

© British Standards Institution. No part of this publication may be photocopied or otherwise reproduced without the prior permission in writing of BSI

Draft for Development

# Computer graphics: Graphical Kernel System (GKS) language bindings

## Part 2. Pascal (ISO/DIS 8651/2)

Projet à développer  
Infographique  
Langage d'interface avec GKS  
Partie 2. Pascal

Vornorm  
Graphische Datenverarbeitung: Sprachschalen für das Graphische Kernsystem(GKS)  
Teil 2. Pascal

To avoid delay in preparing the content of this publication for printing, it was reproduced from the only available copy of the finally agreed document. The resultant poor print quality is very much regretted. The price has been adjusted accordingly.



**Documents**

Document Sharing Hub

## Contents

|                                                             | BS Page    |                                                                              | BS Page |
|-------------------------------------------------------------|------------|------------------------------------------------------------------------------|---------|
| National foreword                                           | 2          | 5.9.8 Pixel inquiries                                                        | 46      |
| Committees responsible                                      | Back cover | 5.9.9 Inquiry function for GKS error state list                              | 47      |
| <b>Draft for Development</b>                                |            | 5.10 Utility functions                                                       | 47      |
| 0 Introduction                                              | 3          | 5.11 Error handling                                                          | 47      |
| 1 Scope and field of application                            | 4          | 6 Pascal GKS data structures                                                 | 49      |
| 2 References                                                | 5          | 6.1 Implementation defined constants                                         | 49      |
| 3 The Pascal language binding of GKS                        | 6          | 6.2 Required constants                                                       | 49      |
| 3.1 Specification                                           | 6          | 6.3 General types                                                            | 49      |
| 3.2 Mapping of GKS function names to Pascal procedure names | 6          | 6.4 Names used by GKS                                                        | 49      |
| 3.3 The many-one nature of the Pascal interface             | 6          | 6.5 GKS enumerated types                                                     | 49      |
| 3.4 The one-one nature of the Pascal interface              | 6          | 6.6 Array types                                                              | 51      |
| 3.5 Implementation of the two interfaces                    | 6          | 6.7 Pointer types                                                            | 51      |
| 3.6 Representation of GKS data types                        | 15         | 6.8 Set types                                                                | 51      |
| 3.7 Naming conventions for data types                       | 15         | 6.9 Record types                                                             | 51      |
| 3.8 Implementation dependent characteristics                | 15         | 7 The one-one mapping for output primitive representations and input classes | 58      |
| 3.9 Level of Pascal                                         | 16         | 7.1 Principles                                                               | 58      |
| 4 Error handling                                            | 18         | 7.2 Additional procedures for output primitive representations               | 58      |
| 4.1 The error handling function                             | 18         | 7.3 Additional procedures for the input classes                              | 60      |
| 4.2 Pascal specific GKS errors                              | 18         | 7.4 Additional procedures for the inquiries                                  | 65      |
| 5 GKS functions                                             | 19         | 7.5 Additional data types in alphabetical order                              | 71      |
| 5.1 Notational conventions                                  | 19         |                                                                              |         |
| 5.2 Control functions                                       | 19         | <b>Annexes</b>                                                               |         |
| 5.3 Output functions                                        | 21         | A Data types in compilation order                                            | 75      |
| 5.4 Output attributes                                       | 24         | A.1 Implementation defined constants                                         | 75      |
| 5.4.1 Workstation independent primitive attributes          | 24         | A.2 Required constants                                                       | 75      |
| 5.4.2 Workstation attributes (representations)              | 27         | A.3 Implementation defined GKS data records                                  | 75      |
| 5.5 Transformation functions                                | 28         | A.4 Error logging and connection files                                       | 75      |
| 5.5.1 Normalization transformation                          | 28         | A.5 General types                                                            | 75      |
| 5.5.2 Workstation transformation                            | 29         | A.6 Types applicable to workstation control procedures                       | 76      |
| 5.6 Segment functions                                       | 29         | A.7 Types applicable to normalization transformation procedures              | 76      |
| 5.6.1 Segment manipulation functions                        | 29         | A.8 Types applicable to attribute setting procedures                         | 77      |
| 5.6.2 Segment attributes                                    | 30         | A.9 Types applicable to segment procedures                                   | 77      |
| 5.7 Input functions                                         | 31         | A.10 Types applicable to input procedures                                    | 78      |
| 5.7.1 Initialisation of input devices                       | 31         | A.11 Types applicable to GKS description                                     | 78      |
| 5.7.2 Setting the mode of input devices                     | 32         | A.12 Types applicable to GKS state                                           | 78      |
| 5.7.3 Request input functions                               | 32         | A.13 Types applicable to workstation state                                   | 78      |
| 5.7.4 Sample input functions                                | 33         | A.14 Types applicable to workstation description                             | 78      |
| 5.7.5 Event input functions                                 | 33         | A.15 Types applicable to segment state                                       | 78      |
| 5.8 Metafile functions                                      | 34         | A.16 Types applicable to the additional one-one procedures                   | 79      |
| 5.9 Inquiry functions                                       | 34         | B Example program                                                            | 80      |
| 5.9.1 Convention                                            | 34         | C Function lists                                                             | 86      |
| 5.9.2 Inquiry function for operating state value            | 34         | C.1 GKS functions                                                            | 86      |
| 5.9.3 Inquiry functions for GKS description table           | 35         | C.2 Pascal functions                                                         | 90      |
| 5.9.4 Inquiry functions for GKS state list                  | 35         |                                                                              |         |
| 5.9.5 Inquiry functions for workstation state list          | 37         |                                                                              |         |
| 5.9.6 Inquiry functions for workstation description table   | 41         |                                                                              |         |
| 5.9.7 Inquiry functions for segment state list              | 45         |                                                                              |         |

This publication is not to be regarded as a British Standard.

It is being issued in the Draft for Development series of publications and is of a provisional nature because the subject matter is in the process of continuing development within the International Standards Committee (ISO/TC 97/ SC 21 Information Retrieval, Transfer and Management for Open Systems Interconnection).

It should be applied on this provisional basis, so that information and experience of its practical application may be obtained.

A review of this Draft for Development will be carried out not later than two years after its publication. Notification of the start of the review period, with a request for the submission of comments from users of this Draft for Development will be made in an announcement in the appropriate issue of *BSI News*. Observations which it is felt should receive attention before the official call for comments will be welcomed.

According to the replies received, the responsible BSI Committee will judge whether the Draft for Development can be converted into a British Standard or what other action should be taken.

## National foreword

This Part of DD 114 has been prepared under the direction of the Office and Information Standards Committee.

The primary objective of this Part of DD 114 is to give advance guidance to product and systems designers concerned with computer graphics.

This Part of DD 114 is identical with the Draft International Standard ISO/DIS 8651/2-1985 'Information processing systems — Computer graphics — Graphical Kernel System (GKS) language bindings — Part 2: Pascal', issued by the International Organization for Standardization (ISO). It is a revision of DD 114 : Part 2 : 1985, which was identical with the International Draft Proposal ISO/DP 8651/2 and which is withdrawn.

For ease of reproduction the ISO text has been reproduced unchanged.

The UK is actively participating through BSI Technical Committee IST/21\* in the development of the ISO Draft International Standard. The Technical Committee will review any agreed amendments or addenda to the ISO text and will, where appropriate, update the text of this Draft for Development via an amendment or revision. When the ISO draft reaches an appropriate level of agreement and stability, this Draft for Development may be converted to a British Standard.

In DD 114, three language bindings to BS 6390 currently either exist or are scheduled for publication. These are as follows.

DD 114 : Part 1 FORTRAN : in preparation.

DD 114 : Part 2 Pascal

DD 114 : Part 3 Ada : first published in December 1985.

It is expected that further Parts will be added to DD 114.

A brochure giving details of existing and projected British Standards publications for Open Systems Interconnection (OSI) may be obtained from the Sales Department, BSI, Linford Wood, Milton Keynes, MK14 6LE, at no charge. Press releases for new OSI, data communications and computer graphics British Standards publications will be issued to the computing press, and details of all new publications will be recorded in full in *BSI News*.

**Terminology and conventions.** The text of the Draft International Standard has been approved as suitable for publication as a Draft for Development without deviation. Some terminology and certain conventions are not identical with those used in Drafts for Development; attention is drawn especially to the following.

Wherever the words 'document' appear, referring to this Draft for Development, they should be read as 'Draft for Development'.

### Cross-references

| International Standard | Corresponding British Standard                                                                                                             |
|------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| ISO 7185-1983          | BS 6192 : 1982 Specification for computer programming language Pascal<br>(Identical)                                                       |
| ISO 7942-1985          | BS 6390 : 1985 Specification for a set of functions for computer graphics programming,<br>the Graphical Kernel System (GKS)<br>(Identical) |

\*Previously OIS/121.

