
INSERT SEGMENT

## 5 Type Definitions

### 5.1 Mapping of GKS data types

The GKS document specifies a set of abstract data types. This clause gives the mapping from those data types to the data types defined in this part of ISO/IEC 8651.

| GKS data type |                       | C binding data type  |
|---------------|-----------------------|----------------------|
| I             | integer               | Gint                 |
| R             | real                  | Gfloat               |
| S             | string                | char *               |
| P             | point                 | Gpoint               |
| L             | list of points        | Gpoint_list          |
| 2*R           | vector                | Gvec, Gfloat_size    |
| N             | name                  | Gint, char *, void * |
| E             | enumeration type      | typedef enum         |
| CLR           | colour representation | Gcolr_rep            |

### 5.2 Environmental Type Definitions

The data types defined in this section allow for the ease of porting GKS/C implementations between different environments. These types are used as the basis for all the other data types.

An implementation shall document the C types used for GKS/C types Gfloat and Gint.

**Gfloat** floating point number

This data type shall be defined by the implementation as a floating point type suitable for use within GKS/C. Suggest:

```
typedef float Gfloat;
```

**Gint** integer

This data type shall be defined by the implementation as a signed integral type suitable for use within GKS/C. Suggest:

```
typedef int Gint;
```

**size\_t**

An additional environmental data type is the type `size_t`. Depending on the environment, `size_t` can be defined as `int`, `unsigned`, `unsigned long`, etc. `size_t` is defined in the `#include` file `<stddef.h>`.

### 5.3 Implementation Dependent Type Definitions

This subclause presents the skeleton structure of the implementation dependent data types.

#### REMARKS:

- In order to produce syntactically correct data types, most implementation and registration dependent parts are of type `Gdata`. They can be replaced by any other structure.
- The data records of registered prompt and echo types are represented by structures with the names

## Implementation Dependent Type Definitions

## Type Definitions

`pet_rn` (n the prompt and echo type), whereas the data records of unregistered prompt and echo types are represented by structures with the names `pet_un` (-n the prompt and echo type).

- The data records of registered and unregistered GDPs, escape input and escape output are denoted in a similar way.
- For the input data records, the mandatory part, as specified in GKS (ISO 7942), is defined outside the union part of `Gxxx_data`. This mandatory part is implementation independent. Some of the `pet_rn` fields are given as structures of other type than `Gdata` for illustrative reasons. Thus, e.g. the `pet_r2` field of `Gchoice_data` contains a list of prompts, because GKS defines prompt and echo type 2 this way for the CHOICE device;
- For illustrative reasons, `Gitem_data` is sketchily (and incompletely) defined such that the items described in ISO 7942, Annex E could be represented by GKS/C data types. This definition is, however, not mandatory. Anyway, `Gitem_data` shall be defined in such a way that items which correspond with unsupported GDP's can be read from GKSM and written to GKSM.

**Gchoice\_data choice data record**

```
typedef struct {
    union Gchoice_pets {
        struct Gchoice_pet_r1 {
            Gdata    impl_dep;           /* impl. dep. */
        } pet_r1; /* pet 1 data */
        struct Gchoice_pet_r2 {
            Gint      num_prs;           /* number of prompts */
            Gpr_flag  *prs;              /* prompt array */
            Gdata      impl_dep;         /* impl. dep. */
        } pet_r2; /* pet 2 data */
        struct Gchoice_pet_r3 {
            Gint      num_strings;       /* number of choice strings */
            char      **strings;         /* array of choice strings */
            Gdata      impl_dep;         /* impl. dep. */
        } pet_r3; /* pet 3 data */
        struct Gchoice_pet_r4 {
            Gint      num_strings;       /* number of choice strings */
            char      **strings;         /* array of choice strings */
            Gdata      impl_dep;         /* impl. dep. */
        } pet_r4; /* pet 4 data */
        struct Gchoice_pet_r5 {
            Gint      seg_name;          /* segment name */
            Gint      num_pick_ids;      /* number of pick identifiers */
            Gint      *pick_ids;         /* array of pick identifiers */
            Gdata      impl_dep;         /* impl. dep. */
        } pet_r5; /* pet 5 data */
        struct Gchoice_pet_ul {
            Gdata      impl_dep;         /* impl. dep. */
        } pet_ul; /* pet -1 data */
        /* . . . impl. defined PET's */
    } pets;
} Gchoice_data;
```

