American National Standard

for information systems -
computer graphics -
graphical kernel system (GKS)
functional description
This standard has been adopted for Federal Government use.
Details concerning its use within the Federal Government are contained in Federal Information Processing Standards Publication 120-1, Graphical Kernel System (GKS). For a complete list of the publications available in the Federal Information Processing Standards Series, write to the Standards Processing Coordinator (ADP), National Institute of Standards and Technology, Gaithersburg, MD 20899.
American National Standard for Information Systems –
Computer Graphics –
Graphical Kernel System (GKS)
Functional Description

secretariat
Computer and Business Equipment Manufacturers Association

Approved June 24, 1985
American National Standards Institute, Inc

Abstract

The graphical kernel system (GKS) is a set of basic functions for computer graphics programming usable by many graphics producing applications. This standard (1) allows graphics application programs to be easily transported between installations, (2) aids graphics application programmers in understanding and using graphics methods, and (3) guides device manufacturers on useful graphics capabilities.

This standard defines an application level programming interface to a graphics system. Hence, it contains functions for (1) outputting graphical primitives, (2) controlling the appearance of graphical primitives with attributes, (3) controlling graphical workstations, (4) controlling transformations and coordinate systems, (5) generating and controlling groups of primitives called segments, (6) obtaining graphical input, (7) manipulating groups of device-independent instructions called metafiles, (8) inquiring the capabilities and states of the graphics system, and (9) handling errors. Twelve upwardly compatible levels of conformance are defined, addressing the most common classes of equipment and applications.
Approval of an American National Standard requires verification by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made toward their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give an interpretation of any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.
Foreword

This American National Standard provides a set of basic functions for computer graphics programming. These functions, taken as a whole, are called the graphical kernel system (GKS). The design of this standard is based on the work of many groups. Much of the early design methodology was developed at the Workshop on Graphics Standards Methodology held in May 1976 in Seillac, France, under IFIP WG5.2 sponsorship. GKS itself was originally developed by Deutsches Institut für Normung (DIN), the West German standardization institute, in 1978 and was subsequently refined extensively between 1980 and 1982 by Working Group 2 of the Subcommittee on Programming Languages of the Technical Committee on Information Processing of the International Organization for Standardization (ISO TC 97/SC5/WG2). The resulting International Standard (Information Processing – Computer Graphics – Graphical Kernel System (GKS) Functional Description, ISO 7942-1985) was the basis for this American National Standard. The development of the GKS was heavily influenced by the work of the Graphic Standards Planning Committee of the Special Interest Group on Computer Graphics of the Association for Computing Machinery (ACM SIGGRAPH GSPC). This work, known as the Core System Proposal, was published and widely distributed in 1977 and again (in a revised version) in 1979.

This American National Standard on GKS is identical to ISO 7942-1985 (GKS) in almost all areas. All functional capabilities of ISO GKS are found in the ANSI GKS. The ANSI GKS does, however, differ in the following ways:

1. A new minimal output level (denoted m) is defined.
2. A new section defining a conforming program and a conforming implementation replaces a more restrictive conformance statement found in the body of the ISO GKS standard document.
3. Several of the Annexes in the ISO GKS document have been modified. Also, the word "Annex" has been changed to "Appendix."
4. The default for ASFs is INDIVIDUAL.
5. The data records for INPUT have been defined.


This standard was developed by Technical Committee X31H3 of Accredited Standards Committee X3 under two projects authorized by X3; namely, project 268D and project 362D. More specifically, GKS, as a whole, meets the goals of project 268D, while the minimal output level m found in this American National Standard, but not present in ISO 7942-1985, meets the goals of project 362D.

This standard was approved as an American National Standard by the American National Standards Institute on June 24, 1985.

Suggestions for improvement of this standard will be welcome. They should be sent to the Computer and Business Equipment Manufacturers Association, 311 First Street, NW, Suite 500, Washington, DC 20001.
This standard was processed and approved for submittal to ANSI by Accredited Standards Committee on Information Processing Systems, X3. Committee approval of the standard does not necessarily imply that all committee members voted for its approval. At the time it approved this standard, the X3 Committee had the following members:

Edward Lohse, Chair  
Catherine A. Kachurik, Administrative Secretary

<table>
<thead>
<tr>
<th>Organization Represented</th>
<th>Name of Representative</th>
</tr>
</thead>
<tbody>
<tr>
<td>American Library Association</td>
<td>Paul Peters</td>
</tr>
<tr>
<td>American Nuclear Society</td>
<td>Geraldine C. Main</td>
</tr>
<tr>
<td>AMP Incorporated</td>
<td>D. R. Vondy (Alt)</td>
</tr>
<tr>
<td>Association of American Railroads</td>
<td>Patrick E. Lannan</td>
</tr>
<tr>
<td>Association for Computing Machinery</td>
<td>Edward Kelly (Alt)</td>
</tr>
<tr>
<td>Association of the Institute for Certification of Computer Professionals</td>
<td>R. A. Petrash</td>
</tr>
<tr>
<td>AT&amp;T Corporation</td>
<td>Kenneth Magel</td>
</tr>
<tr>
<td>AT&amp;T Information Systems</td>
<td>Jon A. Meads (Alt)</td>
</tr>
<tr>
<td>Burroughs Corporation</td>
<td>Thomas M. Kurihara</td>
</tr>
<tr>
<td>Control Data Corporation</td>
<td>Azdyn E. Dubnow (Alt)</td>
</tr>
<tr>
<td>Cooperating Users of Burroughs Equipment</td>
<td>Henry L. Marchese</td>
</tr>
<tr>
<td>Data General Corporation</td>
<td>Stuart H. Garland (Alt)</td>
</tr>
<tr>
<td>Data Processing Management Association</td>
<td>Richard Gibson (Alt)</td>
</tr>
<tr>
<td>Digital Equipment Computer Users Society</td>
<td>Herbert V. Bertine</td>
</tr>
<tr>
<td>Digital Equipment Corporation</td>
<td>Paul D. Bartoli (Alt)</td>
</tr>
<tr>
<td>General Electric Company</td>
<td>Ira R. Purchis</td>
</tr>
<tr>
<td>General Services Administration</td>
<td>Stanley Fenner (Alt)</td>
</tr>
<tr>
<td>Harris Corporation</td>
<td>Charles E. Cooper</td>
</tr>
<tr>
<td>Hewlett-Packard</td>
<td>Keith Lucke (Alt)</td>
</tr>
<tr>
<td>Honeywell Information Systems</td>
<td>Donald Miller (Alt)</td>
</tr>
<tr>
<td>IBM Corporation</td>
<td>John Pilat</td>
</tr>
<tr>
<td>IEEE Computer Society</td>
<td>Lyman Chapin (Alt)</td>
</tr>
<tr>
<td>Lawrence Berkeley Laboratory</td>
<td>Frank Kishenbaum</td>
</tr>
<tr>
<td>Life Office Management Association</td>
<td>Thomas F. O'Leary, Jr (Alt)</td>
</tr>
<tr>
<td>Moore Business Forms</td>
<td>Sam Mathan</td>
</tr>
<tr>
<td>National Bureau of Standards</td>
<td>Rajiv Sinha (Alt)</td>
</tr>
<tr>
<td>National Communications System</td>
<td>Donald C. Loughry</td>
</tr>
<tr>
<td>NCR Corporation</td>
<td>Thomas J. McNamara</td>
</tr>
<tr>
<td>Perkin-Elmer Corporation</td>
<td>David M. Taylor (Alt)</td>
</tr>
<tr>
<td>Prime Computer, Inc</td>
<td>Mary Anne Gray</td>
</tr>
<tr>
<td>Recognition Technology Users Association</td>
<td>Robert H. Folletti (Alt)</td>
</tr>
<tr>
<td>Moore Business Forms</td>
<td>Sava I. Sherr</td>
</tr>
<tr>
<td>National Bureau of Standards</td>
<td>David Gelperin (Alt)</td>
</tr>
<tr>
<td>National Communications System</td>
<td>Thomas A. Varetoni (Alt)</td>
</tr>
<tr>
<td>Lawrence Berkeley Laboratory</td>
<td>David Stevens</td>
</tr>
<tr>
<td>Life Office Management Association</td>
<td>John S. Colonias (Alt)</td>
</tr>
<tr>
<td>Moore Business Forms</td>
<td>James J. Merrick</td>
</tr>
<tr>
<td>National Bureau of Standards</td>
<td>James F. Foley, Jr (Alt)</td>
</tr>
<tr>
<td>National Communications System</td>
<td>Delmer H. Oddy</td>
</tr>
<tr>
<td>NCR Corporation</td>
<td>Robert E. Rumtree</td>
</tr>
<tr>
<td>Perkin-Elmer Corporation</td>
<td>James H. Burrows (Alt)</td>
</tr>
<tr>
<td>Prime Computer, Inc</td>
<td>Marshall L. Cain</td>
</tr>
<tr>
<td>Recognition Technology Users Association</td>
<td>George W. White (Alt)</td>
</tr>
<tr>
<td>Moore Business Forms</td>
<td>Thomas W. Kern</td>
</tr>
<tr>
<td>National Bureau of Standards</td>
<td>A. R. Daniels (Alt)</td>
</tr>
<tr>
<td>Lawrence Berkeley Laboratory</td>
<td>Christopher Beling</td>
</tr>
<tr>
<td>Life Office Management Association</td>
<td>Russ Lombardo (Alt)</td>
</tr>
<tr>
<td>Moore Business Forms</td>
<td>Andrew F. Burlingame</td>
</tr>
<tr>
<td>National Communications System</td>
<td>Jacqueline Barbour (Alt)</td>
</tr>
<tr>
<td>NCR Corporation</td>
<td>Herbert F. Schantz</td>
</tr>
<tr>
<td>Lawrence Berkeley Laboratory</td>
<td>G. W. Wetzel (Alt)</td>
</tr>
</tbody>
</table>
Organization Represented

SHARE, Inc
Sperry Corporation
Texas Instruments, Inc
3M Company
Travelers Insurance Companies, Inc
U.S. Department of Defense
VIM
Wang Laboratories, Inc
Xerox Corporation

Name of Representative

Thomas R. Steel
Daniel Schuster (Alt)
Marvin W. Bass
Presley Smith
Richard F. Trow, Jr (Alt)
J. Wade Van Valkenburg
Joseph T. Brophy
Belkis Leong-Hong
Fred Virtue (Alt)
Chris Tanner
Madeleine Sparks (Alt)
Marsha Hayek
Joseph St. Amand (Alt)
John L. Wheeler
Arthur R. Machell (Alt)

Technical Committee X3H3 on Computer Graphics, which developed the draft proposals, which held the U.S. Technical Advisory Group responsibilities for ISO TC 97/SC5/WG2, and through which this standard was completed, had the following members:

Peter R. Bono, Chair
(Athena Systems)
Barry Shepherd, Vice-Chair
(IBM)
Randall L. Simons, Secretary
(Sandia, NM)
Janet Chin, International Representative
(Tymshare, Inc)

David C. Bailey
(Sanders Associates)
Jerry Bedrick
(Wang Laboratories)
John Blair
(Mindset Corporation)
Robert Bruns
(Megatek)
Albert Bunshaft
(RPI)
Debbie Cahn
(LBL)
Fred Canfield
(Systonetics, Inc)
George S. Carson
(GSC Associates)
Tom Clarkson
(GSS)
Bruce Cohen
(Intel)
Geraldine Cuthbert
(Harris)
Warren Dale
(Calma)
Richard Eilers
(F & S)
Sam Gill
(Benson)
Jim Hargrove
(Data General)
Terry Harney
(Hughes Aircraft)
Lofton Henderson
(NCAR)
Margaret Journey
(Precision Visuals)
Jim Kearney
(US Army)
Fred Langhorst
(Digital Research)
Olga Lapczak
(Norpak)
Allen Leinwand
(Olivetti)
K. C. Leung
(Computer Sciences)
Chris Mannhardt
(BNR)
Tom Morrissey
(Hewlett-Packard)
Paul Norman
(Imac Corporation)
Brian Plunkett
(Houston Instrument)
Tom Powers
(DEC)
L. Preheim
(JPL)
Richard F. Puk
(Puk Consulting Services)
Tom Reed
(LANL)
Jeff Rowe
(Lawrence Livermore National Laboratory)
Kurt Schmucker
(NSA)
Chuck Seum
(CDC)
David Shuey
(McAuto)
Mark Skall
(NBS)
Elaine Sonderegger
(SIGGRAPH)
Norman Soong
(Sperry)
Madeleine Sparks
(SDC)
Richard Stout
(SAI)
David Straayer
(Tektronix)
Meredith L. Whyles
(Amoco Production)
Thomas Wright
(ISSCO)
Contents

0 Introduction 1
  0.1 Conformance 3
1 Scope and Field of Application 4
2 References 5
3 Definitions 6
4 The Graphical Kernel System 11
  4.1 The Standard 11
    4.1.1 Specification 11
    4.1.2 Registration 11
  4.2 Introduction 12
  4.3 Concepts 14
  4.4 Graphical output 16
    4.4.1 Output primitives 16
    4.4.2 Output primitive attributes 16
    4.4.3 Polyline attributes 20
    4.4.4 Polymarker attributes 22
    4.4.5 Text attributes 22
    4.4.6 Fill area attributes 33
    4.4.7 Cell array attributes 34
    4.4.8 Generalized Drawing Primitive attributes 34
    4.4.9 Colour 34
  4.5 Workstations 36
    4.5.1 Workstation characteristics 36
    4.5.2 Selecting a workstation 37
    4.5.3 Deferring picture changes 38
    4.5.4 Clearing the display surface 41
    4.5.5 Elimination of primitives outside segments 41
    4.5.6 Sending messages to a workstation 41
  4.6 Coordinate systems and transformations 42
    4.6.1 Normalization transformations 42
    4.6.2 Clipping 42
    4.6.3 Workstation transformations 43
    4.6.4 Transformation of locator input 46
    4.6.5 Transformation of stroke input 48
  4.7 Segments 49
    4.7.1 Introduction to segments 49
    4.7.2 Segment attributes 50
    4.7.3 Segment transformations 51
    4.7.4 Clipping and WDSS 51
    4.7.5 Workstation Independent Segment Storage 51
    4.7.6 WISS functions and clipping 52
  4.8 Graphical input 53
    4.8.1 Introduction to logical input devices 53
    4.8.2 Logical input device model 54
    4.8.3 Operating modes of logical input devices 55
    4.8.4 Measures of each input class 57
    4.8.5 Input queue and current event report 58
    4.8.6 Initialisation of input devices 59
  4.9 GKS Metafile interface 61
  4.10 GKS levels 62
    4.10.1 Introduction 62
    4.10.2 The level structure 62
    4.10.3 Level functionality 63
Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>E.3 Generation of metafiles</td>
<td>245</td>
</tr>
<tr>
<td>E.4 Interpretation of metafiles</td>
<td>248</td>
</tr>
<tr>
<td>E.4.1 Introduction</td>
<td>248</td>
</tr>
<tr>
<td>E.4.2 Control items</td>
<td>248</td>
</tr>
<tr>
<td>E.4.3 Output primitives</td>
<td>248</td>
</tr>
<tr>
<td>E.4.4 Output primitive attributes</td>
<td>248</td>
</tr>
<tr>
<td>E.4.5 Workstation attributes</td>
<td>248</td>
</tr>
<tr>
<td>E.4.6 Transformations</td>
<td>248</td>
</tr>
<tr>
<td>E.4.7 Segment manipulation</td>
<td>248</td>
</tr>
<tr>
<td>E.4.8 Segment attributes</td>
<td>248</td>
</tr>
<tr>
<td>E.5 Control items</td>
<td>248</td>
</tr>
<tr>
<td>E.6 Items for output primitives</td>
<td>251</td>
</tr>
<tr>
<td>E.7 Items for output primitive attributes</td>
<td>252</td>
</tr>
<tr>
<td>E.8 Items for workstation attributes</td>
<td>255</td>
</tr>
<tr>
<td>E.9 Items for transformations</td>
<td>256</td>
</tr>
<tr>
<td>E.10 Items for segment manipulation</td>
<td>256</td>
</tr>
<tr>
<td>E.11 Items for segment attributes</td>
<td>257</td>
</tr>
<tr>
<td>E.12 User items</td>
<td>258</td>
</tr>
<tr>
<td>F GKS functions summary</td>
<td>259</td>
</tr>
<tr>
<td>F.1 Control functions</td>
<td>259</td>
</tr>
<tr>
<td>F.2 Output functions</td>
<td>259</td>
</tr>
<tr>
<td>F.3 Output attributes</td>
<td>260</td>
</tr>
<tr>
<td>F.3.1 Workstation independent primitive attributes</td>
<td>260</td>
</tr>
<tr>
<td>F.3.2 Workstation attributes (representations)</td>
<td>261</td>
</tr>
<tr>
<td>F.4 Transformation functions</td>
<td>261</td>
</tr>
<tr>
<td>F.4.1 Normalization transformation</td>
<td>261</td>
</tr>
<tr>
<td>F.4.2 Workstation transformation</td>
<td>261</td>
</tr>
<tr>
<td>F.5 Segment functions</td>
<td>262</td>
</tr>
<tr>
<td>F.5.1 Segment manipulation functions</td>
<td>262</td>
</tr>
<tr>
<td>F.5.2 Segment attributes</td>
<td>262</td>
</tr>
<tr>
<td>F.6 Input functions</td>
<td>262</td>
</tr>
<tr>
<td>F.6.1 Initialisation of input devices</td>
<td>262</td>
</tr>
<tr>
<td>F.6.2 Setting mode of input devices</td>
<td>263</td>
</tr>
<tr>
<td>F.6.3 Request input functions</td>
<td>263</td>
</tr>
<tr>
<td>F.6.4 Sample input functions</td>
<td>263</td>
</tr>
<tr>
<td>F.6.5 Event input functions</td>
<td>264</td>
</tr>
<tr>
<td>F.7 Metafile functions</td>
<td>264</td>
</tr>
<tr>
<td>F.8 Inquiry functions</td>
<td>265</td>
</tr>
<tr>
<td>F.9 Utility functions</td>
<td>265</td>
</tr>
<tr>
<td>F.10 Error handling</td>
<td>265</td>
</tr>
<tr>
<td>G Differences Between ANS GKS and ISO GKS</td>
<td>267</td>
</tr>
</tbody>
</table>
0 Introduction

The Graphical Kernel System (GKS) provides a set of functions for computer graphics programming. GKS is a basic graphics system that can be used by the majority of applications that produce computer generated pictures.