**DRAFT INTERNATIONAL STANDARD**

# **Information processing systems - Computer graphics - Graphical Kernel System (GKS) language bindings - Part 2: Pascal**

## **0 Introduction**

The Graphical Kernel System (GKS) functional description is ISO 7942. As explained in the Scope and Field of Applications 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 document is to define a standard binding for the Pascal computer programming language.

## 1 Scope and field of application

ISO 7942 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 document specifies such a language dependent layer for the Pascal language.

[ISO page 2]

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1.

## 2 References

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

ISO 7185, *Programming languages - Pascal*.

[ISO page 3]

**Documents**

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1.

### 3 The Pascal language binding of GKS

#### 3.1 Specification

The GKS language binding interface for ISO Pascal (ISO 7185) shall be as described in clauses 3, 4, 5, 6, and 7.

#### 3.2 Mapping of GKS function names to Pascal procedure names

The function names of GKS are all mapped to Pascal procedures beginning with the letter "G". Words and phrases used in the GKS function names are often abbreviated in the Pascal representation. There is a set of such abbreviations given in Tables 1 to 3. For example, the abbreviation for the GKS function DELETE SEGMENT FROM WORKSTATION is GDelSegWs. "Del", "Seg", "Ws" are the abbreviations for DELETE, SEGMENT and WORKSTATION. Conjunctives such as "from", "and", "of" and "to" are mapped to null strings, as are a number of other words used in the GKS abstract names. For example, INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES is mapped to GInqMaxWsSt. Here LENGTH and TABLES are represented by null strings.

#### 3.3 The many-one nature of the Pascal interface

There is not a strict one-to-one correspondence between GKS abstract functions and Pascal procedures. A method employing variant records is used to represent several logically related GKS abstract functions by one Pascal procedure. The first parameter of such a procedure is always an enumerated type which is the tag field of a variant record which is itself a parameter of the Pascal procedure. This technique is used across two classes of abstract functions - those relating to the setting and inquiring of output primitive representations, and those relating to the setting and inquiring of the input classes. Where this method is used, the rules for deriving the Pascal name of the GKS abstract function are:

##### a) Output Primitive Representations

- 1) Replace the words Polyline, Polymarker, Text, Fill Area by "Prim" (which is the abbreviation for PRIMITIVE).
- 2) The first parameter of the function is an enumerated type (GTPrim) which has one of the values Polyline, Polymarker, GKSText, Fill.

##### b) Input Classes

- 1) Replace the words Locator, Stroke, Valuator, Choice, Pick, String by "Input".
- 2) The first parameter of the function is an enumerated type (GTInputClass) which has one of the values Locator, Stroke, Valuator, Choice, Pick, InputString.

#### 3.4 The one-one nature of the Pascal interface

The Pascal interface to GKS reflects the GKS major dimensions of Output Primitive Representations and Input Classes. This interface is economical in procedure definitions, and easy to use. However, the possibility exists that on small systems such an interface might cause difficulties, especially with respect to implementation of the Input Classes. Therefore, the Pascal Binding also adopts a mandatory representation which uses a one-one mapping for the setting of primitive representations and input classes. The collection of procedures and data type declarations required to achieve this is given in clause 7 of this document. Table 4 shows the relevant Pascal procedures.

#### 3.5 Implementation of the two interfaces

Since either method referred to in 3.2 and 3.3 can be implemented easily in terms of the other, this does not present a great burden for the implementor, nor does it cause an additional burden for application programs. Implementors are encouraged to use one method in the core of their implementation, and if possible provide the procedures and data type declarations of the other method in a separate library, which the application programmer can load if required. This will in turn encourage application programmers to use only one or other of the two possible interfaces. In any event both sets of procedures shall be provided.

[ISO page 4]

## The Pascal language binding of GKS

## Implementation of the two interfaces

Table 1 - Abbreviations ordered alphabetically

|                               |           |
|-------------------------------|-----------|
| ACCUMULATE                    | Accum     |
| ALIGNMENT                     | Align     |
| ASPECT SOURCE FLAGS           | ASF       |
| ASSOCIATE                     | Assoc     |
| ATTRIBUTE(S)                  | Attr      |
| AVAILABLE                     | NULL      |
| AVAILABLE                     | NULL      |
| CHARACTER                     | Char      |
| CLASSIFICATION                | Class     |
| CLIPPING                      | Clip      |
| COLOUR                        | Col       |
| CONNECTION                    | Conn      |
| CURRENT                       | Cur       |
| DEFAULT                       | Def       |
| DEFERRAL                      | Defer     |
| DELETE                        | Del       |
| DETECTABILITY                 | Det       |
| DEVICE                        | NULL      |
| DIMENSIONS                    | Dim       |
| DYNAMIC                       | Dyn       |
| EVALUATE                      | Eval      |
| EXPANSION                     | Expan     |
| FACILITIES                    | Facil     |
| FACTOR                        | NULL      |
| FILL AREA                     | Fill      |
| GENERALIZED DRAWING PRIMITIVE | GDP       |
| GKSM                          | NULL      |
| HIGHLIGHTING                  | Highlight |
| IDENTIFIER                    | Id        |
| IN USE                        | NULL      |
| INDEX                         | Ind       |
| INDICATOR                     | NULL      |
| INDICES                       | Ind       |
| INDIVIDUAL                    | Indiv     |
| INITIALISE                    | Init      |
| INPUT PRIORITY                | Priority  |
| INQUIRE                       | Inq       |
| INTERIOR                      | Int       |
| LENGTH                        | NULL      |
| LOGICAL                       | NULL      |
| MATRIX                        | NULL      |
| MAXIMUM                       | Max       |
| MODIFICATION                  | Mod       |
| NORMALIZATION                 | Norm      |
| NUMBER(S)                     | Num       |
| OPERATING                     | Op        |
| POLYLINE COLOUR               | LineCol   |
| POLYLMARKER COLOUR            | MarkerCol |
| PRECISION                     | Prec      |
| PREDEFINED                    | Pred      |
| PRIMITIVE                     | Prim      |
| QUEUE                         | NULL      |
| REFERENCE                     | Ref       |

[ISO page 5]

Documents

## Implementation of the two interfaces

## The Pascal language binding of GKS

|                |       |
|----------------|-------|
| REPRESENTATION | Rep   |
| REQUEST        | Req   |
| SCALE FACTOR   | Scale |
| SEGMENT(S)     | Seg   |
| SET OF         | NULL  |
| SIMULTANEOUS   | NULL  |
| SPACE          | NULL  |
| STATE          | St    |
| SUPPORTED      | NULL  |
| TABLES         | NULL  |
| TRANSFORMATION | Tran  |
| UPDATE         | Upd   |
| VALUE(S)       | NULL  |
| VISIBILITY     | Vis   |
| WORKSTATION    | Ws    |