## Type Definitions

## Implementation Dependent Type Definitions

---

**Gcolr\_rep** colour representation

```
typedef union {
    Grgb      rgb;      /* Red Green Blue colour specification */
} Gcolr_rep;
```

**REMARK.**In ISO/IEC 8806-4, Gcolr\_rep is implementation dependent.

---

**Gescape\_in\_data** escape input data record

```
typedef union {
    struct Gescape_in_rl {
        Gdata  reg_dep;  /* reg. dep. */
    } escape_rl;  /* escape 1 data */
    struct Gescape_in_ul {
        Gdata  impl_dep; /* impl. dep. */
    } escape_ul;  /* escape -1 data */
    /* etc. */
} Gescape_in_data;
```

---

**Gescape\_out\_data** escape output data record

```
typedef union {
    struct Gescape_out_rl {
        Gdata  reg_dep;  /* reg. dep. */
    } escape_rl;  /* escape 1 data */
    struct Gescape_out_ul {
        Gdata  impl_dep; /* impl. dep. */
    } escape_ul;  /* escape -1 data */
    /* etc. */
} Gescape_out_data;
```

---

**Ggdp\_data** gdp data record

```
typedef union {
    struct Ggdp_rl {
        Gdata  reg_dep;  /* reg. dep. */
    } gdp_rl;  /* gdp 1 data */
    struct Ggdp_ul {
        Gdata  impl_dep; /* impl. dep. */
    } gdp_ul;  /* gdp -1 data */
    /* etc. */
} Ggdp_data;
```

## Implementation Dependent Type Definitions

## Type Definitions

---

Gitem\_data item data record

```

typedef struct {
    Gint      type;          /* item type                */
    Gint      length;        /* item data record length  */
    union {
        Gctrl_flag clear_ws; /* control flag             */
        Gupd_regen_flag upd_ws; /* regen. flag             */
        struct Gdefer_st {
            Gdefer_mode defer_mode; /* deferral mode          */
            Girg_mode irg_mode; /* irg mode                */
        } defer_st; /* deferrral state          */
        /* etc. */
        Gdata gdp_unsupp; /* GDPs not supported by impl. */
        Gdata impl_dep; /* impl. dependent            */
    } data;
} Gitem_data;

```

## Type Definitions

## Implementation Dependent Type Definitions

---

Gloc\_data locator data record

```

typedef struct {
    union Gloc_pets {
        struct Gloc_pet_r1 {
            Gdata                                impl_dep;      /* impl. dep. */
        } pet_r1; /* pet 1 data */
        struct Gloc_pet_r2 {
            Gdata                                impl_dep;      /* impl. dep. */
        } pet_r2; /* pet 2 data */
        struct Gloc_pet_r3 {
            Gdata                                impl_dep;      /* impl. dep. */
        } pet_r3; /* pet 3 data */
        struct Gloc_pet_r4 {
            Gattr_ctrl_flag                      attr_ctrl_flag;
            /* attribute                          control flag */
            Gline_attrs                          line_attrs;
            /* polyline                          attributes */
            Gdata                                impl_dep;      /* impl. dep. */
        } pet_r4; /* pet 4 data */
        struct Gloc_pet_r5 {
            Gattr_ctrl_flag                      attr_ctrl_flag;
            /* attribute                          control flag */
            Gline_fill_ctrl_flag                 line_fill_ctrl_flag;
            /* polyline/fill area                 control flag */
            union Gloc_attrs {
                Gline_attrs                      line_attrs;
                /* polyline                      attrs. */
                Gfill_attrs                      fill_attrs;
                /* fill area                     attrs. */
            } attrs;
            Gdata                                impl_dep;      /* impl. dep. */
        } pet_r5; /* pet 5 data */
        struct Gloc_pet_r6 {
            Gdata                                impl_dep;      /* impl. dep. */
        } pet_r6; /* pet 6 data */
        struct Gloc_pet_ul {
            Gdata                                impl_dep;      /* impl. dep. */
        } pet_ul; /* pet -1 data */
        /* ... impl. defined PET's */
    } pets;
} Gloc_data;