The main reasons for introducing a standard for basic computer graphics are:

a) to allow application programs involving graphics to be easily portable between different installations;

b) to aid the understanding and use of graphics methods by application programmers;

c) to serve manufacturers of graphics equipment as a guideline in providing useful combinations of graphics capabilities in a device.

In order to reach these main objectives, the GKS design was based on the following requirements:

d) GKS should include all the capabilities that are essential for a broad spectrum of graphics, from simple passive output to highly interactive applications.

e) The whole range of graphics devices, including vector and raster devices, microfilm recorders, storage tube displays, refresh displays and colour displays should be controllable by GKS in a uniform way.

f) GKS should provide all the capabilities required by a majority of applications without becoming unduly large.

These requirements were used to formulate a number of principles that were used to judge specific design alternatives. Thus it was possible to contribute to the overall design goals while focussing on certain aspects. Five design aspects were identified, each having a group of principles.

g) Design goals: The following principles should not be violated by any technical design:

1) consistency: the mandatory requirements of GKS should not be mutually contradictory;

2) compatibility: other standards or commonly accepted rules of practice should not be violated;

3) orthogonality: the functions or modules of GKS should be independent of each other, or the dependency should be structured and well defined.

h) Functional capabilities: The following principles were used to define the extent of GKS:

1) completeness: all functions that a majority of applications want to use on a given level of functionality should be included;
2) minimality: functions that are unnecessary for applications of a given level of functionality should not be provided;

3) compactness: an application should be able to achieve a desired result by a set of functions and parameters that is as small as possible;

4) richness: a rich set of functions offers an extensive range of facilities that stretches beyond the basic functions and includes higher order capabilities.

It is obvious that there is a trade off between the principles in this group. Therefore, the functions of GKS are organized in twelve levels. An implementation of GKS provides at least the functions of one of these levels. While the lowest level contains only a minimal set of functions, higher levels are allowed to extend beyond the basic needs towards greater richness.

i) User interface design: The following principles were used to define the user interface design:

1) user friendliness: GKS should allow the design of a desirable user interface;

2) clarity: the concepts and functional capabilities of GKS should be easily understandable, especially by the application programmer;

3) error handling: failure of system functions or modules, caused by errors of the system itself or by the application program, should be treated in such a way that the error reaction is clearly understandable and informative to the application programmer and that the impact on the system and the application program is as small as possible.

Clarity and sound error handling are essential parts of user friendliness. Error handling is an integral part of GKS. To aid clarity, the system and its state can be presented to the user in an easily comprehensible manner.

Clarity applies not only to the system design but also to the system description. To this end, the GKS specification is divided into a general description, a description of the underlying logical data structures representing the state of the system, and a description of the functions and their effects on these data structures.

j) Graphics devices: The following principles are associated with the range of graphics devices that can be addressed by GKS:

1) device independence: GKS functions should be designed to allow an application program, using these functions, to address facilities of quite different graphics output and input devices without modification of the program structure;

2) device richness: the full capabilities of a wide range of different graphics output and input devices should be accessible from the functions of GKS.

These principles led to a fundamental concept underlying the GKS architecture: the concept of multiple independent graphical workstations connected to and driven by GKS. The application program can inquire the capabilities of every workstation. The GKS design includes escape functions that are easily identifiable within an application program and can be used to access special facilities of a particular device.

k) Implementation: The last group of principles is related to the implementation of GKS:

1) implementability: it should be possible to support the GKS functions in most host languages, on most operating systems and with most graphics devices;

2) language independence: it should be possible to access the standard facilities of GKS from all ISO standard programming languages;

3) efficiency: GKS should be capable of being implemented without time consuming algorithms;

4) robustness: the operator and application programmer should be protected in the best possible way from hardware or software failure of the system.
The five groups of principles are interconnected. For example, design goals and functional capabilities both contribute to user friendliness. Efficiency is also important when considering response time in an interactive environment. Some principles may be conflicting, such as richness versus minimality, comprehensive error handling versus efficiency, and compactness versus device richness. Compromises needed to be made to achieve the overall design objective: GKS should have an easily comprehensible structure and a set of functions that enables a vast majority of computer graphics users to design portable, device independent application programs addressing the whole range of computer graphics equipment.

0.1 Conformance

An implementation conforms to a specific level of this standard if it provides, at least, all the functions specified in that level, but not all the functions of the next higher level. Any function specified by the standard that is provided by an implementation must execute according to the semantics specified in this standard. In addition, an implementation may provide functions not specified in any level of this standard as long as these extensions do not cause standard functions to execute incorrectly. These extensions should be constructed to obey the philosophy of GKS.

A program conforms to a specific level of this standard if it does not use any functions outside of that level.

To conform to the standard, an implementation or program that is written for a language for which a standard binding has been developed must use the syntax specified in that language binding.
1 Scope and Field of Application

This American National Standard specifies a set of functions for computer graphics programming, the Graphical Kernel System (GKS). GKS is a basic graphics system for applications that produce computer generated two dimensional pictures on line graphics or raster graphics output devices. It supports operator input and interaction by supplying basic functions for graphical input and picture segmentation. It allows storage and dynamic modification of pictures. A fundamental concept in GKS is the workstation, consisting potentially of a number of input devices and a single output device. Several workstations can be used simultaneously. The application program is allowed to adapt its behaviour at a workstation to make best use of workstation capabilities. This standard includes functions for storage on and retrieval from an external graphics file. Last, but not least, the functions are organized in upward compatible levels with increasing capabilities.

For certain parameters of the functions, GKS defines value ranges as being reserved for registration or future standardization. The meanings of these values will be defined using the procedures established in an International Standard under development (Procedures for registration of graphical items).

Part 1 of GKS defines only a language independent nucleus of a graphics system. For integration into a language, GKS is embedded in a language dependent layer containing the language conventions, for example, parameter and name assignment. Language bindings are contained in Part 2.

The appendicies are given for information; they do not form part of the specification.
2 References

<table>
<thead>
<tr>
<th>Reference</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANSI X3.4-1977</td>
<td>7-bit coded character set for information processing interchange</td>
</tr>
<tr>
<td>ISO 7942</td>
<td>Information Processing, Graphical Kernel System (GKS), Functional Description</td>
</tr>
<tr>
<td>ISO 646</td>
<td>7-bit coded character set for information processing interchange</td>
</tr>
<tr>
<td>ISO 2022</td>
<td>Information processing - ISO 7-bit and 8-bit coded character sets - Code extension techniques</td>
</tr>
<tr>
<td>ISO 2382/13</td>
<td>Data processing - Vocabulary - Section 13: Computer Graphics</td>
</tr>
<tr>
<td>ISO 6093 ¹</td>
<td>Information processing - Representation of numeric values in character strings for information interchange</td>
</tr>
<tr>
<td>ISO 8632 ¹</td>
<td>Information processing - Computer graphics - Metafile for transfer and storage of picture description information</td>
</tr>
</tbody>
</table>

¹ In course of preparation
This section gives the definition of the important terms of the Graphical Kernel System (GKS).

NOTE As far as possible, commonly accepted graphics terminology is used.

3.1 acknowledgement: Output to the operator of a logical input device indicating that a trigger has fired.

3.2 aspect ratio: A ratio of x to y used to describe the shape of a rectangle in a particular coordinate system (for example, of a workstation window or a workstation viewport).

3.3 aspects of primitives: Ways in which the appearance of a primitive can vary. Some aspects are controlled directly by primitive attributes; some can be controlled indirectly through a bundle table. Primitives inside segments have an aspect controlled through the segment containing them, for example highlighting; primitives outside segments do not.

3.4 attribute: A particular property that applies to a display element (output primitive) or a segment. Examples: highlighting, character height. In GKS, some properties of workstations are called workstation attributes.

3.5 baseline: A horizontal line within a character body (see figure 3) which, for many character definitions, has the appearance of being a lower limit of the character shape. A descender passes below this line. All baselines in a font are in the same position in the character bodies.

3.6 bundle index: An index into a bundle table for a particular output primitive. It defines the workstation dependent aspects of the primitive.

3.7 bundle table: A workstation dependent table associated with a particular output primitive. Entries in the table specify all the workstation dependent aspects of a primitive. In GKS, bundle tables exist for the following output primitives: polyline, polymarker, text and fill area.

3.8 capline: A horizontal line within a character body (see figure 3) which, for many character definitions, has the appearance of being the upper limit of the character shape. An ascender may pass above this line and in some languages an additional mark (for example an accent) over the character may be defined above this line. All caplines in a font are in the same position in the character bodies.

3.9 cell array: A GKS output primitive consisting of a rectangular grid of equal size rectangular cells, each having a single colour.

NOTE These cells do not necessarily map one-to-one with pixels.

3.10 centreline: A vertical line bisecting the character body (see figure 3).

3.11 character body: A rectangle used by a font designer to define a character shape (see figure 3). All character bodies in a font have the same height.

3.12 choice device: A GKS logical input device providing a non-negative integer defining one of a set of alternatives.

3.13 clipping: Removing parts of display elements that lie outside a given boundary, usually a window or viewport.

3.14 colour table: A workstation dependent table, in which the entries specify the values of the red, green and blue intensities defining a particular colour.

3.15 coordinate graphics; line graphics: Computer graphics in which display images are generated from display commands and coordinate data.

3.16 device coordinate (DC): A coordinate expressed in a coordinate system that is device dependent. In GKS, DC units are metres on a device capable of producing a precisely scaled image and appropriate workstation dependent units otherwise.

3.17 device driver: The device dependent part of a GKS implementation intended to support a graphics device. The device driver generates device dependent output and handles device dependent interaction.
Definitions

3.18 device space: The space defined by the addressable points of a display device.

3.19 display device; graphics device: A device (for example refresh display, storage tube display, plotter) on which display images can be represented.

3.20 display image; picture: A collection of output primitives or segments that are represented together at any one time on a display surface.

3.21 display space: (1) That portion of the device space corresponding to the area available for displaying images. (2) The working space of an input device such as a digitiser.

3.22 display surface; view surface: In a display device, that medium on which display images may appear.

3.23 echo: The immediate notification of the current value provided by an input device to the operator at the display console.

3.24 escape: A function in GKS used to access implementation or device dependent features, other than for the generation of graphical output, that are not otherwise addressed by GKS.

3.25 feedback: Output indicating to the operator the application program's interpretation of a logical input value.

3.26 fill area: A GKS output primitive consisting of a polygon (closed boundary) which may be hollow or may be filled with a uniform colour, a pattern, or a hatch style.

3.27 fill area bundle table: A table associating specific values for all workstation dependent aspects of a fill area primitive with a fill area bundle index. In GKS, this table contains entries consisting of interior style, style index, and colour index.

3.28 Generalized Drawing Primitive (GDP): An output primitive used to address special geometrical workstation capabilities such as curve drawing.

3.29 GKS level: Two values from the sets \((m,0,1,2)\) and \((a,b,c)\) which together define the minimal functional capabilities provided by a specific GKS implementation.

3.30 GKS Metafile (GKSM): A sequential file that can be written or read by GKS and is used for long-term storage (and for transmittal and transferral) of graphical information.

3.31 halfline: A horizontal line between the capline and the baseline within the character body (see figure 3), about which a horizontal string of characters in a font would appear centrally placed in the vertical direction. All halflines in a font are in the same position in the character bodies.

3.32 hatch: One possible method of filling the interior of a polygon specified by a fill area primitive. The interior is filled with an arrangement of one or more sets of parallel lines.

3.33 highlighting: A device independent way of emphasizing a segment by modifying its visual attributes. For example, blinking.

3.34 Implementation mandatory: Implementation mandatory describes a property that is required to be realized identically on all workstations of all implementations of GKS.

3.35 Input class: A set of input devices that are logically equivalent with respect to their function. In GKS, the input classes are: LOCATOR, STROKE, VALUATOR, CHOICE, PICK and STRING.

3.36 Inquiry function: A GKS function whose purpose is to return values depending on the current state of GKS or on some fixed property of the GKS implementation. There is no effect on the state of GKS or on the display image.

3.37 locator device: A GKS logical input device providing a position in world coordinates and a normalization transformation number.

3.38 logical Input device: A logical input device is an abstraction of one or more physical devices that delivers logical input values to the program. Logical input devices in GKS can be of type LOCATOR, STROKE, VALUATOR, CHOICE, PICK and STRING.
3.39 logical input value: A value delivered by a logical input device.

3.40 marker: A glyph with a specified appearance which is used to identify a particular location.

3.41 measure: A value (associated with a logical input device), which is determined by one or more physical input devices, and a mapping from the values delivered by the physical devices. The logical input value delivered by a logical input device is the current value of the measure.

3.42 MI: An abbreviation for GKS metafile input, a category of workstation.

3.43 MO: An abbreviation for GKS metafile output, a category of workstation.

3.44 normalization transformation; viewing transformation; window-to-viewport transformation: A transformation that maps the boundary and interior of a window to the boundary and interior of a viewport. In GKS, this transformation maps positions in world coordinates to normalized device coordinates.

3.45 normalized device coordinates (NDC): A coordinate specified in a device independent intermediate coordinate system, normalized to some range, typically 0 to 1. In GKS, during an intermediate state the coordinates may lie outside the defined range, but associated clipping information ensures that the output does not exceed the coordinate range $[0,1] \times [0,1]$.

3.46 operator: Person manipulating physical input devices so as to change the measures of logical input devices and cause their triggers to fire.

3.47 output primitive; graphic primitive; display element: A basic graphic element that can be used to construct a display image. Output primitives in GKS are polyline, polymarker, text, fill area, cell array, and generalized drawing primitive.

3.48 pick device: A GKS logical input device providing the pick identifier attached to an output primitive and the associated segment name.

3.49 pick Identifier: A name, attached to individual output primitives within a segment, and returned by the pick device. The same pick identifier can be assigned to different output primitives.

3.50 pixel; picture element: The smallest element of a display surface that can be independently assigned a colour or intensity.

3.51 polyline: A GKS output primitive consisting of a set of connected lines.

3.52 polyline bundle table: A table associating specific values for all workstation dependent aspects of a polyline primitive with a polyline bundle index. In GKS, this table contains entries consisting of linetype, linewidth scale factor, and colour index.

3.53 polymarker: A GKS output primitive consisting of a set of locations, each to be indicated by the same type of marker.

3.54 polymarker bundle table: A table associating specific values for all workstation dependent aspects of a polymarker primitive with a polymarker bundle index. In GKS, this table contains entries consisting of marker type, marker size scale factor, and colour index.

3.55 primitive attribute: An attribute that applies to output primitives as opposed to attributes that apply to other aspects of the graphical system such as segments. Primitive attribute values (for output primitives) are selected by the application in a workstation independent manner, but can have workstation dependent effects.

3.56 prompt: Output to the operator indicating that a specific logical input device is available.

3.57 raster graphics: Computer graphics in which a display image is composed of an array of pixels arranged in rows and columns.

3.58 rotation: Turning all or part of a display image about an axis. In GKS, this capability is
Definitions

restricted to segments.

3.59 **scaling; zooming:** Enlarging or reducing all or part of a display image by multiplying the coordinates of display elements by a constant value. In GKS, this capability is restricted to segments.

NOTE For different scaling in two orthogonal directions two constant values are required.