NOTE - NULL represents the null string

## The Pascal language binding of GKS

## Implementation of the two interfaces

Table 2 - GKS function names and Pascal names in order of appearance

|                                     |                             |
|-------------------------------------|-----------------------------|
| OPEN GKS                            | G-Open-GKS                  |
| CLOSE GKS                           | G-Close-GKS                 |
| OPEN WORKSTATION                    | G-Open-Ws                   |
| CLOSE WORKSTATION                   | G-Close-Ws                  |
| ACTIVATE WORKSTATION                | G-Activate-Ws               |
| DEACTIVATE WORKSTATION              | G-Deactivate-Ws             |
| CLEAR WORKSTATION                   | G-Clear-Ws                  |
| REDRAW ALL SEGMENTS ON WORKSTATION  | G-Redraw-Seg-Ws             |
| UPDATE WORKSTATION                  | G-Upd-Ws                    |
| SET DEFERRAL STATE                  | G-Set-Defer-St              |
| MESSAGE                             | G-Message                   |
| MESSAGE (Pascal level 0)            | G-MessageString             |
| ESCAPE                              | G-Escape                    |
| POLYLINE                            | G-Polyline                  |
| POLYMARKER                          | G-Polymarker                |
| TEXT                                | G-Text                      |
| TEXT (Pascal level 0)               | G-TextString                |
| FILL AREA                           | G-Fill                      |
| CELL ARRAY                          | G-CellArray                 |
| GENERALIZED DRAWING PRIMITIVE       | G-gdp                       |
| SET POLYLINE INDEX                  | G-Set-Prim-Ind-(Polyline)   |
| SET POLYMARKER INDEX                | G-Set-Prim-Ind-(Polymarker) |
| SET TEXT INDEX                      | G-Set-Prim-Ind-(GKSText)    |
| SET FILL AREA INDEX                 | G-Set-Prim-Ind-(Fill)       |
| SET LINETYPE                        | G-Set-LineType              |
| SET LINEWIDTH SCALE FACTOR          | G-Set-LineWidth-Scale       |
| SET POLYLINE COLOUR INDEX           | G-Set-Line-Col-Ind          |
| SET MARKER TYPE                     | G-Set-Marker-Type           |
| SET MARKER SIZE SCALE FACTOR        | G-Set-Marker-Size-Scale     |
| SET POLYMARKER COLOUR INDEX         | G-Set-Marker-Col-Ind        |
| SET TEXT FONT AND PRECISION         | G-Set-Text-Font-Prec        |
| SET CHARACTER EXPANSION FACTOR      | G-Set-Char-Expan            |
| SET CHARACTER SPACING               | G-Set-Char-Spacing          |
| SET TEXT COLOUR INDEX               | G-Set-Text-Col-Ind          |
| SET CHARACTER HEIGHT                | G-Set-Char-Height           |
| SET CHARACTER UP VECTOR             | G-Set-Char-Up-Vector        |
| SET TEXT PATH                       | G-Set-Text-Path             |
| SET TEXT ALIGNMENT                  | G-Set-Text-Align            |
| SET FILL AREA INTERIOR STYLE        | G-Set-Fill-Int-Style        |
| SET FILL AREA STYLE INDEX           | G-Set-Fill-Style-Ind        |
| SET FILL AREA COLOUR INDEX          | G-Set-Fill-Col-Ind          |
| SET PATTERN SIZE                    | G-Set-Pattern-Size          |
| SET PATTERN REFERENCE POINT         | G-Set-Pattern-Ref-Point     |
| SET ASPECT SOURCE FLAGS             | G-Set-ASF                   |
| SET PICK IDENTIFIER                 | G-Set-Pick-Id               |
| SET POLYLINE REPRESENTATION         | G-Set-Prim-Rep-(Polyline)   |
| SET POLYMARKER REPRESENTATION       | G-Set-Prim-Rep-(Polymarker) |
| SET TEXT REPRESENTATION             | G-Set-Prim-Rep-(GKSText)    |
| SET FILL AREA REPRESENTATION        | G-Set-Prim-Rep-(Fill)       |
| SET PATTERN REPRESENTATION          | G-Set-Pattern-Rep           |
| SET COLOUR REPRESENTATION           | G-Set-Col-Rep               |
| SET WINDOW                          | G-Set-Window                |
| SET VIEWPORT                        | G-Set-Viewport              |
| SET VIEWPORT INPUT PRIORITY         | G-Set-Viewport-Priority     |
| SELECT NORMALIZATION TRANSFORMATION | G-Select-Norm-Tran          |
| SET CLIPPING INDICATOR              | G-Set-Clip                  |
| SET WORKSTATION WINDOW              | G-Set-Ws-Window             |
| SET WORKSTATION VIEWPORT            | G-Set-Ws-Viewport           |
| CREATE SEGMENT                      | G-Create-Seg                |
| CLOSE SEGMENT                       | G-Close-Seg                 |
| RENAME SEGMENT                      | G-Rename-Seg                |
| DELETE SEGMENT                      | G-Del-Seg                   |
| DELETE SEGMENT FROM WORKSTATION     | G-Del-Seg-Ws                |
| ASSOCIATE SEGMENT WITH WORKSTATION  | G-Assoc-Seg-Ws              |

[ISO page 7]

Documents

## Implementation of the two interfaces

COPY SEGMENT TO WORKSTATION  
 INSERT SEGMENT  
 SET SEGMENT TRANSFORMATION  
 SET VISIBILITY  
 SET HIGHLIGHTING  
 SET SEGMENT PRIORITY  
 SET DETECTABILITY  
 INITIALISE LOCATOR  
 INITIALISE STROKE  
 INITIALISE VALUATOR  
 INITIALISE CHOICE  
 INITIALISE PICK  
 INITIALISE STRING  
 SET LOCATOR MODE  
 SET STROKE MODE  
 SET VALUATOR MODE  
 SET CHOICE MODE  
 SET PICK MODE  
 SET STRING MODE  
 REQUEST LOCATOR  
 REQUEST STROKE  
 REQUEST VALUATOR  
 REQUEST CHOICE  
 REQUEST PICK  
 REQUEST STRING  
 SAMPLE LOCATOR  
 SAMPLE STROKE  
 SAMPLE VALUATOR  
 SAMPLE CHOICE  
 SAMPLE PICK  
 SAMPLE STRING  
 AWAIT EVENT  
 FLUSH DEVICE EVENTS  
 GET LOCATOR  
 GET STROKE  
 GET VALUATOR  
 GET CHOICE  
 GET PICK  
 GET STRING  
 WRITE ITEM TO GKSM  
 GET ITEM TYPE FROM GKSM  
 READ ITEM FROM GKSM  
 INTERPRET ITEM  
 INQUIRE OPERATING STATE VALUE  
 INQUIRE LEVEL OF GKS  
 INQUIRE LIST OF AVAILABLE WORKSTATION TYPES  
 INQUIRE WORKSTATION MAXIMUM NUMBERS  
 INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER  
 INQUIRE SET OF OPEN WORKSTATIONS  
 INQUIRE SET OF ACTIVE WORKSTATIONS  
 INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES  
 INQUIRE CURRENT PICK IDENTIFIER  
 INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES  
 INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER  
 INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS  
 INQUIRE NORMALIZATION TRANSFORMATION  
 INQUIRE CLIPPING  
 INQUIRE NAME OF OPEN SEGMENT  
 INQUIRE SET OF SEGMENT NAMES IN USE  
 INQUIRE MORE SIMULTANEOUS EVENTS  
 INQUIRE WORKSTATION CONNECTION AND TYPE  
 INQUIRE WORKSTATION STATE  
 INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES  
 INQUIRE LIST OF POLYLINE INDICES  
 INQUIRE LIST OF POLYMARKER INDICES  
 INQUIRE LIST OF TEXT INDICES  
 INQUIRE LIST OF FILL AREA INDICES  
 INQUIRE POLYLINE REPRESENTATION

[ISO page 8]

## The Pascal language binding of GKS

G-Copy-Seg-Ws  
 G-Insert-Seg  
 G-Set-Seg-Tran  
 G-Set-Vis  
 G-Set-Highlight  
 G-Set-Seg-Priority  
 G-Set-Det  
 G-Init-Input-(Locator  
 G-Init-Input-(Stroke  
 G-Init-Input-(Valuator  
 G-Init-Input-(Choice  
 G-Init-Input-(Pick  
 G-Init-Input-(InputString  
 G-Set-Input-Mode-(Locator  
 G-Set-Input-Mode-(Stroke  
 G-Set-Input-Mode-(Valuator  
 G-Set-Input-Mode-(Choice  
 G-Set-Input-Mode-(Pick  
 G-Set-Input-Mode-(InputString  
 G-Req-Input-(Locator  
 G-Req-Input-(Stroke  
 G-Req-Input-(Valuator  
 G-Req-Input-(Choice  
 G-Req-Input-(Pick  
 G-Req-Input-(InputString  
 G-Sample-Input-(Locator  
 G-Sample-Input-(Stroke  
 G-Sample-Input-(Valuator  
 G-Sample-Input-(Choice  
 G-Sample-Input-(Pick  
 G-Sample-Input-(InputString  
 G-Await-Event  
 G-Flush-Device-Events  
 G-Get-Input-(Locator  
 G-Get-Input-(Stroke  
 G-Get-Input-(Valuator  
 G-Get-Input-(Choice  
 G-Get-Input-(Pick  
 G-Get-Input-(InputString  
 G-Write-Item  
 G-GetItemType  
 G-ReadItem  
 G-Interpret-Item  
 G-Inq-Op-St  
 G-Inq-Level-GKS  
 G-Inq-List-Ws-Types  
 G-Inq-Ws-Max-Num  
 G-Inq-Max-Norm-Tran-Num  
 G-Inq-Open-Ws  
 G-Inq-Active-Ws  
 G-Inq-Cur-Prim-Attr  
 G-Inq-Cur-Pick-Id  
 G-Inq-Cur-Indiv-Attr  
 G-Inq-Cur-Norm-Tran-Num  
 G-Inq-List-Norm-Tran-Num  
 G-Inq-Norm-Tran  
 G-Inq-Clip  
 G-Inq-Name-Open-Seg  
 G-Inq-Seg-Names  
 G-Inq-More-Events  
 G-Inq-Ws-Conn-Type  
 G-Inq-Ws-St  
 G-Inq-Ws-Defer-Upd-St  
 G-Inq-List-Prim-Ind-(Polyline  
 G-Inq-List-Prim-Ind-(Polymarker  
 G-Inq-List-Prim-Ind-(GKSText  
 G-Inq-List-Prim-Ind-(Fill  
 G-Inq-Prim-Rep-(Polyline