```

---

**Gpick\_data** pick data record

```

typedef struct {
    union Gpick_pets {
        struct Gpick_pet_r1 {
            Gdata impl_dep; /* impl. dep. */
        } pet_r1; /* pet 1 data */
        struct Gpick_pet_r2 {
            Gdata impl_dep; /* impl. dep. */
        } pet_r2; /* pet 2 data */
        struct Gpick_pet_r3 {
            Gdata impl_dep; /* impl. dep. */
        } pet_r3; /* pet 3 data */
        struct Gpick_pet_ul {
            Gdata impl_dep; /* impl. dep. */
        } pet_ul; /* pet -1 data */
        /* etc. */
    } pets;
} Gpick_data;

```

---

**Gstring\_data** string data record

```

typedef struct {
    Gint in_buf_size; /* input buffer size (nr. of bytes) */
    Gint init_pos; /* initial [editing] position */
    union string_pets {
        struct Gstring_pet_r1 {
            Gdata impl_dep; /* impl. dep. */
        } pet_r1; /* pet 1 data */
        struct Gstring_pet_ul {
            Gdata impl_dep; /* impl. dep. */
        } pet_ul; /* pet -1 data */
        /* etc. */
    } pets;
} Gstring_data;

```

## Type Definitions

## Implementation Dependent Type Definitions

---

Gstroke\_data stroke data record

---

```

typedef struct {
    Gint    in_buf_size;    /* input buffer size (nr. of points) */
    Gint    init_pos;      /* initial [editing] position */
    Gfloat  x_interval;    /* X interval */
    Gfloat  y_interval;    /* Y interval */
    Gfloat  time_interval; /* time interval */
    union Gstroke_pets {
        struct Gstroke_pet_r1 {
            Gdata    impl_dep; /* impl. dep. */
        } pet_r1; /* pet 1 data */
        struct Gstroke_pet_r2 {
            Gdata    impl_dep; /* impl. dep. */
        } pet_r2; /* pet 2 data */
        struct Gstroke_pet_r3 {
            Gattr_ctrl_flag attr_ctrl_flag;
            /* attribute control flag */
            Gmarker_attrs marker_attrs;
            /* polymarker attrs. */
            Gdata    impl_dep; /* impl. dep. */
        } pet_r3; /* pet 3 data */
        struct Gstroke_pet_r4 {
            Gattr_ctrl_flag attr_ctrl_flag;
            /* attribute control flag */
            Gline_attrs line_attrs;
            /* polyline attrs. */
            Gdata    impl_dep; /* impl. dep. */
        } pet_r4; /* pet 4 data */
        struct Gstroke_pet_ul {
            Gdata    impl_dep; /* impl. dep. */
        } pet_ul; /* pet -1 data */
        /* etc. */
    } pets;
} Gstroke_data;

```

---

**Gval\_data** valuator data record

```

typedef struct {
    Gfloat  low_value;    /* low value */
    Gfloat  high_value;   /* high value */
    union Gval_pets {
        struct Gval_pet_r1 {
            Gdata  impl_dep; /* impl. dep. */
        } pet_r1; /* pet 1 data */
        struct Gval_pet_r2 {
            Gdata  impl_dep; /* impl. dep. */
        } pet_r2; /* pet 2 data */
        struct Gval_pet_r3 {
            Gdata  impl_dep; /* impl. dep. */
        } pet_r3; /* pet 3 data */
        struct Gval_pet_ul {
            Gdata  impl_dep; /* impl. dep. */
        } pet_ul; /* pet -1 data */
        /* etc. */
    } pets;
} Gval_data;

```

**5.4 Implementation Independent Type Definitions****REMARKS:**

- 1) For some two-field enumeration types, the order of the fields has been reversed with respect to ISO 7942; this reversal has been done in order to keep numerical compatibility with ISO 8651-1 (GKS/FORTRAN); the enumeration types are:  
Gclip\_ind, Gcolr\_avail, Gdisp\_surf\_empty, Gecho\_switch,  
Gupd\_regen\_flag, Gvis;
- 2) For the enumeration type STATUS, the fields NO CHOICE and NO PICK have been compacted to the field NO INPUT, in order to keep numerical compatibility with GKS/FORTRAN.