3.60 **segment:** A collection of display elements that can be manipulated as a unit.

3.61 **segment attributes:** Attributes that apply only to segments. In GKS, segment attributes are visibility, highlighting, detectability, segment priority, and segment transformation.

3.62 **segment priority:** A segment attribute used to determine which of several overlapping segments takes precedence for graphic output and input.

3.63 **segment transformation:** A transformation that causes the display elements defined by a segment to appear with varying position (translation), size (scale), and/or orientation (rotation) on the display surface.

3.64 **string device:** A GKS logical input device providing a character string as its result.

3.65 **stroke device:** A GKS logical input device providing a sequence of points in world coordinates, and a normalization transformation number.

3.66 **text:** A GKS output primitive consisting of a character string.

3.67 **text bundle table:** A table associating specific values for all workstation dependent aspects of a text primitive with a text bundle index. In GKS, this table contains entries consisting of text font and precision, character expansion factor, character spacing and colour index.

3.68 **text font and precision:** An aspect of text in GKS, having two components, font and precision, which together determine the shape of the characters being output, on a particular workstation. In addition, the precision describes the fidelity with which the other text aspects match those requested by an application program. In order of increasing fidelity, the precisions are: STRING, CHAR and STROKE.

3.69 **translation; shift:** The application of a constant displacement to the position of all or part of a display image. In GKS, this capability is restricted to segments.

3.70 **trigger:** A physical input device or set of devices that an operator can use to indicate significant moments in time.

3.71 **valuator device:** A GKS logical input device providing a real number.

3.72 **viewport:** An application program specified part of normalized device coordinate space. In GKS, this definition is restricted to a rectangular region of normalized device coordinate space used in the definition of the normalization transformation.

3.73 **window:** A predefined part of a virtual space. In GKS, this definition is restricted to a rectangular region of the world coordinate space used for the definition of the normalization transformation.

3.74 **workstation:** GKS is based on the concept of abstract graphical workstations, which provide the logical interface through which the application program controls physical devices.

3.75 **Workstation Dependent Segment Storage (WDSS):** Segment storage on a workstation that is used for graphical output. Segments cannot be transferred from WDSS to another workstation.

3.76 **Workstation Independent Segment Storage (WISS):** A special workstation type, where segments can be stored and later transferred to other workstations.

3.77 **workstation mandatory:** Workstation mandatory describes a property that is required to be realized identically on all workstations of a GKS implementation.

3.78 **workstation transformation:** A transformation that maps the boundary and interior of a workstation window into the boundary and interior of a workstation viewport (part of display space), preserving aspect ratio. In GKS, this transformation maps positions in normalized
device coordinates to device coordinates. The effect of preserving aspect ratio is that the interior of the workstation window may not map to the whole of the workstation viewport.

3.70 workstation viewport: A portion of display space currently selected for output of graphics.

3.80 workstation window: A rectangular region within the normalized device coordinate system which is represented on a display space.

3.81 world coordinate (WC): A device independent Cartesian coordinate system used by the application program for specifying graphical input and output.
4 The Graphical Kernel System

4.1 The Standard

4.1.1 Specification
The set of functions known as the Graphical Kernel System shall be as described in sections 4, 5 and 6. These functions are organized in twelve upward compatible levels with increasing capabilities as described in 4.10.

4.1.2 Registration
For certain parameters of the functions, GKS defines value ranges as being reserved for registration or future standardization. The meanings of these values will be defined using the procedures established in an International Standard under development (Procedures for registration of graphical items). These procedures do not apply to values and value ranges defined as being workstation or implementation dependent; these values and ranges are not standardized.
4.2 Introduction

The Graphical Kernel System (GKS) provides a functional interface between an application program and a configuration of graphical input and output devices. The functional interface contains all basic functions for interactive and non-interactive graphics on a wide variety of graphics equipment.

The interface is at such a level of abstraction that hardware peculiarities are shielded from the application program. As a result a simplified interface presenting uniform output primitives (POLYLINE, POLYMARKER, TEXT, FILL AREA, CELL ARRAY, GENERALIZED DRAWING PRIMITIVE), and uniform input classes (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING) is obtained.

In 4.3 the concepts of basic output, input and the organization of input and output sequences are outlined. A central concept both for structuring GKS and for realizing device independence is introduced, called the workstation.

The facilities for picture manipulation and change are introduced via the segment facilities, the dynamic attributes and the transformations. The integral control over all these methods for change is further explained in 4.5.3 on workstations.

The concept of multiple workstations allows simultaneous output to and input from various display systems. Facilities for internal and external storage are provided by special workstations together with the possibility of transferring graphical entities directly from the special workstation for internal storage to other workstations.

Not every GKS implementation needs to support the full set of functions. Twelve levels are defined to meet the different requirements of graphics systems. Each GKS implementation provides at least the functions of one level. The levels are upward compatible.

Part 1 of GKS defines only a language independent nucleus of a graphics system. For integration into a language, GKS is embedded in a language dependent layer containing the language conventions, for example, parameter and name assignment. Language bindings are contained in Part 2.

The layer model represented in figure 1 illustrates the role of GKS in a graphics system. Each layer may call the functions of the adjoining lower layers. In general the application program uses the application oriented layer, the language dependent layer, other application dependent layers, and operating system resources. All workstation capabilities that can be addressed by GKS functions are used only via GKS.
Application Program

Application Oriented Layer

Language Dependent Layer

Graphical Kernel System

Operating System

Other Resources

Graphical Resources

Figure 1. Layer model of GKS
4.3 Concepts

The graphical output that is generated by GKS is built up from two groups of basic elements called output primitives and primitive attributes. The output primitives are abstractions of basic actions a device can perform, such as drawing lines, and printing character strings. The attributes control the aspects of the output primitives on a device, such as linestyle, colour, character height and pick identifier. Non-geometric aspects, such as colour but not character height, can be controlled for each workstation individually, to make best use of its capabilities.

The graphical information that is input from a device, as a result of operator actions, is mapped by GKS onto six classes of input each represented by a data type referred to as a logical input value. An instance of such a device representation is called a logical input device. The effect of input actions on the display surface, such as prompts and echoes, is controlled by GKS for each logical input device individually.

The two abstract concepts (abstract output and abstract input) are the building blocks of a so-called abstract workstation. A workstation of GKS represents a unit consisting of zero or one display surfaces and zero or more input devices, such as keyboard, tablet and lightpen. The workstation presents these devices to the application program as a configuration of abstract devices thereby shielding the hardware peculiarities.

The geometrical information (coordinates) contained in output primitives, attributes and logical input values (locators and strokes) can be subjected to transformations. These transformations perform mappings between three coordinate systems, namely:

a) World Coordinates (WC), used by the application programmer;

b) Normalized Device Coordinates (NDC), used to define a uniform coordinate system for all workstations;

c) Device Coordinates (DC), one coordinate system per workstation, representing its display space coordinates.

Output primitives and attributes are mapped from WC to NDC by normalization transformations, from NDC to NDC by segment transformations (see next paragraph), and from NDC to DC by workstation transformations. Locator input is mapped by an inverse workstation transformation from DC to NDC and by one of the inverse normalization transformations from NDC to WC.

Output primitives and primitive attributes may be grouped together in a segment. Segments are the units for manipulation and change. Manipulation includes creation, deletion, and renaming. Change includes transforming a segment, making a segment visible or invisible, and highlighting a segment. Segments also form the basis for workstation independent storage of pictures at run time. Via this storage, which is set up as a special workstation called workstation independent segment storage, segments can be inserted and transferred to other workstations.

The attributes which control the appearance of parts of the picture (output primitives, segments, prompt and echo types of input devices) on the display surface are organized in a uniform manner. Two groups of attributes apply to the appearance of each output primitive: primitive attributes (that are workstation independent) and workstation attributes. Primitive attributes are specified modally and are bound to a primitive when it is created. The primitive attributes include all geometric aspects of primitives, such as character height for text and pattern size for fill area. In addition, the non-geometric aspects of primitives are controlled by the primitive attributes in one of two ways. Either a single attribute is used to specify all the non-geometric aspects of the primitive by an index which points to a workstation dependent representation (set of values) or one attribute is used to specify each of the non-geometric aspects of the primitive in a workstation independent way. The former is referred to as bundled specification and the latter is referred to as individual specification.
Workstation attributes include the actual representations on a workstation pointed to by indices used in bundled specification of non-geometric aspects. For example, the representations (or bundles) for polyline each contain values of linetype, linewidth scale factor and colour index. Workstation attributes also specify the colour and pattern tables and the control over deferral of picture change. Workstation attributes can be reset dynamically.

The appearance of segments is controlled by segment attributes, which are segment transformation, visibility, highlighting, segment priority, and detectability. These may be reset dynamically. Segment attributes can be a basis for feedback during manipulations (for example, highlighting).

The attributes which control the operation of logical input devices can be specified either upon initialisation or as part of input device setting, depending upon the attributes. Through initialisation, an initial value, a prompt and echo technique, and an area on the screen for echoing can be specified. A data record may further provide device specific attributes. Through input device setting, the operating mode may be selected and the echo may be switched on or off. The operating modes of logical input devices specify who (operator or application program) has the initiative: SAMPLE input is acquired directly by the application program; REQUEST input is produced by the operator in direct response to the application program; EVENT input is generated asynchronously by the operator and is collected in a queue for the application program.

At run time GKS can be in one of five different operating states. Associated with each state are the set of GKS functions allowed in this state, and a set of state variables. The operating state concept and the state variables allow for proper specification of initialisations (for example, at OPEN WORKSTATION) and the effect of various functions, especially with respect to the maintenance of device independence. One special set of functions called inquiry functions is allowed in all states. They give read-only access to the state lists. In this way useful information can be provided when errors occur. Other inquiry functions allow read-only access to the workstation descriptions, to allow the application program to adapt to particular workstation capabilities. Inquiry functions never cause errors. Instead they return information specifying whether a valid inquiry was made.

GKS provides an interface to a system for filing graphical information for the purpose of external long term storage and exchange. The interface consists of a GKS Metafile output workstation, which writes to a so-called graphics metafile (which is sequential), and a GKS Metafile input workstation, which reads from the metafile. In addition to the normal functions for output to workstations, a GKS Metafile output workstation may accept items containing non-graphical information. Input from a metafile is controlled by read and interpret functions which have the same effect as invoking the corresponding functions directly from the application program.
4.4 Graphical output

4.4.1 Output primitives

The graphical information that is generated by GKS and routed to all active workstations is built up of basic pieces called output primitives. GKS provides six output primitives:

a) POLYLINE: GKS generates a set of connected lines defined by a point sequence.
b) POLYMARKER: GKS generates symbols of one type centred at given positions.
c) TEXT: GKS generates a character string at a given position.
d) FILL AREA: GKS generates a polygonal area which may be hollow or filled with a uniform colour, a pattern, or a hatch style.
e) CELL ARRAY: GKS generates an array of pixels with individual colours.
f) GENERALIZED DRAWING PRIMITIVE(GDP): GKS addresses special geometrical output capabilities of a workstation such as the drawing of spline curves, circular arcs, and elliptic arcs. The objects are characterized by an identifier, a set of points and additional data. GKS applies all transformations to the points but leaves the interpretation to the workstation.

4.4.2 Output primitive attributes

Each output primitive potentially has three types of attribute (geometric, non-geometric and identification). The first two attribute types determine the exact appearance of the output primitive while the third attribute type is used in connection with input. The values of these attributes are set modally and are recorded in the GKS state list. A separate GKS function is provided for each primitive attribute (except the ASFs: see later in this subsection), to allow the application program to specify the value of an attribute without unnecessarily specifying the values of other attributes. During creation of an output primitive (that is, when one of the GKS output primitive functions is invoked) these values are bound to the primitive and cannot be changed afterwards.

Attributes of the first type control the geometric aspects of primitives; these are aspects which affect the shape or size of the whole primitive (for example, CHARACTER HEIGHT for TEXT). Hence, they are sometimes referred to as geometric attributes. Attributes of this type are workstation independent and, if they represent coordinate data (points or displacements), are expressed in world coordinates (for example, CHARACTER HEIGHT is expressed in world coordinates but TEXT PATH takes one of a set of enumerated values). They are defined separately for each primitive and a primitive may have zero, one or many geometric attributes.

Current values of (workstation independent) geometric attributes, which are expressed in world coordinates, are stored in world coordinates. When they are bound to their respective primitives, the values are subject to the same transformations as the geometric data contained in the definition of the primitive. Hence, current values are unaffected by changes in the normalization transformation and the workstation transformation.

Attributes of the second type control the non-geometric aspects of primitives; these are aspects
which merely affect a primitive’s appearance (for example, linetype for POLYLINE, or colour index for all primitives except CELL ARRAY) or the shape or size of the component parts of the primitive (for example, marker size scale factor for POLYMARKER). Non-geometric aspects do not represent coordinate data. The non-geometric aspects of a primitive may be specified in one of two ways, namely via a bundle or individually.

For specification of aspects via a bundle, there is one attribute per primitive, called the `<primitive>` INDEX. This attribute is an index into a bundle table, each entry of which contains all the non-geometric aspects of the primitive. There is a separate bundle table for each primitive with the exception of GENERALIZED DRAWING PRIMITIVE and CELL ARRAY (see later in this subsection). The non-geometric aspects are workstation dependent in this method of specification and each workstation has its own set of bundle tables (stored in the workstation state list). The values in a particular bundle (or entry in the bundle table) may be different for different workstations.

For individual specification of aspects, there is a separate attribute for each non-geometric aspect. As with the attributes controlling the geometric aspects, these attributes are workstation independent and are stored in the GKS state list. Since each non-geometric aspect only occurs in one primitive bundle type, each of these attributes applies to only one primitive type.

For a given non-geometric aspect, the values that can be assigned to the appropriate bundle component are the same as the values that can be assigned to the corresponding attribute for individual specification. Since the bundles are set separately for each workstation, the values of their components are restricted to the valid values for that workstation. In the case of individual attribute specification, such restrictions are not imposed. Default actions for the display of a primitive are defined to occur if it is created with a value of an individually specified attribute that is invalid on a particular workstation.

As indicated above, GENERALIZED DRAWING PRIMITIVE (GDP) and CELL ARRAY do not have associated bundle tables nor corresponding individually specified attributes. The GDP may use the most appropriate bundle tables or sets of individually specified attributes for each GDP function. For example, if one GDP function is essentially a FILL AREA, then the fill area bundle table or the set of individually specified fill area attributes would be used. CELL ARRAY contains colour index information as part of its definition but has no other non-geometric aspects and so does not use a bundle table nor does it have a set of individually specified attributes.

The method of specification of the non-geometric aspects of a primitive may be chosen separately for each aspect. A further group of primitive attributes, called ASPECT SOURCE FLAGS (ASFs), take the values INDIVIDUAL and BUNDLED to specify the choice. As with the other primitive attributes, these attributes are workstation independent and are stored in the GKS state list. There is one ASF for each non-geometric aspect of each primitive. The initial values of all the ASFs are the individual. The flags may be set at any time when GKS is open by the function SET ASPECT SOURCE FLAGS. This enables some non-geometric aspects of a primitive to be specified individually and others via a bundle.

When a primitive is displayed, the values of the non-geometric aspects with which it is displayed are determined as follows.

a) If the ASF for an aspect is INDIVIDUAL, the value used on all workstations is the value of the corresponding individually specified attribute of that primitive.

b) If the ASF for an aspect is BUNDLED, the value used on a workstation is obtained via the bundle table for that primitive on the workstation; the corresponding component of the bundle, pointed to by the bundle index, is used.
Colour is specified as an index into a separate colour table. There is only one colour table per workstation into which all the colour indices point. Similarly, other entries in a bundle, or corresponding individually specified attributes, may be indices either into another workstation table (for example, style index when interior style PATTERN is used) or into a fixed list (for example, linetypes for polyline).

There is precisely one attribute of the third type per primitive, namely PICK IDENTIFIER. This is used for identifying a primitive, or a group of primitives, in a segment when that segment is picked.