## The Pascal language binding of GKS

## Implementation of the two interfaces

|                                                          |                                    |
|----------------------------------------------------------|------------------------------------|
| INQUIRE POLYMARKER REPRESENTATION                        | G-Inq-Prim-Rep-(Polymarker         |
| INQUIRE TEXT REPRESENTATION                              | G-Inq-Prim-Rep-(GKSText            |
| INQUIRE FILL AREA REPRESENTATION                         | G-Inq-Prim-Rep-(Fill               |
| INQUIRE TEXT EXTENT                                      | G-Inq-Text-Extent                  |
| INQUIRE TEXT EXTENT (Pascal level 0)                     | G-Inq-TextString-Extent            |
| INQUIRE LIST OF PATTERN INDICES                          | G-Inq-List-Pattern-Ind             |
| INQUIRE PATTERN REPRESENTATION                           | G-Inq-Pattern-Rep                  |
| INQUIRE LIST OF COLOUR INDICES                           | G-Inq-List-Col-Ind                 |
| INQUIRE COLOUR REPRESENTATION                            | G-Inq-Col-Rep                      |
| INQUIRE WORKSTATION TRANSFORMATION                       | G-Inq-Ws-Tran                      |
| INQUIRE SET OF SEGMENT NAMES ON WORKSTATION              | G-Inq-Seg-Names-Ws                 |
| INQUIRE LOCATOR DEVICE STATE                             | G-Inq-Input-Device-St-(Locator     |
| INQUIRE STROKE DEVICE STATE                              | G-Inq-Input-Device-St-(Stroke      |
| INQUIRE VALUATOR DEVICE STATE                            | G-Inq-Input-Device-St-(Valuator    |
| INQUIRE CHOICE DEVICE STATE                              | G-Inq-Input-Device-St-(Choice      |
| INQUIRE PICK DEVICE STATE                                | G-Inq-Input-Device-St-(Pick        |
| INQUIRE STRING DEVICE STATE                              | G-Inq-Input-Device-St-(InputString |
| INQUIRE WORKSTATION CATEGORY                             | G-Inq-Ws-Category                  |
| INQUIRE WORKSTATION CLASSIFICATION                       | G-Inq-Ws-Class                     |
| INQUIRE DISPLAY SPACE SIZE                               | G-Inq-Display-Size                 |
| INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES   | G-Inq-Dyn-Mod-Ws-Attr              |
| INQUIRE DEFAULT DEFERRAL STATE VALUES                    | G-Inq-Def-Defer-St                 |
| INQUIRE POLYLINE FACILITIES                              | G-Inq-Prim-Facil-(Polyline         |
| INQUIRE POLYMARKER FACILITIES                            | G-Inq-Prim-Facil-(Polymarker       |
| INQUIRE TEXT FACILITIES                                  | G-Inq-Prim-Facil-(GKSText          |
| INQUIRE FILL AREA FACILITIES                             | G-Inq-Prim-Facil-(Fill             |
| INQUIRE PREDEFINED POLYLINE REPRESENTATION               | G-Inq-Pred-Prim-Rep-(Polyline      |
| INQUIRE PREDEFINED POLYMARKER REPRESENTATION             | G-Inq-Pred-Prim-Rep-(Polymarker    |
| INQUIRE PREDEFINED TEXT REPRESENTATION                   | G-Inq-Pred-Prim-Rep-(GKSText       |
| INQUIRE PREDEFINED FILL AREA REPRESENTATION              | G-Inq-Pred-Prim-Rep-(Fill          |
| INQUIRE PATTERN FACILITIES                               | G-Inq-Pattern-Facil                |
| INQUIRE PREDEFINED PATTERN REPRESENTATION                | G-Inq-Pred-Pattern-Rep             |
| INQUIRE COLOUR FACILITIES                                | G-Inq-Col-Facil                    |
| INQUIRE PREDEFINED COLOUR REPRESENTATION                 | G-Inq-Pred-Col-Rep                 |
| INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES | G-Inq-List-gdp                     |
| INQUIRE GENERALIZED DRAWING PRIMITIVE                    | G-Inq-gdp                          |
| INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES       | G-Inq-Max-Ws-St                    |
| INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED           | G-Inq-Num-Seg-Priorities           |
| INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES       | G-Inq-Dyn-Mod-Seg-Attr             |
| INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES        | G-Inq-Num-Input-Devices            |
| INQUIRE DEFAULT LOCATOR DEVICE DATA                      | G-Inq-Def-Input-Data-(Locator      |
| INQUIRE DEFAULT STROKE DEVICE DATA                       | G-Inq-Def-Input-Data-(Stroke       |
| INQUIRE DEFAULT VALUATOR DEVICE DATA                     | G-Inq-Def-Input-Data-(Valuator     |
| INQUIRE DEFAULT CHOICE DEVICE DATA                       | G-Inq-Def-Input-Data-(Choice       |
| INQUIRE DEFAULT PICK DEVICE DATA                         | G-Inq-Def-Input-Data-(Pick         |
| INQUIRE DEFAULT STRING DEVICE DATA                       | G-Inq-Def-Input-Data-(String       |
| INQUIRE SET OF ASSOCIATED WORKSTATIONS                   | G-Inq-Assoc-Ws                     |
| INQUIRE SEGMENT ATTRIBUTES                               | G-Inq-Seg-Attr                     |
| INQUIRE PIXEL ARRAY DIMENSIONS                           | G-Inq-Pixel-Array-Dim              |
| INQUIRE PIXEL ARRAY                                      | G-Inq-Pixel-Array                  |
| INQUIRE PIXEL                                            | G-Inq-Pixel                        |
| INQUIRE INPUT QUEUE OVERFLOW                             | G-Inq-Input-Overflow               |
| EVALUATE TRANSFORMATION MATRIX                           | G-Eval-Tran                        |
| ACCUMULATE TRANSFORMATION MATRIX                         | G-Accum-Tran                       |
| EMERGENCY CLOSE GKS                                      | G-Emergency-Close-GKS              |
| ERROR HANDLING                                           | G-Error-Handling                   |
| ERROR LOGGING                                            | G-Error-Logging                    |