---

**Gasf** aspect source flag

```

typedef enum {
    GASF_BUNDLED,
    GASF_INDIV
} Gasf;

```

## Type Definitions

## Implementation Independent Type Definitions

**Gasfs** aspect source flags

---

```

typedef struct {
    Gasf  linetype;           /* linetype ASF */
    Gasf  linewidth;         /* linewidth scale factor ASF */
    Gasf  line_colr_ind;      /* polyline colour index ASF */
    Gasf  marker_type;        /* marker type ASF */
    Gasf  marker_size;        /* marker size scale factor ASF */
    Gasf  marker_colr_ind;    /* polymarker colour index ASF */
    Gasf  text_font_prec;     /* text font and precision ASF */
    Gasf  char_expan;         /* character expansion factor ASF */
    Gasf  char_space;         /* character spacing ASF */
    Gasf  text_colr_ind;      /* text colour index ASF */
    Gasf  fill_int_style;     /* fill area interior style ASF */
    Gasf  fill_style_ind;     /* fill area style index ASF */
    Gasf  fill_colr_ind;      /* fill area colour index ASF */
} Gasfs;

```

---

**Gattr\_ctrl\_flag** attribute control flag

```

typedef enum {
    GFLAG_CUR,
    GFLAG_SPECIF
} Gattr_ctrl_flag;

```

---

**Gattrs** attributes [used]

```

typedef enum {
    GATTR_LINE,
    GATTR_MARKER,
    GATTR_TEXT,
    GATTR_FILL
} Gattrs;

```

---

**Gclip** clipping

```

typedef struct {
    Gclip_ind clip_ind;      /* clipping indicator */
    Glimit clip_rect;        /* clipping rectangle */
} Gclip;

```

---

**Gclip\_ind** clipping indicator

```

typedef enum {
    GIND_NO_CLIP,
    GIND_CLIP
} Gclip_ind;

```

---

## Implementation Independent Type Definitions

## Type Definitions

---

Gcolr\_avail colour available

```
typedef enum {
    GAVAIL_MONOCHR,
    GAVAIL_COLR
} Gcolr_avail;
```

---

## Gcolr\_fac colour facilities

```
typedef struct {
    Gint      num_colrs;      /* num. of colours          */
    Gcolr_avail colr_avail;   /* colour availability      */
    Gint      num_pred_inds;  /* num. of predef. colour indices */
} Gcolr_fac;
```

---

## Gcoord\_switch coordinate switch

```
typedef enum {
    GCOORD_WC,
    GCOORD_NDC
} Gcoord_switch;
```

---

## Gctrl\_flag control flag

```
typedef enum {
    GFLAG_COND,
    GFLAG_ALWAYS
} Gctrl_flag;
```

---

## Gdata data

```
typedef struct {
    size_t size; /* size of data */
    void *data; /* pointer to data */
} Gdata;
```

---

## Gdc\_units device coordinate units

```
typedef enum {
    GDC_METRES,
    GDC_OTHER
} Gdc_units;
```

---

## Gdefer\_mode deferral mode

```
typedef enum {
    GDEFER_ASAP,
    GDEFER_BNIG,
    GDEFER_BNIL,
    GDEFER_ASTI
} Gdefer_mode;
```

## Type Definitions

## Implementation Independent Type Definitions

---

Gdet detectability

```
typedef enum {
    GSEG_UNDET,
    GSEG_DET
} Gdet;
```

---

Gdisp\_space\_size display space size

```
typedef struct {
    Gdc_units    dc_units;    /* device coordinate units */
    Gfloat_size  size_dc;    /* display space size [in] dc [units] */
    Gint_size    size_raster; /* display space size [in] raster [units] */
} Gdisp_space_size;
```

---

Gdisp\_surf\_empty display surface empty

```
typedef enum {
    GSURF_NOT_EMPTY,
    GSURF_EMPTY
} Gdisp_surf_empty;
```