The attributes which apply to each output primitive (attributes controlling non-geometric aspects, geometric attributes and PICK IDENTIFIER) are:

c) POLYLINE:
   POLYLINE INDEX
   LINETYPE
   LINELength SCALE FACTOR
   POLYLINE COLOUR INDEX
   LINETYPE ASF
   LINELength SCALE FACTOR ASF
   POLYLINE COLOUR INDEX ASF
   PICK IDENTIFIER

d) POLYMARKER:
   POLYMARKER INDEX
   MARKER TYPE
   MARKER SIZE SCALE FACTOR
   POLYMARKER COLOUR INDEX
   MARKER TYPE ASF
   MARKER SIZE FACTOR ASF
   POLYMARKER COLOUR INDEX ASF
   PICK IDENTIFIER

e) TEXT:
   TEXT INDEX
   TEXT FONT AND PRECISION
   CHARACTER EXPANSION FACTOR
   CHARACTER SPACING
   TEXT COLOUR INDEX
   TEXT FONT AND PRECISION ASF
   CHARACTER EXPANSION FACTOR ASF
   CHARACTER SPACING ASF
   TEXT COLOUR INDEX ASF
   CHARACTER HEIGHT
   CHARACTER UP VECTOR
   TEXT PATH
   TEXT ALIGNMENT
   PICK IDENTIFIER

f) FILL AREA:
   FILL AREA INDEX
   FILL AREA INTERIOR STYLE
   FILL AREA STYLE INDEX
   FILL AREA COLOUR INDEX
   FILL AREA INTERIOR STYLE ASF
   FILL AREA STYLE INDEX ASF
   FILL AREA COLOUR INDEX ASF
   PATTERN SIZE
The Graphical Kernel System

The attributes for each primitive, other than PICK IDENTIFIER, are described in 4.4.3 to 4.4.8. PICK IDENTIFIER is described in more detail in 4.7.1. In the descriptions, attributes appear in upper case (for example, the attributes CHARACTER HEIGHT and PICK IDENTIFIER); aspects appear in both upper and lower case, according to their context. Geometric aspects are always controlled by geometric attributes and so appear in upper case (for example, the aspect CHARACTER HEIGHT). Non-geometric aspects may be controlled via a <primitive> INDEX or by individually specified attributes. Non-geometric aspects appear in lower case unless the corresponding individually specified attributes are being used which appear in upper case (for example, the aspect linetype but the individually specified attribute LINE-TYPE).

The entries in the bundle, pattern, and colour tables may be set separately for each workstation. Some standard definitions for table entries are contained in the workstation description table and are used as initial values. The application program may select a standard definition or may define the values of a specific entry explicitly. Only the most commonly used (or anticipated) combinations of values need be predefined for each output type workstation. At least those predefined entries with indices up to the minimum number of predefined entries at a given level (see 4.10.3) are distinguishable from each other. Other combinations of values can be specified by the SET <primitive | PATTERN | COLOUR> REPRESENTATION function, possibly after inquiring the workstation capabilities. The tables, which are on every workstation of category OUTPUT, OUTIN or MO (i.e. they are workstation attributes), are:

polyline bundle table
polymarker bundle table
text bundle table
fill area bundle table
pattern table
colour table

The values in these tables may be (dynamically) changed. In fact, the only way of changing the aspects of a primitive which are stored in a bundle table is by changing that table. However, note that a change in a bundle table entry can only be reflected in a displayed primitive if the values of the corresponding ASFs (of that primitive) for the aspects in the bundle table are BUNDLED. The entry 'dynamic modification accepted' in the workstation description table indicates which changes:

i) lead to an implicit regeneration (may be deferred) (IRG);

j) can be performed immediately (IMM).

The deferral state is explained in more detail in 4.5.3. If changes can be performed immediately, those changes may affect primitives outside segments in addition to those inside segments.
4.4.8 Polyline attributes

Polyline has no geometric attributes. The representation of polyline on the workstation is controlled by the POLYLINE INDEX, or the set of individually specified polyline attributes (LINETYPE, LINEWIDTH SCALE FACTOR, and POLYLINE COLOUR INDEX) or some combination of the two, depending upon the values of the ASFs for linetype, linewidth scale factor and polyline colour index. The POLYLINE INDEX is a pointer into the polyline bundle table, each entry of which contains values for linetype, linewidth scale factor and polyline colour index.

Linetypes 1 to 4 are solid, dashed, dotted and dashed-dotted. Every workstation of category OUTPUT or OUTIN realizes linetypes 1 to 4 with recognizable styles. Linetypes greater than 4 are reserved for registration or future standardization. Linetypes less than 0 may be available but their styles are implementation dependent. The linetype specifies a sequence of line segments and gaps which are repeated to draw a polyline. It is workstation dependent whether this sequence is restarted or continued at the start of the polyline, at the start of a clipped piece of a polyline, and at each vertex of a polyline.
Figure 2 - Binding of attributes

Note: PICK IDENTIFIER is separated from the other attributes for a primitive only in this figure. See 4.4.2 for a full list.
The Graphical Kernel System

The linewidth is calculated as a nominal linewidth multiplied by the linewidth scale factor. This value is mapped by the workstation to the nearest available linewidth.

4.4.4 Polymarker attributes

Polymarker has no geometric attributes. The representation of polymarker at the workstation is controlled by the POLYMARKER INDEX, or the set of individually specified polymarker attributes (MARKER TYPE, MARKER SIZE SCALE FACTOR, and POLYMARKER COLOUR INDEX) or some combination of the two, depending upon the values of the ASFs for marker type, marker size scale factor, and polymarker colour index. The POLYMARKER INDEX is a pointer into the polymarker bundle table, each entry of which contains values for marker type, marker size scale factor and polymarker colour index.

Marker types 1 to 5 are dot, plus sign, asterisk, circle, and diagonal cross each centred on the positions they are identifying. Every workstation of category OUTPUT or OUTIN realizes marker types 1 to 5 with recognizable shapes at the given positions. Marker types greater than 5 are reserved for registration or future standardization. Marker types less than 0 may be available but their forms are implementation dependent.

The marker size is calculated as a nominal size multiplied by the marker size scale factor. This size is mapped by the workstation to the nearest available size. Marker type 1 is always displayed as the smallest displayable dot.

The marker is visible if, and only if, the marker position is within the clipping rectangle. The clipping of partially visible markers is workstation dependent.

4.4.5 Text attributes

Text has the geometric attributes CHARACTER HEIGHT, CHARACTER UP VECTOR, TEXT PATH, and TEXT ALIGNMENT which are specified and used as described in this subsection.

Text also has two implicitly specified geometric attributes CHARACTER WIDTH and CHARACTER BASE VECTOR. These are implicitly specified by the functions SET CHARACTER HEIGHT and SET CHARACTER UP VECTOR respectively. They otherwise behave like ordinary geometric attributes (their values are bound to TEXT primitives when the primitives are created and cannot be changed afterwards and these values are subject to the same transformations as the geometric data contained in the definition of the primitive).

The representation of text at the workstation is controlled by the TEXT INDEX, or the set of individually specified text attributes (TEXT FONT AND PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, and TEXT COLOUR INDEX) or some combination of the two, depending upon the values of the ASFs for text font and precision, character expansion factor, character spacing and text colour index. The TEXT INDEX is a pointer into the text bundle table, each entry of which contains values for text font and precision, character expansion factor, character spacing and text colour index.

Precise control of the appearance of TEXT on a workstation is provided by the following aspects: CHARACTER HEIGHT, CHARACTER WIDTH, character expansion factor, TEXT PATH, CHARACTER UP VECTOR, CHARACTER BASE VECTOR, character spacing and TEXT ALIGNMENT. However, the use of these values in displaying text is determined by the setting of the text font and precision aspect (font and precision are two components of the same aspect). The CHARACTER HEIGHT specifies the nominal height of a capital letter character. The CHARACTER WIDTH specifies the nominal width of a character; the actual width depends upon the width to height ratio of the character indicated by the font designer and may vary from character to character. The character expansion factor specifies the
deviation of the width to height ratio of the character from the ratio indicated by the font designer. The CHARACTER UP VECTOR gives the up direction of a character. The CHARACTER BASE VECTOR gives the direction of the baseline of a character. Only the directions, not the lengths, of these vectors are relevant. TEXT PATH has the possible values RIGHT, LEFT, UP and DOWN. It specifies the writing direction of the text string. For RIGHT, the text string is written along a baseline in the direction of the CHARACTER BASE VECTOR. For LEFT, the baseline direction is the opposite direction of the CHARACTER BASE VECTOR. For UP, the character path coincides with the direction of the CHARACTER UP VECTOR. For DOWN, it is the opposite direction of the CHARACTER UP VECTOR. For the UP and DOWN text path directions the characters are arranged so that the centres of the character bodies are on a straight line in the direction of the CHARACTER UP VECTOR.

The character spacing value specifies how much additional space is to be inserted between two adjacent character bodies. If the value of character spacing is zero, the character bodies are arranged one after the other along the TEXT PATH, without any additional space between. A positive value of character spacing will insert additional space between character bodies. A negative value of character spacing will cause adjacent character bodies to overlap. Character spacing is specified as a fraction of the font nominal character height.

The effect of the aspects CHARACTER HEIGHT, CHARACTER WIDTH, character expansion factor, TEXT PATH, character spacing and text font is to define an (imaginary) rectangle with its sides parallel to the x and y axes, enclosing the text. The bounds of this enclosing rectangle are as follows. For TEXT PATH = LEFT or RIGHT, the height of the rectangle is the height of the character body of the specified font; the left side of the rectangle is the left side of the character body of the leftmost character and the right side of the rectangle is the right side of the character body of the rightmost character. For TEXT PATH = UP or DOWN, the top of the rectangle is the top of the character body of the topmost character and similarly, the bottom of the rectangle is the bottom of the bottommost character; the width of the rectangle is the width of the widest character in the specified font.

The effect of the CHARACTER UP VECTOR and CHARACTER BASE VECTOR attributes is to transform the enclosing rectangle, thus defining an enclosing parallelogram, the text extent parallelogram (the rectangle has been rotated and sheared).
The TEXT ALIGNMENT attribute controls the positioning of this text extent parallelogram in relation to the text position. For simplicity the TEXT ALIGNMENT is described in terms of the default CHARACTER UP VECTOR and CHARACTER BASE VECTOR, when the text extent parallelogram is actually a rectangle. The horizontal component of TEXT ALIGNMENT has four values: LEFT, CENTRE, RIGHT and NORMAL. If the horizontal component is LEFT, the left side of the text extent parallelogram passes through the text position. Similarly, if the value is RIGHT, the right side of the text extent parallelogram passes through the text position. If the horizontal component is CENTRE, the text position lies midway between the left and right sides of the text extent parallelogram. Thus, if TEXT PATH = UP or DOWN, the straight line passing through the centrelines of the characters also passes through the text position. The vertical component of TEXT ALIGNMENT has six values: TOP, CAP, HALF, BASE, BOTTOM and NORMAL. These each correspond to one of the font specific horizontal lines in the definition of a character (see figure 3). A value of TOP causes the top of the text extent parallelogram to pass through the text position. A value of CAP causes the text position to lie on the capline of the whole string (TEXT PATH = LEFT or RIGHT) or on the capline of the topmost character in the string (TEXT PATH = UP or DOWN). A value of HALF causes the text position to lie on the halfline of the whole string (TEXT PATH = LEFT or RIGHT) or on a line halfway between the halflines of the top and bottom characters (TEXT PATH = UP or DOWN). A value of BASE causes the text position to lie on the baseline of the whole string (TEXT PATH = LEFT or RIGHT) or on the baseline of the bottom character in the string (TEXT PATH = UP or DOWN). A value of BOTTOM causes the bottom of the text extent parallelogram to pass through the text position.

In the general case, the orientation referred to as horizontal is that of the CHARACTER BASE VECTOR with RIGHT representing direction of that vector and LEFT being opposite to it. Similarly the orientation referred to as vertical is that of the CHARACTER UP VECTOR with UP representing the direction of that vector and DOWN being opposite to it.
Either component of TEXT ALIGNMENT can take the value NORMAL. For each value of TEXT PATH, the effect of a particular component being NORMAL is equivalent to one of the other values of that component. In each case, the equivalent alignment value is chosen to achieve a natural alignment for that TEXT PATH value. The complete list of equivalent values is:

<table>
<thead>
<tr>
<th>TEXT PATH</th>
<th>NORMAL Horizontal and Vertical Alignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIGHT</td>
<td>(LEFT, BASE)</td>
</tr>
<tr>
<td>LEFT</td>
<td>(RIGHT, BASE)</td>
</tr>
<tr>
<td>UP</td>
<td>(CENTRE, BASE)</td>
</tr>
<tr>
<td>DOWN</td>
<td>(CENTRE, TOP)</td>
</tr>
</tbody>
</table>

The initial values of the geometric text attributes are:

<table>
<thead>
<tr>
<th>CHARACTER HEIGHT</th>
<th>WC</th>
<th>0.01 (ie 1% of the height of the default window)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARACTER UP VECTOR</td>
<td>WC</td>
<td>(0,1)</td>
</tr>
<tr>
<td>TEXT PATH</td>
<td>WC</td>
<td>RIGHT</td>
</tr>
<tr>
<td>TEXT ALIGNMENT</td>
<td>WC</td>
<td>(NORMAL, NORMAL)</td>
</tr>
</tbody>
</table>

and the initial values of the implicitly specified geometric text attributes are:

<table>
<thead>
<tr>
<th>CHARACTER WIDTH</th>
<th>WC</th>
<th>0.01 (ie the same value as the initial value of CHARACTER HEIGHT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARACTER BASE VECTOR</td>
<td>WC</td>
<td>(1,0)</td>
</tr>
</tbody>
</table>

Text font and precision together constitute one aspect. The text font value is used to select a particular font on the workstation. Every workstation supports at least one font that is able to generate a graphical representation of the characters defined in ANSI X3.4-1977 (commonly referred to as ASCII). This is font number 1. Font numbers greater than 1 are reserved for registration or future standardization. Font numbers less than 0 may be supported but are implementation dependent.

The text precision value is used to select the 'closeness' of the text representation at the workstation in relation to that defined by the workstation independent text attributes and the transformation and clipping currently applicable. The text precision value has the following possible values:

a) STRING: The TEXT character string is generated in the requested text font and is positioned by aligning the TEXT output primitive at the given text position. CHARACTER HEIGHT, CHARACTER WIDTH and character expansion factor are evaluated as closely as reasonable, given the capabilities of the workstation. CHARACTER UP VECTOR, CHARACTER BASE VECTOR, TEXT PATH, TEXT ALIGNMENT and character spacing, need not be used. Clipping is done in an implementation and workstation dependent way.

b) CHAR: The TEXT character string is generated in the requested text font. For the representation of each individual character, the aspects CHARACTER HEIGHT, CHARACTER WIDTH, the up direction of the CHARACTER UP VECTOR, the baseline direction of the CHARACTER BASE VECTOR, and character expansion factor are evaluated as closely as possible, in a workstation dependent way. The spacing used between character bodies is evaluated exactly; the character body, for this purpose, is an ideal character body, calculated precisely from the text aspects and the font dimensions. The position of the resulting text extent parallelogram is determined by the TEXT ALIGNMENT and the text position. Clipping is performed at least on a character by character basis.
c) STROKE: The text character string in the requested text font is displayed at the text position by applying all text aspects. The character string is clipped exactly at the clipping rectangle.

STROKE precision does not necessarily mean vector strokes; as long as the representation adheres to the rules governing STROKE precision, the font may be realized in any form, for example by raster fonts.

A GKS output level m implementation need only support STRING text precision, and level 0 need only support text precisions STRING and CHAR. All text precisions must be supported above output level 0. A workstation may use a higher precision than the one requested for this purpose, i.e., if STROKE precision is supported in a particular font, the implication is that both STRING and CHAR precision are available in that font. However, it is not necessary for a workstation to support all precisions for a given font (i.e. for a given font, STROKE can be missing or both STROKE and CHAR can be missing). Text font and precision are workstation mandatory. That is, for any GKS level supporting a STROKE precision font, every workstation of a particular installation supports at least one STROKE precision text font. This is font number 1, containing the character set defined by ANSI X3.4-1977. This implies that, for STROKE precision text, some sort of software character generator is required for those implementations that have inadequate hardware. Not all workstations need to support all fonts, but for those that do, the same font number is used to select that font on all workstations of a particular installation.

Fonts are defined only within the GKS implementation. The font designer specifies the shape of the symbol representing each character in a local 2D cartesian font coordinate system. Fonts are either monospaced or proportionally spaced. Each character in a font coordinate system has an associated character body, a font baseline, a font halfline, a capline and a centreline (see figure 3). For monospaced fonts, the character bodies of all characters have the same size. For proportionally spaced fonts, the width of the bodies may differ from character to character. The character body edges are parallel to the axes of the font coordinate system. The font baseline, the font halfline and the capline are parallel to the x-axis of the font coordinate system, and within the vertical extent of the body. The position of the font halfline is defined by the font designer for use in aligning text strings. The centreline is parallel to the y-axis and bisects the body. Their exact positions are specified by the font designer.

The height of a character in the font coordinate system is given by the height from the font baseline to the capline. The width of a character is given by the width of the character body. The width of a character may include space on either side of the character and this space is generally evenly split between the left and right sides of the character. It is assumed that the characters lie within their body, except that kerned characters may exceed the side limits of the character body.

In general, the top limits of the bodies for a font are identical with, or very close to, the typographical capline or ascender line, and the bottom limit to the descender line. The space, if any, between the topline and the capline may be used for an additional mark over the character, for example an accent. However, these and other details are purely for the use of the font designer. The intention is only that characters placed with their bodies touching in the horizontal direction should give an appearance of good normal spacing, and characters touching in the vertical direction will avoid clashes between ascenders and descenders (typographically 'set solid').