## Implementation of the two interfaces

## The Pascal language binding of GKS

Table 3 - GKS function names and Pascal names ordered alphabetically

|                                                          |                                |
|----------------------------------------------------------|--------------------------------|
| ACCUMULATE TRANSFORMATION MATRIX                         | G-Accum-Tran                   |
| ACTIVATE WORKSTATION                                     | G-Activate-Ws                  |
| ASSOCIATE SEGMENT WITH WORKSTATION                       | G-Assoc-Seg-Ws                 |
| AWAIT EVENT                                              | G-Await-Event                  |
| CELL ARRAY                                               | G-CellArray                    |
| CLEAR WORKSTATION                                        | G-Clear-Ws                     |
| CLOSE GKS                                                | G-Close-GKS                    |
| CLOSE SEGMENT                                            | G-Close-Seg                    |
| CLOSE WORKSTATION                                        | G-Close-Ws                     |
| COPY SEGMENT TO WORKSTATION                              | G-Copy-Seg-Ws                  |
| CREATE SEGMENT                                           | G-Create-Seg                   |
| DEACTIVATE WORKSTATION                                   | G-Deactivate-Ws                |
| DELETE SEGMENT FROM WORKSTATION                          | G-Del-Seg-Ws                   |
| DELETE SEGMENT                                           | G-Del-Seg                      |
| EMERGENCY CLOSE GKS                                      | G-Emergency-Close-GKS          |
| ERROR HANDLING                                           | G-Error-Handling               |
| ERROR LOGGING                                            | G-Error-Logging                |
| ESCAPE                                                   | G-Escape                       |
| EVALUATE TRANSFORMATION MATRIX                           | G-Eval-Tran                    |
| FILL AREA                                                | G-Fill                         |
| FLUSH DEVICE EVENTS                                      | G-Flush-Device-Events          |
| GENERALIZED DRAWING PRIMITIVE                            | G-gdp                          |
| GET CHOICE                                               | G-Get-Input-(Choice            |
| GET ITEM TYPE FROM GKSM                                  | G-GetItemType                  |
| GET LOCATOR                                              | G-Get-Input-(Locator           |
| GET PICK                                                 | G-Get-Input-(Pick              |
| GET STRING                                               | G-Get-Input-(InputString       |
| GET STROKE                                               | G-Get-Input-(Stroke            |
| GET VALUATOR                                             | G-Get-Input-(Valuator          |
| INITIALISE CHOICE                                        | G-Init-Input-(Choice           |
| INITIALISE LOCATOR                                       | G-Init-Input-(Locator          |
| INITIALISE PICK                                          | G-Init-Input-(Pick             |
| INITIALISE STRING                                        | G-Init-Input-(InputString      |
| INITIALISE STROKE                                        | G-Init-Input-(Stroke           |
| INITIALISE VALUATOR                                      | G-Init-Input-(Valuator         |
| INQUIRE CHOICE DEVICE STATE                              | G-Inq-Input-Device-St-(Choice  |
| INQUIRE CLIPPING                                         | G-Inq-Clip                     |
| INQUIRE COLOUR FACILITIES                                | G-Inq-Col-Facil                |
| INQUIRE COLOUR REPRESENTATION                            | G-Inq-Col-Rep                  |
| INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES              | G-Inq-Cur-Indiv-Attr           |
| INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER      | G-Inq-Cur-Norm-Tran-Num        |
| INQUIRE CURRENT PICK IDENTIFIER                          | G-Inq-Cur-Pick-Id              |
| INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES               | G-Inq-Cur-Prim-Attr            |
| INQUIRE DEFAULT CHOICE DEVICE DATA                       | G-Inq-Def-Input-Data-(Choice   |
| INQUIRE DEFAULT DEFERRAL STATE VALUES                    | G-Inq-Def-Defer-St             |
| INQUIRE DEFAULT LOCATOR DEVICE DATA                      | G-Inq-Def-Input-Data-(Locator  |
| INQUIRE DEFAULT PICK DEVICE DATA                         | G-Inq-Def-Input-Data-(Pick     |
| INQUIRE DEFAULT STRING DEVICE DATA                       | G-Inq-Def-Input-Data-(String   |
| INQUIRE DEFAULT STROKE DEVICE DATA                       | G-Inq-Def-Input-Data-(Stroke   |
| INQUIRE DEFAULT VALUATOR DEVICE DATA                     | G-Inq-Def-Input-Data-(Valuator |
| INQUIRE DISPLAY SPACE SIZE                               | G-Inq-Display-Size             |
| INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES       | G-Inq-Dyn-Mod-Seg-Attr         |
| INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES   | G-Inq-Dyn-Mod-Ws-Attr          |
| INQUIRE FILL AREA FACILITIES                             | G-Inq-Prim-Facil-(Fill         |
| INQUIRE FILL AREA REPRESENTATION                         | G-Inq-Prim-Rep-(Fill           |
| INQUIRE GENERALIZED DRAWING PRIMITIVE                    | G-Inq-gdp                      |
| INQUIRE INPUT QUEUE OVERFLOW                             | G-Inq-Input-Overflow           |
| INQUIRE LEVEL OF GKS                                     | G-Inq-Level-GKS                |
| INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES | G-Inq-List-gdp                 |
| INQUIRE LIST OF AVAILABLE WORKSTATION TYPES              | G-Inq-List-Ws-Types            |
| INQUIRE LIST OF COLOUR INDICES                           | G-Inq-List-Col-Ind             |
| INQUIRE LIST OF FILL AREA INDICES                        | G-Inq-List-Ind-(Fill           |

[ISO page 10]

## The Pascal language binding of GKS

## Implementation of the two interfaces

|                                                      |                                    |
|------------------------------------------------------|------------------------------------|
| INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS | G-Inq-List-Norm-Tran-Num           |
| INQUIRE LIST OF PATTERN INDICES                      | G-Inq-List-Pattern-Ind             |
| INQUIRE LIST OF POLYLINE INDICES                     | G-Inq-List-Ind-(Polyline           |
| INQUIRE LIST OF POLYMARKER INDICES                   | G-Inq-List-Ind-(Polymarker         |
| INQUIRE LIST OF TEXT INDICES                         | G-Inq-List-Ind-(GKSText            |
| INQUIRE LOCATOR DEVICE STATE                         | G-Inq-Input-Device-St-(Locator     |
| INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES   | G-Inq-Max-Ws-St                    |
| INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER  | G-Inq-Max-Norm-Tran-Num            |
| INQUIRE MORE SIMULTANEOUS EVENTS                     | G-Inq-More-Events                  |
| INQUIRE NAME OF OPEN SEGMENT                         | G-Inq-Name-Open-Seg                |
| INQUIRE NORMALIZATION TRANSFORMATION                 | G-Inq-Norm-Tran                    |
| INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES    | G-Inq-Num-Input-Devices            |
| INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED       | G-Inq-Num-Seg-Priorities           |
| INQUIRE OPERATING STATE VALUE                        | G-Inq-Op-St                        |
| INQUIRE PATTERN FACILITIES                           | G-Inq-Pattern-Facil                |
| INQUIRE PATTERN REPRESENTATION                       | G-Inq-Pattern-Rep                  |
| INQUIRE PICK DEVICE STATE                            | G-Inq-Input-Device-St-(Pick        |
| INQUIRE PIXEL ARRAY DIMENSIONS                       | G-Inq-Pixel-Array-Dim              |
| INQUIRE PIXEL ARRAY                                  | G-Inq-Pixel-Array                  |
| INQUIRE PIXEL                                        | G-Inq-Pixel                        |
| INQUIRE POLYLINE FACILITIES                          | G-Inq-Prim-Facil-(Polyline         |
| INQUIRE POLYLINE REPRESENTATION                      | G-Inq-Prim-Rep-(Polyline           |
| INQUIRE POLYMARKER FACILITIES                        | G-Inq-Prim-Facil-(Polymarker       |
| INQUIRE POLYMARKER REPRESENTATION                    | G-Inq-Prim-Rep-(Polymarker         |
| INQUIRE PREDEFINED COLOUR REPRESENTATION             | G-Inq-Pred-Col-Rep                 |
| INQUIRE PREDEFINED FILL AREA REPRESENTATION          | G-Inq-Pred-Prim-Rep-(Fill          |
| INQUIRE PREDEFINED PATTERN REPRESENTATION            | G-Inq-Pred-Pattern-Rep             |
| INQUIRE PREDEFINED POLYLINE REPRESENTATION           | G-Inq-Pred-Prim-Rep-(Polyline      |
| INQUIRE PREDEFINED POLYMARKER REPRESENTATION         | G-Inq-Pred-Prim-Rep-(Polymarker    |
| INQUIRE PREDEFINED TEXT REPRESENTATION               | G-Inq-Pred-Prim-Rep-(GKSText       |
| INQUIRE SEGMENT ATTRIBUTES                           | G-Inq-Seg-Attr                     |
| INQUIRE SET OF ACTIVE WORKSTATIONS                   | G-Inq-Active-Ws                    |
| INQUIRE SET OF ASSOCIATED WORKSTATIONS               | G-Inq-Assoc-Ws                     |
| INQUIRE SET OF OPEN WORKSTATIONS                     | G-Inq-Open-Ws                      |
| INQUIRE SET OF SEGMENT NAMES IN USE                  | G-Inq-Seg-Names                    |
| INQUIRE SET OF SEGMENT NAMES ON WORKSTATION          | G-Inq-Seg-Names-Ws                 |
| INQUIRE STRING DEVICE STATE                          | G-Inq-Input-Device-St-(InputString |
| INQUIRE STROKE DEVICE STATE                          | G-Inq-Input-Device-St-(Stroke      |
| INQUIRE TEXT EXTENT LEVEL 0 PASCAL                   | G-Inq-TextString-Extent            |
| INQUIRE TEXT EXTENT                                  | G-Inq-Text-Extent                  |
| INQUIRE TEXT FACILITIES                              | G-Inq-Prim-Facil-(GKSText          |
| INQUIRE TEXT REPRESENTATION                          | G-Inq-Prim-Rep-(GKSText            |
| INQUIRE VALUATOR DEVICE STATE                        | G-Inq-Input-Device-St-(Valuator    |
| INQUIRE WORKSTATION CATEGORY                         | G-Inq-Ws-Category                  |
| INQUIRE WORKSTATION CLASSIFICATION                   | G-Inq-Ws-Class                     |
| INQUIRE WORKSTATION CONNECTION AND TYPE              | G-Inq-Ws-Conn-Type                 |
| INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES       | G-Inq-Ws-Defer-Upd-St              |
| INQUIRE WORKSTATION MAXIMUM NUMBERS                  | G-Inq-Ws-Max-Num                   |
| INQUIRE WORKSTATION STATE                            | G-Inq-Ws-St                        |
| INQUIRE WORKSTATION TRANSFORMATION                   | G-Inq-Ws-Tran                      |
| INSERT SEGMENT                                       | G-Insert-Seg                       |
| INTERPRET ITEM                                       | G-Interpret-Item                   |
| MESSAGE LEVEL 0 PASCAL                               | G-MessageString                    |
| MESSAGE                                              | G-Message                          |
| OPEN GKS                                             | G-Open-GKS                         |
| OPEN WORKSTATION                                     | G-Open-Ws                          |
| POLYLINE                                             | G-Polyline                         |
| POLYMARKER                                           | G-Polymarker                       |
| READ ITEM FROM GKSM                                  | G-ReadItem                         |
| REDRAW ALL SEGMENTS ON WORKSTATION                   | G-Redraw-Seg-Ws                    |
| RENAME SEGMENT                                       | G-Rename-Seg                       |
| REQUEST CHOICE                                       | G-Req-Input-(Choice                |
| REQUEST LOCATOR                                      | G-Req-Input-(Locator               |
| REQUEST PICK                                         | G-Req-Input-(Pick                  |
| REQUEST STRING                                       | G-Req-Input-(InputString           |
| REQUEST STROKE                                       | G-Req-Input-(Stroke                |