---

Gdyn\_mod dynamic modification [accepted]

```
typedef enum {
    GDYN_IRG,
    GDYN_IMM
} Gdyn_mod;
```

---

Gdyn\_mod\_seg\_attrs dynamic modification [of] segment attributes

```
typedef struct {
    Gdyn_mod  tran;    /* changeability of: */
    Gdyn_mod  invis_vis; /* segment transformation */
    Gdyn_mod  vis_invis; /* appearing (invisible → visible) */
    Gdyn_mod  highl;    /* disappearing (visible → invisible) */
    Gdyn_mod  pri;      /* highlighting */
    Gdyn_mod  add_prims; /* priority */
    Gdyn_mod  del;      /* addition of primitives to segment */
    Gdyn_mod  del;      /* deletion of segment */
} Gdyn_mod_seg_attrs;
```

## Implementation Independent Type Definitions

## Type Definitions

**Gdyn\_mod\_ws\_attrs** dynamic modification [of] workstation attributes

```
typedef struct {
    Gdyn_mod    line_bundle;    /* changeability of:          */
    Gdyn_mod    marker_bundle;  /* polyline representation    */
    Gdyn_mod    text_bundle;    /* polymarker representation  */
    Gdyn_mod    fill_bundle;    /* text representation        */
    Gdyn_mod    pat_rep;        /* fill area representation   */
    Gdyn_mod    colr_rep;       /* pattern representation     */
    Gdyn_mod    colr_rep;       /* colour representation      */
    Gdyn_mod    ws_tran;       /* workstation transformation  */
} Gdyn_mod_ws_attrs;
```

**Gecho\_switch** echo switch

```
typedef enum {
    GSWITCH_NO_ECHO,
    GSWITCH_ECHO
} Gecho_switch;
```

**Gfill\_attrs** fill area attributes

```
typedef struct {
    Gasf        int_style_asf;  /* fill area interior style asf */
    Gasf        style_ind_asf;  /* fill area style index asf    */
    Gasf        colr_ind_asf;   /* fill area colour index asf   */
    Gint        ind;            /* fill area index              */
    Gfill_bundle bundle;        /* fill area bundle             */
} Gfill_attrs;
```

**Gfill\_bundle** fill area bundle

```
typedef struct {
    Gfill_int_style int_style;  /* fill area interior style */
    Gint            style_ind;  /* fill area style index    */
    Gint            colr_ind;   /* fill area colour index   */
} Gfill_bundle;
```

**Gfill\_fac** fill area facilities

```
typedef struct {
    Gint        num_int_styles;  /* num. of interior styles */
    Gfill_int_style int_styles[4]; /* list of available interior styles */
    Gint_list    hatch_styles;  /* list of available hatch styles */
    Gint        num_pred_inds;  /* num. of predef. fill area indices */
} Gfill_fac;
```

## Type Definitions

## Implementation Independent Type Definitions

---

**Gfill\_int\_style** fill area interior style

```
typedef enum {
    GSTYLE_HOLLOW,
    GSTYLE_SOLID,
    GSTYLE_PAT,
    GSTYLE_HATCH
} Gfill_int_style;
```

---

**Gfloat\_size** float size

```
typedef struct {
    Gfloat size_x; /* x size */
    Gfloat size_y; /* y size */
} Gfloat_size;
```

---

**Ghighl** highlighting

```
typedef enum {
    GSEG_NORM,
    GSEG_HIGHL
} Ghighl;
```

---

**Ghor\_text\_align** horizontal text alignment

```
typedef enum {
    GHOR_NORM,
    GHOR_LEFT,
    GHOR_CTR,
    GHOR_RIGHT
} Ghor_text_align;
```

---

**Gin\_class** input class

```
typedef enum {
    GIN_NONE,
    GIN_LOC,
    GIN_STROKE,
    GIN_VAL,
    GIN_CHOICE,
    GIN_PICK,
    GIN_STRING
} Gin_class;
```

---

**Gin\_status** [input] status

```
typedef enum {
    GIN_STATUS_NONE,
    GIN_STATUS_OK,
    GIN_STATUS_NO_IN
} Gin_status;
```