Since the values of CHARACTER HEIGHT, CHARACTER WIDTH, CHARACTER UP VECTOR and CHARACTER BASE VECTOR are given in world coordinate units, but the characters are generated on the workstation in device coordinates, using the workstation dependent font and precision, the geometric attributes need to be transformed in such a way that the
workstation can generate the characters in the way intended.

The effect to be achieved is now described. Together with the text coding, a height vector parallel to the CHARACTER UP VECTOR with length equal to CHARACTER HEIGHT, and a width vector parallel to the CHARACTER BASE VECTOR with length equal to CHARACTER WIDTH, are passed down the viewing pipeline. These vectors are transformed by the normalization transformation, by a segment transformation if within a segment, and by the workstation transformation. They are also stored in segments. Then the vectors can be used by the workstation character generator. Thus, the shape of individual characters can be transformed by a normalization transformation that is unequal in x and y and by a segment transformation.

On the workstation, the height of a character is given by the length of the transformed height vector; the character up direction is given by the direction of the transformed height vector; the width of a character is given by the length of the transformed width vector multiplied by the font width to height ratio for the character and by the character expansion factor; the character base direction is given by the direction of the transformed width vector. The characters are arranged together in a text extent parallelogram, depending on the values of TEXT PATH and character spacing. The text extent parallelogram is then positioned according to the value of TEXT ALIGNMENT and the text position, contained in the definition of the TEXT primitive.

Figures 4 to 7 give examples of the effects of different values of text aspects. Figure 8 gives examples of the effect of different normalization transformations on the displayed form of the text.
Examples are illustrated with STROKE precision, a character expansion factor of 1 and a zero character spacing.

![Graphical output](image)

**CHARACTER HEIGHT** = 1, **CHARACTER UP VECTOR** = (0, 1), **TEXT PATH** = RIGHT, **TEXT ALIGNMENT** = (NORMAL, NORMAL)

**CHARACTER HEIGHT** = 0.5, **CHARACTER UP VECTOR** = (0, 1), **TEXT PATH** = RIGHT, **TEXT ALIGNMENT** = (NORMAL, NORMAL)

**CHARACTER HEIGHT** = 1, **CHARACTER UP VECTOR** = (0, 1), **TEXT PATH** = RIGHT, **TEXT ALIGNMENT** = (RIGHT, TOP)

**CHARACTER HEIGHT** = 1
**CHARACTER UP VECTOR** = (0, 1)
**TEXT PATH** = DOWN
**TEXT ALIGNMENT** = (NORMAL, NORMAL)

**CHARACTER HEIGHT** = 1
**CHARACTER UP VECTOR** = (−1, 0)
**TEXT PATH** = RIGHT
**TEXT ALIGNMENT** = (NORMAL, NORMAL)

× text position
----- baseline or centreline
----- text extent rectangle (indicated for PATH = DOWN)

Note: capline = topline in these examples

Figure 4. Effects of changes in geometric text attributes

NOTE: Changed attributes are underlined
Examples are illustrated with default values of the geometric text attributes and with STROKE precision.

**Figure 5. Effects of changes in non-geometric text aspects**

NOTE: Changed aspects are underlined
Note:
halflines of all characters are shown in this example

**Figure 6. Effects of combined changes in text aspects**

NOTE: Changes from the top example of figures 4 and 5 are underlined.
Figure 7. Effects of several changes in text aspects

NOTE: Changes from the top example of figures 4 and 5 are underlined.
Figure 8. Effects of different normalization transformations on text in STROKE precision
4.4.6 Fill area attributes

Fill area has the geometric attribute PATTERN REFERENCE POINT. It also has two implicitly specified geometric attributes PATTERN WIDTH VECTOR and PATTERN HEIGHT VECTOR. These are implicitly specified by the function SET PATTERN SIZE. Like ordinary geometric attributes, their values are bound to FILL AREA primitives when the primitives are created and cannot be changed afterwards and these values are subject to the same transformations as the geometric data contained in the definition of the primitive. The usage of the fill area geometric attributes is described later in this subsection.

The representation of fill area at the workstation is controlled by the FILL AREA INDEX, or the set of individually specified fill area attributes (FILL AREA INTERIOR STYLE, FILL AREA STYLE INDEX, and FILL AREA COLOUR INDEX) or some combination of the two, depending upon the values of the ASFs for fill area interior style, fill area style index, and fill area colour index. The FILL AREA INDEX is a pointer into the fill area bundle table, each entry of which contains values for the fill area interior style, fill area style index and fill area colour index.

The fill area interior style is used to determine in what style the area should be filled. It has the following values:

a) HOLLOW: No filling, but draw the bounding polyline, using the fill area colour index currently selected (either via the fill area bundle or individually, depending upon the corresponding ASF). The linetype and linewidth are implementation dependent.

b) SOLID: Fill the interior of the polygon using the fill area colour index currently selected (either via the fill area bundle or individually, depending upon the corresponding ASF).

c) PATTERN: Fill the interior of the polygon using the fill area style index currently selected (either via the fill area bundle or individually, depending upon the corresponding ASF) as an index into the pattern table. In this context, the fill area style index is sometimes referred to as the pattern index.

d) HATCH: Fill the interior of the polygon using the fill area colour index and the fill area style index currently selected (either via the fill area bundle or individually, depending upon the corresponding ASFs). The fill area style index is used as a pointer into the list of hatch styles, in which case it is sometimes referred to as the hatch index.

For interior style PATTERN, the pattern is defined by the pattern representation, which specifies an array \((DX \times DY)\) of colour indices, that are pointers into the colour table. The size and position of the start of the pattern are determined by a pattern box. The pattern box, which is a parallelogram, is defined by the PATTERN WIDTH VECTOR and the PATTERN HEIGHT VECTOR located relative to the PATTERN REFERENCE POINT. The pattern box is conceptually divided into a grid of \(DX \times DY\) cells. The colour index array is associated with the cells as follows: the element \((1, DY)\) is associated with the cell having the PATTERN REFERENCE POINT at one corner; elements with increasing first dimension are associated with successive cells in the direction of the PATTERN WIDTH VECTOR; elements with decreasing second dimension are associated with successive cells in the direction of the PATTERN HEIGHT VECTOR. The attributes defining the pattern box are subject to all the transformations producing a transformed pattern box. The pattern is mapped onto the polygon by conceptually replicating the transformed pattern box in directions parallel to its sides until the interior of the complete polygon is covered.

Mapping the transformed pattern cells to the pixels of a raster display is performed by the following rules:
e) If the centre of a pixel lies inside the parallelogram defined by the transformed cell, its colour is set;
f) The pixel is assigned the colour of the cell corresponding to the pixel's centre.

For a workstation which can implement patterns but not transformable patterns, a suitable action is to generate non-transformed patterns to fill a polygon.

For interior style HATCH, the hatch index selects among hatch styles: hatch styles greater than 0 are reserved for registration or future standardization; hatch styles less than 0 are workstation dependent. Whether hatching is affected by transformations or not is workstation dependent.

Interior style HOLLOW is available on every workstation of category OUTPUT or OUTIN. It is workstation dependent which of the interior styles SOLID, PATTERN and HATCH are available.

4.4.7 Cell array attributes

Cell array has no attributes other than PICK IDENTIFIER. However, an array of colour indices, which are pointers into the colour table, is part of the definition of a cell array.

4.4.8 Generalized Drawing Primitive attributes

Generalized Drawing Primitive (GDP) has no explicit geometric attributes. Such information may be specified in the GDP data record. The representation of the GDP at the workstation is controlled by zero or more of the sets of polyline, polymarker, text and fill area attributes (see 4.4.2). Whether bundle indices or associated individually specified attributes are used depends upon the values of the appropriate ASFs. The sets of attributes most appropriate for the specified GDP function are selected for the GDP as part of the definition of the GDP and are recorded in the workstation description table. Consequently, if a GDP is essentially a cell array, then an array of colour indices would be specified in the GDP data record.

4.4.9 Colour

In GKS, colour is specified in a number of different situations. It may be an aspect of a primitive, in which case it is specified either in the bundle for that primitive or by the individual colour attribute for that primitive. It may be part of a pattern for FILL AREA, in which case an array of colours is specified, or it may be part of a primitive itself, namely CELL ARRAY, when an array of colours is also specified. In each case, the colour is specified as an index into a colour table on the workstation. On each workstation, there is one colour table into which all the colour indices point.

The size of the colour table is workstation dependent but entries 0 and 1 always exist. Entry 0 corresponds to the background colour. The background colour is the colour of the display surface after it has been cleared. Entry 1 is the default foreground colour and entries higher than 1 correspond to alternative foreground colours. Entries in the table may be set by the function SET COLOUR REPRESENTATION which specifies the colour as a combination of red, green, and blue intensities. The specified colour is mapped to the nearest available by the workstation. On some workstations it may not be possible to change the background colour, and in this case the mapping of a specific colour to the nearest available for the background colour may be different from the mapping of the same colour for the foreground colours.

Some workstations are not capable of displaying colours (for example, workstations only capable of displaying colours with equal red, green, and blue intensities or workstations only capable of displaying colours which are different intensities of the same colour); these are referred to as monochrome workstations. Whether a workstation is capable of colour is recorded in the 'colour available' entry in the workstation description table. On monochrome workstations, the intensity is computed from the colour values in a workstation dependent way. See Appendix D
for a recommended algorithm.
4.5 Workstations

4.5.1 Workstation characteristics

GKS is based on the concept of abstract graphical workstations. These provide the logical interface through which the application program controls physical devices. Certain special workstations provide facilities for the storage and exchange of graphical information.

For every type of workstation present in a GKS implementation (except for the special workstations), there exists a workstation description table which describes the capabilities and characteristics of the workstation. The application program can inquire which capabilities are available and adapt its behaviour accordingly. If a capability is requested that a particular workstation does not provide, a standard error reaction is defined. Certain minimal capabilities of a workstation are detailed in 4.10.

An abstract graphical workstation with maximum capabilities:

a) has one addressable display surface of fixed resolution;
b) allows only rectangular display spaces (the display space does not consist of a number of separate parts);
c) permits the specification and use of smaller display spaces than the maximum while guaranteeing that no display image is generated outside the specified display space;
d) supports several linetypes, text fonts, character sizes, etc, to allow output primitives to be drawn with different aspects;
e) has one or more logical input devices for each input class;
f) permits REQUEST, SAMPLE and EVENT type input;
g) allows logical input devices to be set in REQUEST, SAMPLE or EVENT mode independently of each other;
h) stores segments and provides facilities for changing and manipulating them.

In practice, the workstation is not necessarily equipped with all of these capabilities.

Each workstation has a type. Each workstation type falls into one of six categories:

<table>
<thead>
<tr>
<th>OUTPUT</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT</td>
<td>Input</td>
</tr>
<tr>
<td>OUTIN</td>
<td>Output and input</td>
</tr>
<tr>
<td>WISS</td>
<td>Workstation Independent Segment Storage (WISS)</td>
</tr>
<tr>
<td>MO</td>
<td>GKS Metafile (GKSM) output</td>
</tr>
<tr>
<td>MI</td>
<td>GKSM input</td>
</tr>
</tbody>
</table>

A workstation of category OUTPUT has only output capabilities. It can display all output primitives, with the possible exception of the GENERALIZED DRAWING PRIMITIVE which is optional. Minimal requirements for displaying TEXT and FILL AREA primitives are listed in 4.4, and for CELL ARRAY in 5.3.

GKS allows the appearance of output primitives to vary between workstations, thus allowing advantage to be taken of their differing capabilities. The facilities which allow this variation are:
polyline representation (see 4.4)
polymarker representation (see 4.4)
text representation (see 4.4)
fill area representation (see 4.4)
pattern representation (see 4.4)
colour representation (see 4.4)
deferral state (see 4.5.3)
workstation transformation (see 4.6.3)

Figure 2 (see 4.4) illustrates the binding of the workstation attributes.

A workstation of category INPUT has at least one logical input device, but no output capability.

A workstation of category OUTIN has the characteristics of both an OUTPUT and INPUT workstation. In addition, the existence of a workstation in this category in a GKS implementation gives rise to additional requirements regarding logical input devices (see 4.8.1).

The last three categories WISS, MO and MI are special GKS facilities that provide a means for temporarily or permanently storing graphical information. They are treated as workstations for the purposes of control, but otherwise have quite different characteristics (see 4.7.5 and 4.9).

Section A.5 of appendix A gives a complete listing of all GKS functions which apply directly or indirectly to each category of workstation.

Actual workstations may provide more capabilities than those listed in the workstation description table. These cannot be used by GKS. However, if the workstation itself provides sufficient intelligence, the additional capabilities may be accessed via the GENERALIZED DRAWING PRIMITIVE or ESCAPE functions, or used locally by the workstation operator. As an example, if a workstation has two display surfaces, the operator may switch locally from one to the other without notifying GKS or the application program. More than one display surface can be controlled by GKS only by defining a separate workstation for each display surface.

4.5.2 Selecting a workstation

The application program references a workstation by means of a workstation identifier. Connection to a particular workstation is established by the function OPEN WORKSTATION, which associates the workstation identifier with a workstation type and a connection identifier. The current state of each open workstation is kept in a workstation state list. Segment manipulation and input can be performed on all open workstations. Output primitives are sent to, and segments are stored on, all active workstations and no others; an open workstation is made active by the function ACTIVATE WORKSTATION.

An active workstation is made inactive by the function DEACTIVATE WORKSTATION; an open workstation is closed by the function CLOSE WORKSTATION.

The following sequence of functions illustrates workstation selection:
4.5.9 Deferring picture changes

It is desirable that the display of a workstation reflects, as far as possible, the actual state of the picture as defined by the application program. However, to use the capabilities of a workstation efficiently, GKS allows a workstation to delay, for a certain period of time, the actions requested by the application program. During this period, the state of the display may be undefined.

The function SET DEFERRAL STATE allows the application program to choose that deferral state which takes into account the capabilities of the workstation and the requirements of the application program. Two attributes are defined for this purpose. Deferral mode controls the time at which output functions have their visual effects. Implicit regeneration controls the time at which picture changes have their visual effects: picture changes in general imply an alteration not just an addition to the picture.

The concept of deferral refers only to visible effects of GKS functions. Effects on the segment storage or on the state of the workstation are (conceptually) not deferred.

Deferral mode controls the possible delaying of output functions: for example, data sent to a device may be buffered to optimize data transfer. The values of deferral mode (in increasing order of delay) are:

a) ASAP: The visual effect of each function will be achieved on the workstation As Soon As Possible (ASAP). GKS ensures that the actions necessary to achieve this visual effect are initiated before control is returned to the application program, but, owing to possible delays not under the influence of GKS, the actions are not necessarily completed before control is returned.

b) BNIG: The visual effect of each function will be achieved on the workstation Before the Next Interaction Globally (BNIG), i.e. before the next interaction with a logical input device gets underway on any workstation (see 4.8.2). If an interaction on any workstation is already underway, the visual effect will be achieved as soon as possible.
c) BNIL: The visual effect of each function will be achieved on the workstation Before the Next Interaction Locally (BNIL), i.e. before the next interaction with a logical input device gets underway on that workstation (see 4.8.2). If an interaction on that workstation is already underway, the visual effect will be achieved as soon as possible.

d) ASTI: The visual effect of each function will be achieved on the workstation At Some Time (ASTI).

Deferral applies to the following functions that generate output:

POLYLINE
POLYMARKER
TEXT
FILL AREA
CELL ARRAY
GENERALIZED DRAWING PRIMITIVE
INSERT SEGMENT
ASSOCIATE SEGMENT WITH WORKSTATION
COPY SEGMENT TO WORKSTATION
INTERPRET ITEM

For none of the possible values of deferral mode is it mandatory for an implementation to delay the visual effect of output functions. If the application program requires a delay, it can achieve this using the segment storage facility and the visibility attribute. This restriction means that the buffer for deferred actions can be chosen in an implementation dependent manner.

Certain functions can be performed immediately on some workstations, but on other workstations imply a regeneration of the whole picture to achieve their effect. For example, an implicit regeneration is necessary when picture changes require new paper to be put on a plotter. The entries 'dynamic modification accepted' in the workstation description table indicate which changes:

e) lead to an implicit regeneration (IRG);
f) can be performed immediately (IMM).

If changes can be performed immediately, those changes may affect primitives outside segments in addition to those inside segments. If regeneration occurs, all primitives outside segments will be deleted from the display surface.

An implicit regeneration is equivalent to an invocation of the function REDRAW ALL SEGMENTS. Its possible delay is controlled by the implicit regeneration mode, a single entry in the workstation state list. The values of implicit regeneration mode are:

g) SUPPRESSED: Implicit regeneration of the picture is suppressed, until it is explicitly requested: the entry 'new frame necessary at update' is set to YES.

h) ALLOWED: Implicit regeneration of the picture is allowed.