[ISO page 11]  
Documents

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page Hub

## Implementation of the two interfaces

## The Pascal language binding of GKS

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  
 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 INPUT PRIORITY  
 SET VIEWPORT  
 SET VISIBILITY  
 SET WINDOW  
 SET WORKSTATION VIEWPORT  
 SET WORKSTATION WINDOW  
 TEXT LEVEL 0 PASCAL  
 TEXT  
 UPDATE WORKSTATION  
 WRITE ITEM TO GKSM

G-Req-Input-(Valuator  
 G-Sample-Input-(Choice  
 G-Sample-Input-(Locator  
 G-Sample-Input-(Pick  
 G-Sample-Input-(InputString  
 G-Sample-Input-(Stroke  
 G-Sample-Input-(Valuator  
 G-Select-Norm-Tran  
 G-Set-ASF  
 G-Set-Char-Expan  
 G-Set-Char-Height  
 G-Set-Char-Spacing  
 G-Set-Char-Up-Vector  
 G-Set-Input-Mode-(Choice  
 G-Set-Clip  
 G-Set-Col-Rep  
 G-Set-Defer-St  
 G-Set-Det  
 G-Set-Fill-Col-Ind  
 G-Set-Prim-Ind-(Fill  
 G-Set-Fill-Int-Style  
 G-Set-Prim-Rep-(Fill  
 G-Set-Fill-Style-Ind  
 G-Set-Highlight  
 G-Set-LineType  
 G-Set-LineWidth-Scale  
 G-Set-Input-Mode-(Locator  
 G-Set-Marker-Size-Scale  
 G-Set-Marker-Type  
 G-Set-Pattern-Ref-Point  
 G-Set-Pattern-Rep  
 G-Set-Pattern-Size  
 G-Set-Pick-Id  
 G-Set-Input-Mode-(Pick  
 G-Set-Line-Col-Ind  
 G-Set-Prim-Ind-(Polyline  
 G-Set-Prim-Rep-(Polyline  
 G-Set-Marker-Col-Ind  
 G-Set-Prim-Ind-(Polymarker  
 G-Set-Prim-Rep-(Polymarker  
 G-Set-Seg-Priority  
 G-Set-Seg-Tran  
 G-Set-Input-Mode-(InputString  
 G-Set-Input-Mode-(Stroke  
 G-Set-Text-Align  
 G-Set-Text-Col-Ind  
 G-Set-Text-Font-Prec  
 G-Set-Prim-Ind-(GKSText  
 G-Set-Text-Path  
 G-Set-Prim-Rep-(GKSText  
 G-Set-Input-Mode-(Valuator  
 G-Set-Viewport-Priority  
 G-Set-Viewport  
 G-Set-Vis  
 G-Set-Window  
 G-Set-Ws-Viewport  
 G-Set-Ws-Window  
 G-TextString  
 G-Text  
 G-Upd-Ws  
 G-Write-Item

[ISO page 12]

**The Pascal language binding of GKS****Implementation of the two interfaces****3.6 Representation of GKS data types**

Clause 6 of ISO 7942 defines GKS data structures which are an integral part of the system. These data structures are represented by the following Pascal data structures.

- a) The GKS types integer and real are mapped to Pascal integers and reals.
- b) The GKS enumeration types are mapped to Pascal enumerated types, often with the addition of sentinel words or characters to avoid name clashes. For example, the GKS type (HIGHER,LOWER) in the SET VIEWPORT INPUT PRIORITY function is in Pascal the type GTpriority with elements (HigherPriority,LowerPriority).
- c) String is represented as a packed conformant array of characters when the string is a parameter to a Pascal procedure, or as a fixed length string (GTstring) when it is a component of another data structure. In Pascal Level 0, string is mapped to be the fixed length type.
- d) A GKS point is mapped to the type GRPoint in Pascal, which is a record with x,y( : real) fields. Conformant arrays of GRPoint are used in Level 1 procedure definitions, and fixed length array of GRPoint (GAPointArray) in Level 0.
- e) Sets in GKS are represented as linear lists in Pascal when the cardinality of the set is potentially infinite (for example, the set of workstation identifiers). Otherwise GKS sets are Pascal sets - the only two being the set of primitives associated with a Generalized Drawing Primitive, and the set of fill area interior styles.
- f) The GKS Data Records are represented as implementation defined records. An empty data record is specified by setting the Boolean field (empty) as TRUE.
- g) More complex data structures are represented as Pascal records and linear lists. The definition of all the data structures is given in clause 6.

**3.7 Naming conventions for data types**

A consistent naming scheme for data types is used.

- a) Constants are prefixed with "GC".
- b) Enumerated and scalar types are prefixed with "GT".
- c) Arrays are prefixed with "GA".
- d) Records are prefixed with "GR".
- e) Pointers are prefixed with "GPT".
- f) Sets are prefixed with "GS".

Mixtures of upper and lower case letters are used extensively in the type and procedure names. This is to aid readability and is not otherwise significant.

**3.8 Implementation dependent characteristics**

There are a number of implementation defined characteristics necessary in a Pascal interface. Resolutions of such implementation dependent features shall be detailed in the documentation of a Pascal implementation.

- a) The error reporting file name specified by GKS as a parameter to the Open GKS function is represented as a string in Pascal. Many ISO Pascal compilers provide an extended definition of the Pascal "reset" procedure to allow an external file to be associated with a Pascal file variable. It is expected that this extension would be employed by an implementor. If such an extension is not available, the implementation provides an alternative method for associating a file with a Pascal file variable.
- b) Similarly, the workstation connection identifier, as specified by GKS as a parameter to the Open Workstation function is also represented in Pascal as a string. The same considerations apply as in a).
- c) There are number of implementation defined constants, which are shown in 6.1.
- d) The GKS Data Record is represented as a number of implementation defined types depending on the purpose of the records. GRLocatorData, GRStrokeData, GRValuatorData, GRChoiceData, GRPickData and GRStringData are used in the Initialise Input functions and the corresponding inquiry functions. GREscapeData is for use with the Escape function. GRGdpData is for use with the Generalized Drawing Primitive, and GRFileData is for use with the metafile functions. The implementation documentation shall detail the structure of these records, and the implementation may provide procedures for

## Implementation dependent characteristics

## The Pascal language binding of GKS

constructing and manipulating such records.

e) ISO 7185 does not prohibit separate compilation, but does not give a standard manner for achieving this. Separate compilation is crucial for GKS, since an implementation is not feasible unless the underlying Pascal supports separate compilation. Implementation documentation shall detail how this is to be achieved.

f) The Open GKS function (GOpenGKS) has a parameter "amount of memory units for buffer area". An implementation shall document the meaning and usage of this parameter. The value -1 is the default value, which is given as the constant GCdefmemory.

g) The Pascal binding defines a large number of data types. The implementation documentation shall specify how these types are to be imported into the application environment.

### 3.9 Level of Pascal

ISO 7185 has two levels of conformance. Level 0 Pascal does not have conformant array parameters whereas Level 1 does.

This Pascal interface to GKS may be used under Level 0 or Level 1 Pascal, with the conformant array parameters replaced by parameters of the appropriate fixed length array type. All of the fixed length types needed are defined in this document. The names of the procedures are the same under Level 0 or Level 1 Pascal. However, those which use the fixed length array types are headed in clause 5 as "Pascal Level 0".