---

## Implementation Independent Type Definitions

## Type Definitions

---

Gindiv\_attrs individual attributes

```

typedef struct {
    Gint      linetype;          /* linetype                */
    Gfloat    linewidth;        /* linewidth scale factor  */
    Gint      line_colr_ind;     /* polyline colour index   */
    Gint      marker_type;      /* marker type             */
    Gfloat    marker_size;      /* marker size scale factor */
    Gint      marker_colr_ind;   /* polymarker colour index */
    Gtext_font_prec text_font_prec; /* text font and precision */
    Gfloat    char_expan;       /* character expansion factor */
    Gfloat    char_space;       /* character spacing        */
    Gint      text_colr_ind;     /* text colour index       */
    Gfill_int_style fill_int_style; /* fill area interior style */
    Gint      fill_style_ind;    /* fill area style index   */
    Gint      fill_colr_ind;     /* fill area colour index  */
    Gasfs     asfs;             /* aspect source flags     */
} Gindiv_attrs;

```

---

## Ginq\_type inquire type

```

typedef enum {
    GINQ_SET,
    GINQ_REALIZED
} Ginq_type;

```

---

## Gint\_list integer list

```

typedef struct {
    Gint  num_ints; /* num. of integers in list */
    Gint  *ints;    /* list of integers         */
} Gint_list;

```

---

## Gint\_size integer size

```

typedef struct {
    Gint  size_x; /* x size */
    Gint  size_y; /* y size */
} Gint_size;

```

---

## Girg\_mode implicit regeneration mode

```

typedef enum {
    GIRG_SUPPR,
    GIRG_ALLOWED
} Girg_mode;

```

---

## Type Definitions

## Implementation Independent Type Definitions

## Glevel GKSlevel

```
typedef enum {
    GLEVEL_0A,
    GLEVEL_0B,
    GLEVEL_0C,
    GLEVEL_1A,
    GLEVEL_1B,
    GLEVEL_1C,
    GLEVEL_2A,
    GLEVEL_2B,
    GLEVEL_2C
} Glevel;
```

## Glimit limit

```
typedef struct {
    Gfloat x_min; /* x min */
    Gfloat x_max; /* x max */
    Gfloat y_min; /* y min */
    Gfloat y_max; /* y max */
} Glimit;
```

## Gline\_attrs polyline attributes

```
typedef struct {
    Gasf type_asf; /* linetype asf */
    Gasf width_asf; /* linewidth asf */
    Gasf colr_ind_asf; /* polyline colour index asf */
    Gint ind; /* polyline index */
    Gline_bundle bundle; /* polyline bundle */
} Gline_attrs;
```

## Gline\_bundle polyline bundle

```
typedef struct {
    Gint type; /* linetype */
    Gfloat width; /* linewidth scale factor */
    Gint colr_ind; /* polyline colour index */
} Gline_bundle;
```

## Gline\_facs polyline facilities

```
typedef struct {
    Gint_list types; /* list of linetypes */
    Gint num_widths; /* num. of available linewidths */
    Gfloat nom_width; /* nominal linewidth */
    Gfloat min_width; /* min. linewidth */
    Gfloat max_width; /* max. linewidth */
    Gint num_pred_inds; /* num. of predef. polyline indices */
} Gline_facs;
```

## Implementation Independent Type Definitions

## Type Definitions

**Gline\_fill\_ctrl\_flag** polyline/fill area control flag

```
typedef enum {
    GFLAG_LINE,
    GFLAG_FILL
} Gline_fill_ctrl_flag;
```

**Gmarker\_attr**s polymarker attributes

```
typedef struct {
    Gasf      type_asf;      /* marker type asf          */
    Gasf      size_asf;      /* marker size scale factor asf */
    Gasf      colr_ind_asf;  /* marker colour index asf    */
    Gint      ind;           /* polymarker index          */
    Gmarker_bundle bundle;   /* polymarker bundle         */
} Gmarker_attr;
```

**Gmarker\_bundle** polymarker bundle

```
typedef struct {
    Gint      type;          /* marker type                */
    Gfloat     size;         /* marker size scale factor   */
    Gint      colr_ind;      /* polymarker colour index    */
} Gmarker_bundle;
```