An implicit regeneration is made necessary, if any of the following occur:

i) if the functions listed below have a visible effect on the display image of the respective workstation:

1) if the 'dynamic modification accepted' entry in the workstation description table is IRG (implicit regeneration necessary) for the specified representation:
2) if the 'dynamic modification accepted' entry in the workstation description table is IRG for the workstation transformation:

   SET WORKSTATION WINDOW
   SET WORKSTATION VIEWPORT

3) if the 'dynamic modification accepted' entry in the workstation description table is IRG for segment priority and this workstation supports segment priority:

   i) if primitives are added to open segment overlapping a segment of higher priority:

      POLYLINE
      POLY MARKER
      TEXT
      FILL AREA
      CELL ARRAY
      GENERALIZED DRAWING PRIMITIVE
      INSERT SEGMENT

      (since only segments have priority, primitives outside segments do not make an implicit regeneration necessary.)

   ii) if the complete execution of one of the following actions would be affected by segment priority:

      DELETE SEGMENT
      DELETE SEGMENT FROM WORKSTATION
      ASSOCIATE SEGMENT WITH WORKSTATION
      SET SEGMENT TRANSFORMATION
      SET VISIBILITY
      SET SEGMENT PRIORITY

4) if the 'dynamic modification accepted' entry in the workstation description table is IRG for segment transformation:

   SET SEGMENT TRANSFORMATION

5) if the 'dynamic modification accepted' entry in the workstation description table is IRG for 'visibility (visible \(\rightarrow\) invisible)'

   SET VISIBILITY (INVISIBLE)

6) if the 'dynamic modification accepted' entry in the workstation description table is IRG for 'visibility (invisible \(\rightarrow\) visible)'

   SET VISIBILITY (VISIBLE)

7) if the 'dynamic modification accepted' entry in the workstation description table is IRG for highlighting:

   SET HIGHLIGHTING
8) if the 'dynamic modification accepted' entry in the workstation description table is 
IRG for delete segment:

DELETE SEGMENT
DELETE SEGMENT FROM WORKSTATION

j) if any of the above situations occurs as a result of INTERPRET ITEM.

An implicit regeneration has to be done (including deletion of primitives outside segments) 
only if one of the functions listed causes a visible effect on the display; for example, if an 
invisible segment is deleted, a regeneration need not be done. However, an implementation is 
allowed to perform an implicit regeneration in any of the cases listed above.

Deferred actions can be made visible at any time by the use of the UPDATE WORKSTATION 
function or by an appropriate change of the deferral state.

4.5.4 Clearing the display surface

Two capabilities for clearing the display surface are recognized, namely:

a) clear the display surface even if it is empty;

b) ensure that the display surface is clear without clearing the display surface needlessly.

The second capability means that the display surface is only cleared when needed: this would 
normally be when the display surface is not clear (i.e. when the 'display surface empty' entry in 
the workstation state list is NOTEMPTY). The entry 'display surface empty' in the workstation 
state list is set to NOTEMPTY if output is sent to the device. It may be set to NOTEMPTY 
even if output does not appear on the display surface (for example, a GDP primitive which has 
been clipped at the device to non-existence).

Both capabilities for clearing the display surface are available to the user through the function 
CLEAR WORKSTATION. The second capability is also used in UPDATE WORKSTATION 
and REDRAW ALL SEGMENTS ON WORKSTATION.

4.5.5 Elimination of primitives outside segments

Elimination of primitives outside segments occurs in the following situations:

a) if the following GKS functions are invoked:

CLEAR WORKSTATION
REDRAW ALL SEGMENTS ON WORKSTATION
UPDATE WORKSTATION
if the parameter update regeneration flag is PERFORM and if 'new frame action necessary at update' in the workstation state list is YES;

SET DEFERRAL STATE
if 'implicit regeneration mode' is ALLOWED and 'new frame action necessary at update' is YES;

b) if an implicit regeneration is made necessary (see 4.5.3) and 'implicit regeneration mode' is ALLOWED;

c) if any of the above situations occurs as a result of INTERPRET ITEM.

4.5.6 Sending messages to a workstation

The MESSAGE function allows a character string to be sent to a workstation. The application 
program has no control over the position and appearance of the character string and an implement-
ation is allowed to place the string on a device distinct from, but associated with, the 
workstation. The rules to be followed by an implementation are stated in 5.2.
4.6 Coordinate systems and transformations

4.6.1 Normalization transformations

In GKS, the application programmer can compose his graphical picture from separate parts each of which, conceptually, is defined with its own world coordinate system (WC). The relative positioning of the separate parts is defined by having a single normalized device coordinate space (NDC) onto which all the defined world coordinate systems are mapped. A set of normalization transformations define the mappings from the world coordinate systems onto the single normalized device coordinate space, which can be regarded as a workstation independent abstract viewing surface. This normalized picture can be stored and manipulated via the segment mechanism; it can also be stored on a metafile.

For output, a single normalization transformation is current at any one time and this is used to transform world coordinates specified, for example in output primitives and geometric attributes, into normalized device coordinates.

A normalization transformation is specified by defining the limits of an area in the world coordinate system (window) which is to be mapped onto a specified area of the normalized device coordinate space (viewport). Window and viewport limits specify rectangles parallel to the coordinate axes in WC and NDC. The rectangles include their boundaries. The normalization transformation performs a mapping from WC onto NDC that includes translation and differential scaling with positive scale factors for the two axes.

Although NDC space conceptually extends to infinity, the part of NDC space in which the viewport needs to be located and that can be viewed at a workstation is the closed range \([0,1] \times [0,1]\). In addition, an implementation may support only a restricted range of NDCs. However, this range is always sufficiently greater than the \([0,1] \times [0,1]\) square that useful effects of INSERT SEGMENT can be achieved. In particular, NDCs in the range \([-7,7] \times [-7,7]\) are always handled.

Each normalization transformation is identified by a transformation number which is an integer between 0 and an implementation dependent value \(n\) which can be inquired from the GKS state list. The normalization transformation with transformation number 0 is the unity transformation which maps \([0,1] \times [0,1]\) in world coordinates to \([0,1] \times [0,1]\) in normalized device coordinates. It cannot be changed.

Initially, all other normalization transformations are set to a default transformation which is the same as transformation number 0. Different transformations can be specified at any time when GKS is open. Since GKS provides a number of different normalization transformations, it is possible for the application program to specify them prior to outputting the graphical picture. The separate parts of the picture are output by selecting a particular normalization transformation before outputting the associated graphical primitives. However, specifying a normalization transformation, while the graphical output is taking place, is allowed.

A normalization transformation may be selected by SELECT NORMALIZATION TRANSFORMATION, and it will be used for all output until another is selected. By default, normalization transformation 0 is selected.

4.6.2 Clipping

The viewport of a particular normalization transformation is used to define a clipping rectangle, as well as, with the window, specifying the normalization transformation. When the viewport of the current normalization transformation is set or when a normalization transformation is selected, the ‘clipping rectangle’ entry in the GKS state list is set to the resulting viewport of the current normalization transformation. Clipping to this clipping rectangle can either be
enabled or disabled. There is a single global switch (the clipping indicator) which defines whether or not the clipping rectangle is to be used for clipping.

Clipping does not take place when the normalization transformation is performed but is delayed until the output primitives are to be displayed on the display surface of a workstation. Output primitives stored in segments have their coordinates transformed to NDC and the associated clipping rectangle is stored with the primitives. The INSERT SEGMENT function allows the clipping rectangle in the GKS state list to replace the clipping rectangle, stored with an output primitive when the segment was defined.

Primitives sent to a workstation of category MO are not clipped.

4.6.3 Workstation transformations
The normalized device coordinate space can be regarded as a workstation independent abstract viewing surface. Each open workstation can select independently some part of the NDC space in the range \([0,1] \times [0,1]\) to be displayed somewhere on the workstation display surface. A particular workstation transformation is a mapping from NDC space onto the device coordinates (DC) for that workstation.

The units of device coordinates are metres on a device capable of producing a precisely scaled image (for example, on most plotters) and appropriate workstation dependent units otherwise (for example, on a display unit with unknown monitor size). In either case the device coordinate system maps onto the display space in the following way:

a) the DC origin is at the bottom left corner of the display space;
b) the device coordinate units are related to the display space in such a way that a square in device coordinates appears as a square on the display surface (this is trivially true if device coordinate units are metres);
c) \(x\) and \(y\) increase to the right and upwards respectively.

On some devices, device coordinate units do not coincide with addressable units, for example if the addressable units do not satisfy the above conditions.

The size of the display space in device coordinate units is recorded in the workstation description table.

The workstation transformation is a uniform mapping from NDC onto DC and thus performs translation and equal scaling with a positive scale factor for the two axes. Thus picture composition can be achieved using the normalization transformations whereas the workstation transformation allows different aspects of the composed picture to be viewed on different workstations. For example, a drawing could be output to a plotter at the correct scale and simultaneously some part of the drawing could be displayed on the full display surface of an interactive terminal.

The workstation transformation can be specified at any time after the workstation has been opened. However, actually changing the workstation transformation may cause an implicit regeneration of the picture.

A workstation transformation is specified by defining the limits of an area in the normalized device coordinate system within the range \([0,1] \times [0,1]\) (workstation window) which is to be mapped onto a specified area of the device coordinate space (workstation viewport). Workstation window and workstation viewport limits specify rectangles parallel to the coordinate axes in NDC and DC. The rectangles include their boundaries. To ensure that no output outside the workstation window is displayed, GKS clips at the workstation window boundaries, and this clipping cannot be disabled. As the workstation window is defined somewhere in the NDC
range $[0,1] \times [0,1]$, this ensures that the only part of NDC space that can be viewed on any workstation lies in the range $[0,1] \times [0,1]$. If the workstation window and workstation viewport have different aspect ratios, the scaling specified would be different on each axis if the workstation window was mapped onto the workstation viewport in its entirety. To ensure equal scaling on each axis, the workstation transformation maps the workstation window onto the largest rectangle that can fit within the workstation viewport such that:

- d) aspect ratio is preserved;
- e) the lower left-hand corner of the workstation window is mapped to the lower left-hand corner of the workstation viewport.

Thus, space is left unused at the top or right side of the workstation viewport if the aspect ratios of the workstation window and workstation viewport are different.

All workstation transformations are set by default to map NDC space $[0,1] \times [0,1]$ onto the whole of the workstation display space. If the display space is not square, the same rules as above apply to achieve equal scaling on each axis.

Workstation transformations can be changed by SET WORKSTATION WINDOW or SET WORKSTATION VIEWPORT. As a specification of the workstation transformation may be deferred (see 4.5.3), these two functions only set the 'requested workstation window' and 'requested workstation viewport' entries in the workstation state list. The 'current workstation window' and 'current workstation viewport' entries continue to hold the previously set transformation parameters. When the display is updated, the current values are set to the requested values.

A complete data flow chart for graphical output is given in figure 9. It should be noted that all three coordinate systems (WC, NDC and DC) are two dimensional Cartesian coordinate systems.
Figure 9. Data flow chart for graphical output in GKS
4.6.4 Transformation of locator input

The application programmer requires LOCATOR input to define a position in the most appropriate world coordinate system currently defined by the set of normalization transformations.

This is achieved by first transforming the input data from DC to NDC by the inverse workstation transformation which is in effect when LOCATOR input is generated. LOCATOR input can only be obtained from positions within the part of the current workstation viewport into which the current workstation window is mapped (note that this is a subset of the workstation viewport whenever the aspect ratio of the workstation viewport and workstation window differ). Thus, LOCATOR input always defines a position in the NDC range \([0,1] \times [0,1]\).

To return to the application program a position in world coordinates, the position in NDC space needs to be transformed from NDC to WC by the inverse of one of the normalization transformations. Each normalization transformation has associated with it a viewport input priority which is only relevant to LOCATOR and STROKE input. Normalization transformations are ordered in a list defined by the viewport input priority. At GKS initialisation, an implementation defined number of normalization transformations are initialised to have window and viewport set to the unit square and their viewport input priorities are set relative to the transformation number with transformation number 0 given the highest priority, transformation number 1 the next highest and so on. Changing the viewport input priority of any normalization transformation is allowed at any time.

The LOCATOR input position in NDC space is compared with the viewports of the normalization transformations, to find the normalization transformation with the viewport which has the highest viewport input priority and contains the LOCATOR position. The LOCATOR position is transformed by the inverse of this normalization transformation to the associated WC. This LOCATOR position is returned to the application program in WC together with the number of the normalization transformation used.

As transformation number 0 is the unity transformation with viewport \([0,1] \times [0,1]\) and cannot be changed, this ensures that LOCATOR input is always within at least one viewport. A data flow chart for LOCATOR input is given in figure 10.

As transformation number 0 is given the highest viewport input priority initially, LOCATOR input is effectively returned in WC equivalent to NDC until a normalization transformation is defined with a viewport input priority greater than that of transformation number 0. If a normalization transformation is no longer required for mapping LOCATOR input back to WC, it can effectively be hidden by reassigning it a viewport input priority lower than transformation number 0.
Figure 10. Data flow chart for locator input
Changing the viewport input priority of transformation number 0 is allowed.

In an event report, generated by a LOCATOR device in EVENT mode, the DC position is transformed to the appropriate WC position before the event report is placed on the input queue. These transformations may be performed while the normalization and workstation transformations are being changed; thus, there is a race condition. The implementation has therefore to treat the transformations as resources to be allocated and deallocated between the competing processes.

Between placing on the input queue and the execution of AWAIT EVENT which removes the LOCATOR position from the queue, it is possible for the normalization transformation and the workstation transformation to be changed by the application program. To ensure that the DC position located is equivalent to the WC position retrieved from the input queue, it is advisable for the application program not to change a transformation while a LOCATOR is in EVENT mode.

4.6.5 Transformation of stroke input

Similar considerations apply to transformation of STROKE input as apply to LOCATOR input, with the complication that more than one point is involved.

When each point of a stroke is generated, the coordinates of the point are transformed from DC to NDC by the inverse workstation transformation then in effect. STROKE input can only be obtained from positions within the part of the current workstation viewport into which the current workstation window is mapped (analogous to LOCATOR input). Thus STROKE input always consists of points in the NDC range $[0,1] \times [0,1]$.

To return to the application program points in world coordinates, the points in NDC space need to be transformed from NDC to WC by the inverse of one of the normalization transformations. The STROKE points in NDC space are compared with the viewports of the normalization transformations, to find the normalization transformation with the viewport which has the highest viewport input priority and contains all of the points. The STROKE points are then transformed by the inverse of this normalization transformation and returned to the application program in WC together with the number of the normalization transformation used.

If the STROKE device is in SAMPLE mode, the normalization transformation used may vary between successive samples.

In EVENT mode, there is a similar race condition to that applying to LOCATOR input. Between placing an event report on the input queue and the execution of AWAIT EVENT which removes the STROKE event from the queue, it is possible for the normalization transformation and the workstation transformation to be changed by the application program. To ensure that the DC points input by the operator are equivalent to the WC points retrieved from the input queue, it is advisable for the application program not to change transformations while a STROKE device is in EVENT mode.
4.7 Segments

4.7.1 Introduction to segments

In GKS, graphical output primitives may be grouped in segments as well as being created outside segments. Each segment is identified by a unique, application specified segment name. Segments may be:

a) transformed;
b) made visible or invisible;
c) highlighted;
d) ordered front to back, which impacts overlapping primitives;
e) made detectable or undetectable;
f) deleted;
g) renamed;
h) inserted into the open segment or into the stream of primitives outside segments (see 4.7.6).

Only primitives contained inside segments are affected by these operations. The application program has no access to primitives outside segments once they have been created.

Every primitive within a segment has an attribute PICK IDENTIFIER which establishes a second level of naming. The sole function of the PICK IDENTIFIER is the identification of primitives; it cannot be used for manipulations. This level of naming is provided in GKS to reduce the segment overhead for applications where a great number of picture parts need to be distinguished for input but the need for manipulation is less important.

Whereas segment names are unique, the same value for PICK IDENTIFIER can be assigned arbitrarily to single output primitives or groups of output primitives within segments, as illustrated in the following sequence of functions:

```
SET PICK IDENTIFIER(4);
CREATE SEGMENT (1);
Output functions;
SET PICK IDENTIFIER (2);
Output functions;
CLOSE SEGMENT;
Output functions;
SET PICK IDENTIFIER (5);
Output functions;
CREATE SEGMENT (2);
Output functions;
SET PICK IDENTIFIER (3);
Output functions;
CLOSE SEGMENT;
{segment=1, PICK IDENTIFIER=4}
{segment=1, PICK IDENTIFIER=2}
{primitives not pickable}
{PICK IDENTIFIER=2}
{primitives not pickable}
{PICK IDENTIFIER=5}
{segment=2, PICK IDENTIFIER=5}
{segment=2,PICK IDENTIFIER=3}
```

After a segment is closed, primitives in it cannot be modified nor can primitives be added to or deleted from the segment. No function is provided to extend a segment after it has been closed. Clipping rectangles and primitive attributes (geometric attributes, attributes controlling non-geometric aspects, and PICK IDENTIFIER) are stored along with primitives in segment storage. Geometrical transformations, changes of the segment attributes and changes of the workstation specific bundle and colour tables referenced from within a segment are possible.
All values describing the state of a segment (i.e., name, segment attributes, and workstations active at creation time) are stored in a segment state list that GKS keeps during a segment's lifetime.