In the case of procedures involving strings, an efficient Level 0 representation requires that the length of the actual string parameter be given. To cater for this, duplicate procedures for strings are provided which have this additional (length of string) parameter. These additional procedures which use fixed length strings are recognised by the word "String" forming part of the procedure name, which is otherwise the same name as for the procedure using conformant arrays. For example, GInqTextExtent uses a conformant array parameter, whereas GInqTextStringExtent uses a fixed length string parameter and a length of string parameter. These procedures are additionally headed "Pascal Level 0". A Pascal implementation shall provide all of the procedures and data types defined in clauses 5 to 7 of this document, including those marked as "Pascal Level 0".

An implementation which does not use conformant arrays, that is, which is based on a Level 0 Pascal, is called a Level 0 implementation. Likewise, a Level 1 implementation is one which makes use of conformant arrays. An applications program written for a Level 0 Pascal implementation also works on a Level 1 Pascal implementation. In addition an applications program written for a Level 1 Pascal implementation works under Level 0 provided that the following rules are adopted:

- The fixed length arrays provided in the binding are used in the application. For example, in calls to GPolyline, the type GPointArray is used.
- The size of the arrays do not exceed the maximum sizes as specified by the constants (for example, GCmaxpoint for arrays of points).
- The Level 0 string procedures are used, GMessageString, GTextString, and GInqTextStringExtent.

Some examples are given to clarify these points:

#### Example 1

This only works under Level 1.

```
var p : array[1..MAX] of GPoint;
    n : GTint2;
GPolyline(n,p);
```

[ISO page 14]

**The Pascal language binding of GKS****Level of Pascal****Example 2**

This works under both levels.

```
var p : GPointArray;  
    n : GTint2;  
GPolyline(n,p);
```

**Example 3**

This works under both levels.

```
var status : GTstatus;  
    return : GRInput;  
GReqInput(Stroke,DISPLAY,1,status,return);  
if status = StatusOK then  
with return do GFill(num,points);
```

**Example 4**

This works only under Level 1.

```
var status : GTstatus;  
    return : GRInput;  
GReqInput(Locator,DISPLAY,1,status,return);  
if status = StatusOK then  
with return do GText(position,'Hallo world');
```

**Example 5**

This works under both levels.

```
var LocStatus,StrStatus : GTstatus;  
    LocReturn,StrReturn : GRInput;  
GReqInput(InputString,DISPLAY,1,StrStatus,StrReturn);  
GReqInput(Locator,DISPLAY,1,LocStatus,LocReturn);  
if((LocStatus = StatusOK) and (StrStatus = StatusOK)) then  
with StrReturn do  
with LocReturn do  
GTextString(position,stringlength,charstring);
```

## 4 Error handling

### 4.1 The error handling function

Error handling shall be carried out as described in 4.12 of ISO 7942. The implementation documentation shall detail the method for an application program to provide an Error Handling procedure. The default Error Handling procedure shall be available.

### 4.2 Pascal specific GKS errors

Error 2100 of the GKS error list (Annex B of ISO 7942) is allocated to the Pascal language binding:

*2100 There is an incompatibility between the bounds of the array and the actual size parameters specified*

This error is invoked when the "number of points" parameter (numpoints) is not compatible with the bounds of a conformant array. For example, in GPolyline, numpoints must not be greater than (max-min + 1). This error is also invoked in a level 0 procedure when the specified number of points exceeds GCmaxpoint.

*2101 The number of characters in the string exceeds the maximum allowed number of characters*

*2102 The supplied array is too small to store the required data*

The Error Handling and Error Logging procedures each have a parameter "identification of the GKS procedure which caused the error detection". This is represented in Pascal as a string parameter. The values of the strings identifying Pascal GKS procedures are those Pascal names given in Table 2 or Table 3.

## 5 GKS functions

### 5.1 Notational conventions

The GKS abstract function names are given followed by the corresponding Pascal procedure and associated parameters. The GKS Level is shown for each function.

### 5.2 Control functions

#### OPEN GKS

L0a

```
procedure GOpenGKS(  
    errorfile      : GAFileName;  
    memoryunits    : INTEGER  
);
```

An implementation dependent name for Pascal Standard Output shall be provided.

#### CLOSE GKS

L0a

```
procedure GCloseGKS;
```

#### OPEN WORKSTATION

L0a

```
procedure GOpenWs(  
    wsid           : GTWsId;  
    connid         : GAFileName;  
    wstype         : GTWsType  
);
```

#### CLOSE WORKSTATION

L0a

```
procedure GCloseWs(  
    wsid           : GTWsId  
);
```

#### ACTIVATE WORKSTATION

L0a

```
procedure GActivateWs(  
    wsid           : GTWsId  
);
```

#### DEACTIVATE WORKSTATION

L0a

```
procedure GDeactivateWs(  
    wsid           : GTWsId  
);
```

DD 114: Part 2: 1986

**Control functions****GKS functions****CLEAR WORKSTATION**

L0a

```

procedure GClearWs(
    wsid          : GTWsId;
    controlflag   : GTcontrol
);

```

**REDRAW ALL SEGMENTS ON WORKSTATION**

L1a

```

procedure GRedrawSegWs(
    wsid          : GTWsId
);

```

**UPDATE WORKSTATION**

L0a

```

procedure GUpdWs(
    wsid          : GTWsId;
    regenflag     : GTUpdRegen
);

```

**SET DEFERRAL STATE**

L1a

```

procedure GSetDeferSt(
    wsid          : GTWsId;
    defermode     : GTdefer;
    regenmode     : GTImplicitRegen
);

```

**MESSAGE**

L1a

*Pascal level 1*

```

procedure GMessage(
    wsid          : GTWsId;
    message       : packed array[min..max : GTint1] of CHAR
);

```

*Pascal level 0*

```

procedure GMessageString(
    wsid          : GTWsId;
    stringlength  : GTint0;
    message       : GAstring
);

```

The parameter stringlength shall be  $\leq$  GCmaxstr.

**Errors:**

2101 The number of characters in the string exceeds the maximum allowed number of characters

[ISO page 18]

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1

**Documents**

Document Sharing Hub

## GKS functions

## Control functions

## ESCAPE

L0a

```

procedure GEscape(
    escapeid      : GTEscapeId;
    inputdatarec  : GREscapeData;
    VAR outputdatarec : GREscapeData
);

```

## 5.3 Output functions

## POLYLINE

L0a

*Pascal level 1*

```

procedure GPolyline(
    numpoints      : GTint2;
    VAR points     : array[min..max : INTEGER] of GRpoint
);

```

The parameter  $\text{numpoints} \leq \text{max-min}+1$  is the number of points in the array. Although the array is declared as VAR the implementation shall not change the values of the array elements.

Errors:

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

*Pascal level 0*

```

procedure GPolyline(
    numpoints      : GTint2;
    VAR points     : GPointArray
);

```

The parameter  $\text{numpoints} \leq \text{GCMAXPOINT}$  is the number of points in the array. Although the array is declared as VAR the implementation shall not change the values of the array elements.

Errors:

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

## POLYMARKER

L0a

*Pascal level 1*

```

procedure GPolymarker(
    numpoints      : GTint1;
    VAR points     : array[min..max : INTEGER] of GRpoint
);

```

The parameter  $\text{numpoints} \leq \text{max-min}+1$  is the number of points in the array. Although the array is declared as VAR the implementation shall not change the values of the array elements.

Errors:

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

**Output functions****GKS functions***Pascal level 0*

```

procedure GPolymarker(
    numpoints      : GTint2;
    VAR points     : GAPointArray
);

```

The parameter  $\text{numpoints} \leq \text{GCmaxpoint}$  is the number of points in the array. Although the array is declared as VAR the implementation shall not change the values of the array elements.

**Errors:**

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

**TEXT****L0a***Pascal level 1*

```

procedure GText(
    textposition    : GRpoint;
    charstring      : packed array[min..max : GTint1] of CHAR
);

```

*Pascal level 0*

```

procedure GTextString(
    textposition    : GRpoint;
    stringlength    : GTint0;
    charstring      : GAstring
);

```

The parameter  $\text{stringlength}$  shall be  $\leq \text{GCmaxstring}$ .

**Errors:**

2101 *The number of characters in the string exceeds the maximum allowed number of characters*

**FILL AREA****L0a***Pascal level 1*

```

procedure GFill(
    numpoints      : GTint3;
    VAR points     : array[min..max : INTEGER] of GRpoint
);

```

The parameter  $\text{numpoints} \leq \text{max} - \text{min} + 1$  is the number of points in the array. Although the array is declared as VAR the implementation shall not change the values of the array elements.