**Gmarker\_fac**s polymarker facilities

```
typedef struct {
    Gint_list types;         /* list of marker types      */
    Gint      num_sizes;     /* num. of available marker sizes */
    Gfloat     nom_size;     /* nominal marker size       */
    Gfloat     min_size;     /* min. marker size         */
    Gfloat     max_size;     /* max. marker size         */
    Gint      num_pred_inds; /* num. of predef. polymarker indices */
} Gmarker_fac;
```

**Gmax\_ws\_st\_tables** max. [length of] workstation state tables

```
typedef struct {
    /* max. num. of : */
    Gint  line_bundles; /* polyline bundle table entries */
    Gint  marker_bundles; /* polymarker bundle table entries */
    Gint  text_bundles; /* text bundle table entries */
    Gint  fill_bundles; /* fill area bundle table entries */
    Gint  pat_reps; /* pattern table entries */
    Gint  colr_reps; /* colour table entries */
} Gmax_ws_st_tables;
```

## Type Definitions

## Implementation Independent Type Definitions

---

**Gmore\_simult\_events** more simultaneous events

```
typedef enum {
    GSIMULT_NO_MORE,
    GSIMULT_MORE
} Gmore_simult_events;
```

---

**Gnew\_frame\_nec\_upd** new frame [action] necessary [at] update

```
typedef enum {
    GNEW_NO,
    GNEW_YES
} Gnew_frame_nec_upd;
```

---

**Gnum\_in** number [of] input [devices]

```
typedef struct {
    Gint    loc;        /* num. of locator devices    */
    Gint    stroke;     /* num. of stroke devices    */
    Gint    val;        /* num. of valuator devices  */
    Gint    choice;     /* num. of choice devices    */
    Gint    pick;       /* num. of pick devices      */
    Gint    string;     /* num. of string devices    */
} Gnum_in;
```

---

**Gop\_mode** operating mode

```
typedef enum {
    GOP_REQ,
    GOP_SAMPLE,
    GOP_EVENT
} Gop_mode;
```

---

**Gop\_st** operating state

```
typedef enum {
    GST_GKCL,
    GST_GKOP,
    GST_WSOP,
    GST_WSAC,
    GST_SGOP
} Gop_st;
```

---

**Gpat\_rep** pattern representation

```
typedef struct {
    Gint_size  dims;        /* colour array's dimensions */
    Gint       *colr_array; /* colour array                */
} Gpat_rep;
```

## Implementation Independent Type Definitions

## Type Definitions

---

**Gpick pick [value]**

```
typedef struct {
    Gint    seg_name;    /* segment name    */
    Gint    pick_id;     /* pick identifier */
} Gpick;
```

---

**Gpoint point**

```
typedef struct {
    Gfloat  x;    /* x coordinate */
    Gfloat  y;    /* y coordinate */
} Gpoint;
```

---

**Gpoint\_list point list**

```
typedef struct {
    Gint    num_points; /* num. of points in the list */
    Gpoint  *points;    /* list of points             */
} Gpoint_list;
```

---

**Gpr\_flag prompt flag**

```
typedef enum {
    GPR_OFF,
    GPR_ON
} Gpr_flag;
```

---

**Gpres\_inval presence [of] invalid [values]**

```
typedef enum {
    GINVAL_ABSENT,
    GINVAL_PRESENT
} Gpres_inval;
```

---

**Gprim\_attrs primitive attributes**

```
typedef struct {
    Gint    line_ind;    /* polyline index    */
    Gint    marker_ind; /* polymarker index  */
    Gint    text_ind;   /* text index        */
    Gfloat  char_ht;    /* character height  */
    Gvec    char_up_vec; /* character up vector */
    Gfloat  char_width; /* character width    */
    Gvec    char_base_vec; /* character base vector */
    Gtext_path text_path; /* text path          */
    Gtext_align text_align; /* text alignment    */
    Gint    fill_ind;   /* fill area index    */
    Gvec    pat_width_vec; /* pattern width vector */
    Gvec    pat_ht_vec; /* pattern height vector */
    Gpoint  pat_ref_point; /* pattern reference point */
} Gprim_attrs;
```

---