Each segment is stored on all workstations active at the time the segment is created (CREATE SEGMENT). It can be deleted on all workstations by the DELETE SEGMENT function. It can be deleted on a specific workstation by the DELETE SEGMENT FROM WORKSTATION function. All segments stored on a specific workstation can be deleted from it by the CLEAR WORKSTATION function.

Segment storage on an OUTPUT or OUTIN workstation is referred to as Workstation Dependent Segment Storage (WDSS). GKS provides a second segment storage system for workstation independent storage of pictures at run time, Workstation Independent Segment Storage (WISS; see 4.7.5 and 4.7.8). Segments cannot be moved from WDSS to another workstation, but from WISS they can.

Segments have a unique name across all segment storage. A GKS implementation provides a large number of available segment names (for example 32000).

4.7.2 Segment attributes
Segment attributes affect all the primitives in a segment. The segment attributes are:

a) SEGMENT TRANSFORMATION: (see 4.7.3);
b) VISIBILITY: a segment is either displayed or not;
c) HIGHLIGHTING: a visible segment is either highlighted or not;
d) SEGMENT PRIORITY: if parts of segments (for example, FILL AREA, CELL ARRAY) overlap, the segment with the higher priority will be preferred, both when the segments are displayed and when they are picked;
e) DETECTABILITY: a segment can either be selected by a pick input device or it cannot.

The segment attributes are unique for each segment and do not vary on different workstations. The default segment attributes (identity transformation, visible, not highlighted, priority 0, undetectable) are assigned to a segment when it is opened. The segment attributes of any segment in existence, including the open segment, may be changed. The binding of segment attributes is shown in figure 2 (see 4.4).

Segment priority affects segments being displayed (i.e., performing segment and workstation transformations, including clipping, for each primitive of the segment). If parts of primitives overlap with others of a visible segment with higher priority, these parts may be invisible. Whether a workstation supports this feature is indicated in the workstation description table. This feature is intended to address appropriate hardware capabilities only. It is not intended to mandate shielding on non-raster displays. When primitives within a segment overlap, the implementation determines the appearance of the overlapped parts. The actual effect is listed in the documentation accompanying an implementation.

When primitives of segments overlapping each other are picked, the segment with higher priority is selected. When primitives of the same segment or of segments with equal priority overlap, the results are implementation dependent.
4.7.3 Segment transformations

Segment transformations are a mapping from NDC onto NDC. They perform translation, scaling and rotation.

Segment transformations are characterized by:

a) segment name;

b) transformation matrix.

The transformation matrix is a $2 \times 3$ matrix, consisting of a $2 \times 2$ scaling and rotation portion and a $2 \times 1$ translation portion. Utility functions (EVALUATE TRANSFORMATION MATRIX, ACCUMULATE TRANSFORMATION MATRIX) are available to the application program for setting up the transformation matrices. A fixed point for scaling and rotation, and a shift vector in either WC or NDC may be specified. In the former case, the WC values of the fixed point and shift vector are first transformed using the current normalization transformation.

The segment transformation takes place after the normalization transformation, but before any clipping.

A segment transformation, specified by the SET SEGMENT TRANSFORMATION function, is not actually performed in the segment storage but only saved in the segment state list. Every time the segment is redrawn this segment transformation is applied before clipping. Successive SET SEGMENT TRANSFORMATION function calls for the same segment are not accumulated; each succeeding transformation matrix replaces its predecessor. By calling SET SEGMENT TRANSFORMATION with an identity transformation matrix, the original segment can be obtained without loss of information.

Locator input data is not affected by any segment transformation.

4.7.4 Clipping and WDSS

Clipping takes place after the normalization and segment transformations have been applied. Each primitive is clipped against the clipping rectangle associated with the primitive when it was put into the segment. If the 'clipping indicator' entry in the GKS state list was CLIP when this occurred, the clipping rectangle associated with the primitive is the clipping rectangle in the GKS state list at that time, otherwise it is $[0,1] \times [0,1]$ in NDC.

Clipping rectangles are not transformed by the segment transformation and thus clipping is always performed against a rectangle whose edges are parallel to the NDC space coordinate axes.

4.7.5 Workstation Independent Segment Storage

One Workstation Independent Segment Storage (WISS) is defined, where segments can be stored for use by the COPY SEGMENT TO WORKSTATION, ASSOCIATE SEGMENT WITH WORKSTATION, and INSERT SEGMENT functions, as described in 4.7.8. None of these functions modify the contents of the segments to which they are applied. Only one WISS is permitted in a GKS implementation.

The ability to manipulate segments requires the storage of all segments so that they can be reused on whatever workstations are active when they are created. By contrast, primitives outside segments cannot be reused. GKS does not define the manner and format of the storage of segments as long as all segment operations can be performed and as long as the correct clipping is applied to each primitive.

The point in the viewing pipeline at which primitives are recorded in the WISS immediately follows the point at which data are distributed to workstations, as shown in figure 9 (see 4.6). For
this reason the WISS is treated like a workstation (of category WISS), as far as control functions are concerned. Primitives are transformed from world coordinates to NDC before they are distributed to workstations.

Whether the WISS is realized within the GKS nucleus or by utilizing the capabilities of an appropriate physical workstation or other input/output device is left to the implementor.

4.7.6 WISS functions and clipping

Just as in other workstations, a segment is stored in WISS if WISS is active when the segment is created and a clipping rectangle is associated with each primitive. If the ‘clipping indicator’ entry in the GKS state list is CLIP when this occurs, the clipping rectangle associated with the primitive is the clipping rectangle in the GKS state list; otherwise it is [0,1] x[0,1] in NDC.

COPY SEGMENT TO WORKSTATION copies primitives from a segment in WISS to be output on the specified workstation. The function takes a copy of each primitive and its associated clipping rectangle from a segment in the WISS, transforms the primitives by the segment transformation and puts the clipping rectangles and the transformed primitives into the viewing pipeline at the place equivalent to the one where the information left (but it is sent only to the workstation specified in the invocation), as shown in figure 9 (see 4.6). This function cannot be invoked when a segment is open. By contrast with ASSOCIATE SEGMENT WITH WORKSTATION, this function does not cause a segment to exist on the specified workstation.

ASSOCIATE SEGMENT WITH WORKSTATION copies the segment to the WDSS of the specified workstation in the same way as if the workstation were active when the segment was created. Clipping rectangles are copied unchanged. This function cannot be invoked when a segment is open.

INSERT SEGMENT allows previously stored primitives (in segments in WISS) to be transformed and again placed into the stream of output primitives. INSERT SEGMENT reads the primitives from a segment in the WISS, applies the segment transformation followed by the insert transformation and then inserts them into the viewing pipeline at the point before data are distributed to the workstations. All clipping rectangles in the inserted segment are ignored. Each primitive processed is assigned a new clipping rectangle, which is the clipping rectangle in the GKS state list if the ‘clipping indicator’ entry in the GKS state list is CLIP and is [0,1] x[0,1] if the ‘clipping indicator’ entry in the GKS state list is NOCLIP. In other words, inserted primitives are assigned clipping rectangles in the same manner as directly created primitives. Thus, all primitives processed by a single invocation of INSERT SEGMENT receive the same clipping rectangle. Inserted information may re-enter the WISS, if the WISS is active and a segment is open.

An invocation of INSERT SEGMENT has no effect on output primitives passing through the pipeline before or after the invocation. The INSERT SEGMENT function can be used when a segment is open but the open segment cannot, itself, be inserted.
4.8 Graphical Input

4.8.1 Introduction to logical input devices

An application program obtains graphical input from an operator by controlling the activity of one or more logical input devices, which deliver logical input values to the program.

A logical input device is identified by a workstation identifier, an input class and a device number.

The workstation identifier identifies an open workstation, belonging to category INPUT or OUTIN, of which the logical input device is a part. The logical input device is implemented in terms of a physical input device or devices present on the workstation.

The input class determines the type of logical input value that the logical input device delivers. The six input classes and the logical input values they provide are:

a) LOCATOR: a position in world coordinates and a normalization transformation number.

b) STROKE: a sequence of points in world coordinates and a normalization transformation number.

c) VALUATOR: a real number.

d) CHOICE: a CHOICE status and a non-negative integer which represents a selection from a number of choices.

e) PICK: a PICK status, a segment name and a pick identifier. Primitives outside segments cannot be picked.

f) STRING: a character string.

The device number distinguishes different logical input devices of the same class on the same workstation.

A workstation of category INPUT or OUTIN contains at least one logical input device. A GKS implementation providing at least one OUTIN workstation always provides an operator with at least one logical input device in each class defined at the level (see 4.10.3) of the implementation.

Each logical input device can be operated in three modes, called operating modes. At any time a logical input device is in one, and only one, of the modes set by the invocation of a function in the group SET <input class> MODE. The three operating modes are REQUEST, SAMPLE and EVENT. Input from devices is obtained in different ways depending on the mode as follows.
g) REQUEST: A specific invocation of REQUEST <input class> causes an attempt to read a logical input value from a specified logical input device. This can only occur when the logical input device is in REQUEST mode. GKS waits until the input is entered by the operator or a break action is performed by the operator. The break action is dependent on the logical input device and on the implementation. If a break occurs, the logical input value is not valid.

h) SAMPLE: A specific invocation of SAMPLE <input class> causes GKS, without waiting for an operator action, to return the current logical input value of a specified logical input device. This can only occur when the logical input device is in SAMPLE mode.

i) EVENT: GKS maintains one input queue containing temporally ordered event reports. An event report contains the identification of a logical input device and a logical input value from that device. Event reports are generated asynchronously, by operator action only, from input devices in EVENT mode.

The application program can remove the oldest event report from the queue and examine its contents. The application can also flush from the queue all event reports from a specified logical input device.

A specific logical input device is said to be taking part in an interaction during the whole time that it is in SAMPLE or EVENT mode, but, when it is in REQUEST mode, only during the execution of a REQUEST <input class> function for that device. Alternatively, an interaction with the device may be said to be underway during that time. Many devices on many workstations may be taking part in interactions simultaneously.

4.8.2 Logical input device model

To describe the precise actions of the logical input devices, it is first necessary to describe their relationship with physical input devices, using the concept of measures and triggers.

A logical input device contains a measure, a trigger, an initial value, a prompt and echo type, an echo area and a data record containing details about the prompt and echo type. A logical input device's measure and trigger are parts of the implementation of the workstation containing the logical input device. Initial value, prompt and echo type, echo area, and data record can be supplied by the application program.

The measure of a logical input device is a value determined by one or more physical input devices together with a 'measure mapping'. More than one measure can simultaneously be determined by a single physical device; a separate measure mapping applies for each measure. A measure can be seen as the state of an independent, active process (a measure process). Each state corresponds exactly with a logical input value.

The current state of the measure process (i.e. the device's measure) is available to GKS as a logical input value. Whenever the device is taking part in an interaction, the measure process is in existence. Under other conditions, this process does not exist.

When a measure process comes into existence, the data in the workstation state list entry for the logical input device are examined. The initial value is checked for legality according to input class dependent rules explained in 4.8.4. If the check succeeds, the initial value is used
as the current state of the process; otherwise a value dependent on the logical input device is used. Next, a prompt is output to indicate that the device is ready for use. (The prompt technique used by a device is determined by its prompt and echo type, which may be selected by calling the appropriate INITIALISE function.) Creation of the measure process is then complete.

While the measure process is in existence, if echoing is required, output indicating the current state of the measure process is provided to the operator.

The trigger of a logical input device is a physical input device or a set of them together with a 'trigger mapping'. The operator can use a trigger to indicate significant moments in time. At these moments, the trigger is said to 'fire'. A single operator action (for example, pressing a button or a light pen tip switch) causes the firing of not more than one trigger. Several logical input devices can refer to the same trigger.

A trigger can be seen as an independent, active process (a trigger process) that sends a message to one or more recipients when it fires. A logical input device is a recipient of its trigger if there is a pending REQUEST for it or if it is in EVENT mode. Both of these conditions can be true simultaneously for different logical input devices. If there is at least one recipient for a trigger, the trigger process is in existence. Under other conditions this process does not exist.

If a REQUEST for a logical input device is pending when the device's trigger fires, the measure of that device is used to satisfy the REQUEST. If one or more devices containing a given trigger are in EVENT mode when the trigger fires, the identifications of those devices and their measure values are passed to the input queue mechanism as separate event reports. The input queue mechanism is described in detail in 4.8.5.

When a trigger firing succeeds in satisfying a REQUEST, or adding event records to the input queue, GKS provides to the operator an acknowledgement the form of which depends on the implementation of the logical input device. The acknowledgement is not controllable by a GKS function.

4.8.8 Operating modes of logical input devices

The mode of a logical input device may be changed by invoking the appropriate SET <input class> MODE function.

After an invocation of SET <input class> MODE with the parameter REQUEST, no measure process exists for the specified device and the device's identifier is not on its trigger's list of recipients. After an invocation with the parameter EVENT, a newly initiated measure process is in existence for the specified device and the device's identifier is on its trigger's list of recipients.
REQUEST mode

Single value returned to application program on trigger firing. Interaction lasts for single request.

measure ----> trigger

SAMPLE mode

Trigger inoperative. Value returned for each call to SAMPLE. Multiple calls to SAMPLE in a single interaction.

SAMPLE --- measure ----> trigger

EVENT mode

Value and device identification sent to single queue on trigger firing and removed by a call to AWAIT EVENT.

EVENT --- QUEUE --- measure ----> trigger

Note: thick arrows represent flow of input data
thin arrows represent control

Figure 11. The relationship between the measure and trigger for different operating modes, illustrated for a single logical input device
After an invocation with the parameter SAMPLE, a newly initiated measure process is in existence for the specified device, but the device's identifier is not on its trigger's list of recipients.

Initially a logical input device is in REQUEST mode.

While a device is in REQUEST mode, a logical input value may be obtained by invoking the appropriate REQUEST <device class> function. The effects of doing so are as follows.

a) To create a measure process for the specified device and to set its value to the initial value from the workstation state list as described in 4.8.4. Echoing is performed by the measure process if echoing is on for the specified device.

b) To add the device’s identifier to its trigger’s list of recipients. If the list was previously empty, the trigger process is started.

c) To suspend GKS until the trigger of the specified device fires, or the operator invokes the break facility.

d) If the trigger fired, to set the logical input value to the current state of the measure process.

e) To destroy the measure process.

f) To remove the device's identifier from its trigger's list of recipients. If this list becomes empty, the trigger process is destroyed.

g) If the trigger fired, to return the logical input value and the status OK, otherwise to return the status NONE.

While a logical input device is in SAMPLE mode, a logical input value may be obtained by invoking the appropriate SAMPLE <input class> function. The effect of doing so is to set the logical input value to the current state of the measure process without waiting for a trigger firing.

While a logical input device is in EVENT mode, logical input values are added as event reports to the input queue, and may be obtained in sequence by invoking AWAIT EVENT, and then invoking the appropriate GET <device class> function. (More details of the input queue are given in 4.8.5.)

Figure 11 shows the effect of every operating mode on the measure and trigger of a logical input device.

4.8.4 Measures of each input class

Details of the measures of logical input devices of different classes are as follows.

A LOCATOR measure consists of a position in world coordinates and a normalization transformation number. Let P and N denote these values.

Then P transformed to NDC by N lies within the workstation window. Also P lies within the window specified by N and, in addition, P transformed to NDC by N lies outside all viewports of higher priority than N.

A STROKE measure consists of a sequence of points in world coordinates and a normalization transformation number. Let $P_1, \ldots, P_m$ be the points and N be the transformation number.

Then $P_i, (1 \leq i \leq m)$ transformed to NDC by N lie within the workstation window. Also $P_i, (1 \leq i \leq m)$ lie within the window specified by N and, in addition, there is no viewport of higher priority than N containing all the points $P_i$ transformed to NDC by N. Thus, N may change as points are added to the stroke.
Any invocation of SET WINDOW, SET VIEWPORT or SET VIEWPORT INPUT PRIORITY can cause a change in \( P \) (any \( P \), for STROKE) or \( N \) or both, but the above conditions hold for the new values.

The rules imply that no normalization transformation having priority less than that of transformation 0 can appear in the state of a LOCATOR or STROKE measure process (with the default settings of the viewport input priorities, normalization transformation 0 has the highest).

A VALUATOR measure provides logical input values that are real numbers. Each value lies between (possibly including) minimum and maximum values, which are in the data record in the workstation state list.