**Errors:**

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

*Pascal level 0*

```

procedure GFill(
    numpoints      : GTint3;
    VAR points     : GAPointArray
);

```

The parameter  $\text{numpoints} \leq \text{GCmaxpoint}$  is the number of points in the array. Although the array is declared as VAR the implementation shall not change the values of the array elements.

[ISO page 20]

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1.

**GKS functions****Output functions****Errors:**

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

**CELL ARRAY****L0a***Pascal level 1*

```

procedure GCellArray(
    rectpoint1,
    rectpoint2      : GRpoint;
    dx,dy           : GTint1;
    VAR Col         : array[min1..max1 : INTEGER; min2..max2 : INTEGER] of GTint0
);

```

The cell array bounds are min1 to min1+dx-1 ( $\leq$ max1) and min2 to min2+dy-1 ( $\leq$ max2). Although the array is declared as VAR the implementation shall not change the values of the array elements.

**Errors:**

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

*Pascal level 0*

```

procedure GCellArray(
    rectpoint1,
    rectpoint2      : GRPoint;
    dx,dy           : GTint1;
    VAR Col         : GAColArray
);

```

The parameters dx and dy satisfy  $dx \leq GCmaxDX$  and  $dy \leq GCmaxDY$ . Although the array is declared as VAR the implementation shall not change the values of the array elements.

**Errors:**

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

**GENERALIZED DRAWING PRIMITIVE (GDP)****L0a***Pascal level 1*

```

procedure Ggdp(
    numpoints      : GTint0;
    VAR points     : array[min..max : INTEGER] of GRpoint;
    gdpid          : GTgdpId;
    datarec        : GRgdpData
);

```

The parameter numpoints  $\leq$  max-min+1 is the number of points in the array. Although the array is declared as VAR the implementation shall not change the values of the array elements.

**Errors:**

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

*Pascal level 0*

```

procedure Ggdp(
    numpoints      : GTint0;
    VAR points     : GAPointArray;
    gdpid          : GTgdpId;
    datarec        : GRgdpData
);

```

[ISO page 21]

**Documents**

DD 114: Part 2: 1986

**Output functions****GKS functions**

The parameter  $\text{numpoints} \leq \text{GCmaxpoint}$  is the number of points in the array. Although the array is declared as VAR the implementation shall not change the values of the array elements.

**Errors:**

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

**5.4 Output attributes****5.4.1 Workstation independent primitive attributes****SET POLYLINE INDEX**

L0a

**SET POLYMARKER INDEX**

L0a

**SET TEXT INDEX**

L0a

**SET FILL AREA INDEX**

L0a

```
procedure GSetPrimInd(
```

```
    prim          : GTPrim;
    primindind    : GTint1
);
```

This procedure is a general

SET <Primitive> INDEX

By substituting for the components of the type GTPrim, the GKS abstract functions are obtained.

**SET LINETYPE**

L0a

```
procedure GSetLineType(
```

```
    linetype      : INTEGER
);
```

**SET LINEWIDTH SCALE FACTOR**

L0a

```
procedure GSetLineWidthScale(
```

```
    linewidthscale : REAL
);
```

**SET POLYLINE COLOUR INDEX**

L0a

```
procedure GSetLineColInd(
```

```
    linecolind     : GTint0
);
```

**SET MARKER TYPE**

L0a

```
procedure GSetMarkerType(
```

```
    markertype     : INTEGER
);
```

[ISO page 22]

**GKS functions****Output attributes****SET MARKER SIZE SCALE FACTOR****L0a**

```
procedure GSetMarkerSizeScale(  
    markersizescale : REAL  
);
```

**SET POLYMARKER COLOUR INDEX****L0a**

```
procedure GSetMarkerColInd(  
    markercolind : GTint0  
);
```

**SET TEXT FONT AND PRECISION****L0a**

```
procedure GSetTextFontPrec(  
    textfontprec : GRFontPrec  
);
```

**SET CHARACTER EXPANSION FACTOR****L0a**

```
procedure GSetCharExpan(  
    charexpan : REAL  
);
```

**SET CHARACTER SPACING****L0a**

```
procedure GSetCharSpacing(  
    charspacing : REAL  
);
```

**SET TEXT COLOUR INDEX****L0a**

```
procedure GSetTextColInd(  
    textcolind : GTint0  
);
```

**SET CHARACTER HEIGHT****L0a**

```
procedure GSetCharHeight(  
    charheight : REAL  
);
```

**SET CHARACTER UP VECTOR****L0a**

```
procedure GSetCharUpVector(  
    charupvector : GAVector  
);
```

[ISO page 23]

**Documents**

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1.

DD 114: Part 2: 1986

**Output attributes****GKS functions****SET TEXT PATH**

L0a

```

procedure GSetTextPath(
    textpath      : GTpath
);

```

**SET TEXT ALIGNMENT**

L0a

```

procedure GSetTextAlign(
    textalign     : GRAalign
);

```

**SET FILL AREA INTERIOR STYLE**

L0a

```

procedure GSetFillIntStyle(
    fillintstyle  : GTinterior
);

```

**SET FILL AREA STYLE INDEX**

L0a

```

procedure GSetFillStyleInd(
    fillstyleind  : INTEGER
);

```

**SET FILL AREA COLOUR INDEX**

L0a

```

procedure GSetFillColInd(
    fillcolind    : GTint0
);

```

**SET PATTERN SIZE**

L0a

```

procedure GSetPatternSize(
    patternsizes  : GAVector
);

```

**SET PATTERN REFERENCE POINT**

L0a

```

procedure GSetPatternRefPoint(
    refpoint     : GRpoint
);

```

**SET ASPECT SOURCE FLAGS**

L0a

```

procedure GSetASF(
    listasf      : GAasf
);

```

[ISO page 24]

**Documents**

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1.

Document Sharing Hub

## GKS functions

## Output attributes

## SET PICK IDENTIFIER

L1b

```

procedure GSetPickId(
    pickid          : GTPickId
);

```

## 5.4.2 Workstation attributes (representations)

**SET POLYLINE REPRESENTATION**  
**SET POLYMARKER REPRESENTATION**  
**SET TEXT REPRESENTATION**  
**SET FILL AREA REPRESENTATION**

L1a

L1a

L1a

L1a

```

procedure GSetPrimRep(
    prim          : GTPrim;
    wsid         : GTWsid;
    primind       : GTint1;
    primrep       : GRPrimRep
);

```

This procedure is a general

SET <Primitive> REPRESENTATION

By substituting for the components of the type GTPrim, the GKS abstract functions are obtained.

## SET PATTERN REPRESENTATION

L1a

*Pascal level 1*

```

procedure GSetPatternRep(
    wsid          : GTWsid;
    patternind     : GTint1;
    dx,dy         : GTint1;
    VAR patternarray : array[min1..max1 : INTEGER; min2..max2 : INTEGER] of GTint0
);

```

The pattern array bounds are min1 to min1 + dx - 1 ( $\leq$  max1) and min2 to min2 + dy - 1 ( $\leq$  max2). Although the array is declared as VAR the implementation shall not change the values of the array elements.

Errors:

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

*Pascal level 0*

```

procedure GSetPatternRep(
    wsid          : GTWsid;
    patternind     : GTint1;
    dx,dy         : GTint1;
    VAR patternarray : GAColArray
);

```

The parameters dx and dy satisfy  $dx \leq GCmaxDX$  and  $dy \leq GCmaxDY$ . Although the array is declared as VAR the implementation shall not change the values of the array elements.

Errors:

2100 *There is an incompatibility between the bounds of the array and the actual size parameters specified*

## Output attributes

## GKS functions

## SET COLOUR REPRESENTATION

L0a

procedure GSetColRep(

```

    wsid          : GTWslId;
    colind        : GTint0;
    col           : GRCol
);
```

## 5.5 Transformation functions

## 5.5.1 Normalization transformation

## SET WINDOW

L0a

procedure GSetWindow(

```

    trannum       : GTint1;
    windowlimits  : GRbound
);
```

## SET VIEWPORT

L0a

procedure GSetViewport(

```

    trannum       : GTint1;
    viewportlimits : GRbound
);
```

## SET VIEWPORT INPUT PRIORITY

L0b

procedure GSetViewportPriority(

```

    trannum,
    refrannum      : GTint0;
    relativepriority : GTpriority
);
```

## SELECT NORMALIZATION TRANSFORMATION

L0a

procedure GSelectNormTran(

```

    trannum       : GTint0
);
```

## SET CLIPPING INDICATOR

L0a

procedure GSetClip(

```

    clip          : GTclip
);
```

[ISO page 26]