A CHOICE measure provides logical input values whose components are OK or NOCHOICE and an integer in the range 1 to a device dependent maximum specified in the workstation description table. If the first component is OK, then the integer is valid. CHOICE input typically occurs when an operator presses a button (the numeric identifier of the button determines the measure) or combination of buttons (the measure is derived from the combination of buttons pressed).

A PICK measure provides logical input values whose components are OK or NOPICK, segment name and a pick identifier. If the first component is OK, then the segment name and pick identifier obey the following rules:

- a) The segment exists and has VISIBILITY on and DETECTABILITY on.
- b) The segment is present on the workstation containing the PICK device.
- c) The pick identifier is the pick identifier attribute of at least one output primitive in the segment. This is tested using the clipping parameters in effect when the primitive arrived at the workstation. Part of the primitive lies within the workstation window and, if clipping was on, part also lies within the primitive's (normalization) clipping rectangle. Further, the primitive is not completely overlapped by primitives in a segment of higher priority.

The PICK initial value is tested against the above rules whenever the PICK measure process is initiated. If the rules are not satisfied, the process state is set to NOPICK.

For certain workstations, testing rule c) when the PICK measure process is initiated, may be very expensive. In such cases, only rules a) and b) need be tested.

The PICK measure is defined using the properties of output primitives and segments. PICK devices exist only on workstations of category OUTIN.

A STRING measure provides logical input values which are character strings up to a device dependent maximum length specified by the buffer size value in the data record in the workstation state list.

4.8.5 Input queue and current event report

The input queue contains zero or more event reports. Event reports contain pairs of values (device identifier, logical input value) resulting from trigger firings. Event reports can be added to the input queue when logical input devices in EVENT mode are triggered by the operator. Events can be removed from the input queue by invocations of AWAIT EVENT, FLUSH DEVICE EVENTS and CLOSE WORKSTATION.

When a trigger that is part of one or more logical input devices in EVENT mode fires, the resulting event reports are entered into the queue and marked as a group of simultaneous event reports. An event report for each device is added to the input queue, if and only if there is room for the whole group of simultaneous event reports.
The order of reports within a group of simultaneous event reports is undefined.

If there is not room in the queue for all event reports when a trigger fires, input queue overflow has occurred. Input queue overflow is not reported to the application program immediately. It is reported via the error mechanism during the next invocation of any GKS function that can remove event reports from the input queue (AWAIT EVENT, FLUSH DEVICE EVENTS, and CLOSE WORKSTATION). The input queue has to be emptied before further event reports will be added. Between the detection of input queue overflow and the next time AWAIT EVENT is invoked with the input queue empty, no events are generated by trigger firings and thus no acknowledgements are provided. (This permits the application program to determine how many events were in the queue when overflow occurred by calling AWAIT EVENT with zero timeout.)

When the 'input queue overflow' error is reported, the trigger causing the overflow is indicated by placing into the error state list the identification of any one of the logical input devices using that trigger which was in EVENT mode at the time the overflow was detected.

AWAIT EVENT, if the queue is not empty, removes the first event report after copying the logical input value into the current event report in the GKS state list. The workstation identifier, input class and device number are returned to the application program directly by AWAIT EVENT. If the queue is empty, AWAIT EVENT suspends execution until an event report is queued or until the specified timeout period has elapsed.

The application program may obtain the contents of the current event report by calling the appropriate GET <input class> function.

If, after removing the event report there remain in the queue other reports in the same group of simultaneous events as the removed report, the 'more simultaneous events' entry in the GKS state list is set to MORE. Otherwise it is set to NOMORE.

FLUSH DEVICE EVENTS removes all event reports for a specific device from the input queue. CLOSE WORKSTATION removes from the input queue all event reports for all logical input devices on that workstation.

If the 'more simultaneous events' entry has the value MORE, when either FLUSH DEVICE EVENTS or CLOSE WORKSTATION is invoked, and they remove all the remaining reports in the group of simultaneous event reports at the head of the queue, then the entry is set to NOMORE.

4.8.6 Initialisation of input devices

For each input class, there is an INITIALISE function which can only be called if the logical input device it specifies is in REQUEST mode. These functions provide the following information to a device via the workstation state list (if the INITIALISE function is not called, then default values apply):

a) An initial value appropriate to the class. If the initial value violates the rules, an error occurs and the workstation state list is unchanged.

b) A prompt and echo type that selects the prompting technique and, if echoing is on, the echoing technique for a logical input device. An implementation dependent prompt and echo type (type 1) is required for all logical input devices. Further prompt and echo types appropriate to each class are defined but not required. These further types are listed with the appropriate INITIALISE function. Prompt and echo types above those are reserved for registration or future standardization. Prompt and echo types less than 0 are device dependent.
c) An echo area \((x_{\text{min}}, x_{\text{max}}, y_{\text{min}}, y_{\text{max}})\) in device coordinates. Input device implementations may use the echo area for certain prompt and echo types to display prompts or echoes.

d) A data record. Some input classes have mandatory control values in the data record. Some prompt and echo types within an input class also have mandatory control values in the data record. These values occupy well defined places in the data record. In any data record used in initialising an input device, values mandatory to the input class, if any, appear first followed by values mandatory to the prompt and echo type if any. Depending on the device and prompt and echo type, the data record may contain other (additional) information.

When a logical input device is REQUESTed, or when it is set to EVENT or SAMPLE mode, its measure is set to the initial value from the workstation state list, unless this is not a valid measure for the device. If it is not a valid measure for the device, the measure is set to a device dependent value, except for PICK devices, for which the measure is set to NOPICK.

Prompt and echo types describe both the prompt, which informs the operator that the device is available, and the echo, which informs the operator of the state of the measure. The functions provided to control input device mode, \(\text{SET <input class>} \text{MODE}\), also control whether echo is on or off. In addition, an implementation dependent acknowledgement of successful trigger firings is provided.

The items in data records mandatory for each class are: in a STROKE data record, input buffer size in number of points; in a VALUATOR data record, low value and high value; in a STRING data record, input buffer size and initial cursor position. Prompt and echo types which have mandatory values are types 2, 3, 4 and 5 for CHOICE.
4.9 GKS Metafile Interface

For the purpose of long-term filing of graphical information, GKS provides an interface to sequential files called GKS Metafiles (GKSMs). They can be used for:

a) transporting graphical information between systems;

b) transporting graphical information from one place to another (for example, by means of magnetic tapes);

c) transporting graphical information from one GKS application to another;

d) storing accompanying non-graphical information.

The GKSMs behave like workstations. For output and input, several different workstations of the categories MO and MI can be used concurrently. However, some workstation control and inquiry functions are not applicable to these workstations (they are not meaningful). Section A.5 of appendix A gives a complete listing of all GKS functions which affect workstations of category MO and MI.

The application program may write data in a metafile using WRITE ITEM TO GKSM. After closing an MO workstation the metafile may be opened as an MI workstation.

Three functions, GET ITEM TYPE FROM GKSM, READ ITEM FROM GKSM and INTERPRET ITEM are provided to read and interpret metafiles. These functions assume that a metafile consists of a sequence of items. Each item comprises an item type, an item data record length and an item data record. The item type indicates either that the item contains information that can be interpreted by GKS or that it contains information that was written by an application program (using WRITE ITEM TO GKSM). When an MI workstation is opened, the first item in the metafile becomes the 'current item'.

GET ITEM FROM GKSM delivers the item type and item data record length of the current item.

READ ITEM FROM GKSM copies the contents of the item data record of the current item into a data area supplied by the application program and then makes the next item in the metafile the current item.

INTERPRET ITEM takes the contents of a GKSM item data record supplied by the application program (as delivered by READ ITEM FROM GKSM) and causes appropriate changes in the set of GKS state variables and generates appropriate graphical output as determined by the metafile specification. The information in metafile item data records can be regarded as falling into classes corresponding to the classes of GKS functions. The interpretation of primitive attribute, clipping rectangle or clipping indicator information causes appropriate changes to entries in the GKS state list. The geometric primitive attribute information, which is expressed in NDC, is transformed by the inverse of the currently selected normalization transformation before being used to set the appropriate entries in the GKS state list. Information corresponding to GKS functions which control a single workstation, but where a workstation is not specified, may be interpreted on all active workstations.
4.10 GKS levels

4.10.1 Introduction

The GKS system is designed to be usable by a wide range of applications, from static plotting to dynamic motion and real time interaction. In addition, many display devices lack features (such as picking) that would require considerable implementation effort to simulate with software. It is therefore desirable to permit GKS implementations that do not include all of the functional capabilities defined in this standard.

4.10.2 The level structure

The functional capabilities of GKS can be grouped into the major areas:

a) output (minimal performance, full performance);

b) input (no input, REQUEST input, full input);

c) number of workstations (one workstation, multiple workstations);

d) attributes (only predefined bundles and individual attribute specifications possible, full bundle concept);

e) segmentation (none, basic segmentation (without Workstation Independent Segment Storage), full segmentation).

If an arbitrary combination of capabilities were to be considered a valid GKS implementation, an almost unlimited number of different standard dialects would result and program portability, one of the major goals of this standard, would not be achieved. Therefore, twelve valid levels of the GKS system are defined, in order to address the most common classes of equipment and applications. Each GKS implementation provides at least the functions of one level.

The level structure has two independent axes: input and "all the other functions", summarized as output.

The output level axis has the four possibilities:

m: Minimal output;

0: All primitives and attributes;

1: Basic segmentation with full output;

2: Workstation Independent Segment Storage.

The input level axis has the three possibilities:

a: No input;

b: REQUEST input;

c: Full input.

In GKS, capabilities are expressed by functions and by ranges of parameters.

There are three different types of capability at each level:

f) An explicitly defined and required capability. Every GKS implementation at a specific level supports the capability at that level.

g) An explicitly defined and non-required capability. A GKS implementation may support the capability and, if it does, it is implemented according to the explicit function definitions.

h) A conceptually defined and non-required capability. A GKS implementation may provide the capability. Its implementation follows general rules given by the GKS concepts and functional definitions.
The set of explicitly defined and required capabilities includes:

i) predefined bundle numbers up to the required minimum; (output levels 0,1, and 2);
ii) linetypes 1 to 4;
iii) marker types 1 to 5;
iv) text precision STROKE (output levels 1 and 2);
vi) interior style HOLLOW;
vii) one input device for each input class defined at that level (input levels b and c);
viii) prompt and echo type 1 (input levels b and c).

The set of explicitly defined and non-required capabilities includes:

p) text precision STROKE (output levels m and 0);
q) interior style SOLID, PATTERN, HATCH;
r) transformable patterns;
s) segment priority (output levels 1 and 2);
t) prompt and echo types above 1 that are defined (input levels b and c).

The set of conceptually defined and non-required capabilities includes:

u) linetypes above 4;
v) marker types above 5;
w) specific generalized drawing primitives;
x) prompt and echo types above the defined set (input levels b and c);
y) specific escape functions.

Explicitly defined and non-required capabilities of a specific level can become explicitly defined and required capabilities in a higher level of GKS, through variations in the ranges of parameters, for example text precision STROKE and metafile workstations. Each GKS level contains precisely those functions that are explicitly defined and required in that level. However, ranges of parameters may contain additional explicitly defined and non-required capabilities and conceptually defined and non-required capabilities.

4.10.3 Level functionality

The facilities making up each of the components of a level are as follows:

Output level m: Minimal output

a) subset of control available;
b) subset of primitives available;
c) subset of attributes controllable;
d) there are no bundles for attributes;
e) colour representation modification possible;
f) only one workstation with output capabilities available at a time;
g) normalization transformation 0 and only one settable normalization transformation available;
h) subset of inquiries available.
Output level 0: All primitives and attributes
  a) all output level m capabilities;
  b) basic control;
  c) all primitives available at least in minimal performance;
  d) use of predefined bundles only (no modification to bundles);
  e) only one workstation without output capabilities required at a time;
  f) metafile workstations not required; if provided, then both input and output are available;
     if not provided, then metafile functions return appropriate errors;
  g) multiple normalization transformations (but a system with normalization 0 and only one
     settable normalization transformation is allowable);
  h) suitable basic inquiries;
  i) pixel readback provided (non-pixel devices may report non-processing).

Output level 1: Basic segmentation with full output
  a) all output level 0 capabilities;
  b) full workstation control;
  c) full output features;
  d) full bundle concept;
  e) multiple workstation concept;
  f) metafile workstation required;
  g) multiple settable normalization transformations;
  h) basic segmentation (no Workstation Independent Segment Storage);
  i) suitable inquiries.

Output level 2: Workstation Independent Segment Storage
  a) all output level 1 capabilities;
  b) Workstation Independent Segment Storage.

Input level a: No input
  a) no facilities.

Input level b: REQUEST input
  a) input device initialization and mode setting functions;
  b) REQUEST functions on all appropriate devices;
  c) appropriate logical input devices include PICK if and only if combined with output level
     1 capabilities;
  d) function to set viewport input priority.

Input level c: Full input
  a) all input level b capabilities;
  b) SAMPLE and EVENT mode input.

Table 1 gives a summary of the functionality of each valid GKS level. Each box contains only
those functions added to the previous boxes of the same row and column.
### Table 1 - GKS level concept

<table>
<thead>
<tr>
<th>Output Level</th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>m</td>
<td>No input, minimal control, only individually set attributes, 1 settable normalization transformation, and subset of output functions and attributes.</td>
<td>REQUEST input, mode setting and initialise functions for logical input devices, no PICK.</td>
<td>SAMPLE and EVENT input, no PICK.</td>
</tr>
<tr>
<td>0</td>
<td>Basic control, predefined bundles, multiple normalization transformation facilities but minimum settable required is 1, and all output functions, metafile workstations optional.</td>
<td>Set viewport input priority</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Full output including full bundle concept, multiple workstation concept, basic segmentation (everything except Workstation Independent Segment Storage), metafile workstations required.</td>
<td>REQUEST PICK, mode setting and initialise for PICK.</td>
<td>SAMPLE and EVENT input for PICK.</td>
</tr>
<tr>
<td>2</td>
<td>Workstation Independent Segment Storage</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Embedded in the levels summarized above are variations in the number of possibilities required in the set of explicitly defined and required capabilities. Table 2 exactly identifies the minimum support which is always provided at each level.
<table>
<thead>
<tr>
<th>CAPABILITY</th>
<th>Level</th>
<th>ma</th>
<th>mb</th>
<th>mc</th>
<th>0a</th>
<th>0b</th>
<th>0c</th>
<th>1a</th>
<th>1b</th>
<th>1c</th>
<th>2a</th>
<th>2b</th>
<th>2c</th>
</tr>
</thead>
<tbody>
<tr>
<td>Foreground Colours (intensity)</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Linetypes</td>
<td></td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Linewidths</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Predefined polyline bundles</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Settable polyline bundles</td>
<td></td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Marker types</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Marker sizes</td>
<td></td>
<td>-</td>
<td>-</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Predefined polymarker bundles</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Settable polymarker bundles</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Character expansion factors (see note a)</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>String precision fonts</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Character precision fonts</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Stroke precision fonts</td>
<td></td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Predefined text bundles</td>
<td></td>
<td>-</td>
<td>-</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Settable text bundles</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Predefined patterns (see note b)</td>
<td></td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Settable patterns (see notes b and c)</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Hatch styles (see note c)</td>
<td></td>
<td>-</td>
<td>-</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Predefined fill area bundles</td>
<td></td>
<td>-</td>
<td>-</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Settable fill area bundles</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Segment priorities (see note d)</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Input classes</td>
<td></td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Prompt and echo types per device</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Length of input queue (see note e)</td>
<td></td>
<td>-</td>
<td>20</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>20</td>
<td>-</td>
<td>-</td>
<td>20</td>
<td>-</td>
<td>20</td>
</tr>
<tr>
<td>Maximum string buffer size (characters)</td>
<td></td>
<td>-</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
<td>72</td>
</tr>
<tr>
<td>Maximum stroke buffer size (points)</td>
<td></td>
<td>-</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>Workstations of category OUTPUT or OUTIN</td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Workstations of category INPUT or OUTIN</td>
<td></td>
<td>-</td>
<td>1</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Workstation Independent Segment Storage</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MO workstations</td>
<td></td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>MI workstations</td>
<td></td>
<td>-</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

0 indicates explicitly defined and non-required at that level
- indicates not defined at that level

Notes:

a) relevant only for character and string precision text
b) relevant only for workstation supporting pattern interior style
c) relevant only for workstation supporting hatch interior style
d) relevant only for workstation supporting segment priorities
e) since available resources are finite and entries have variable size, it may not always be possible to achieve the minimal values in a particular application

Table 2 - Minimal Support Required at Each Level
4.11 States of GKS and inquiry functions

4.11.1 Description of states

GKS exists in one of five different operating states (see figure 12):

- GKCL = GKS closed;
- GKOP = GKS open;
- WSOP = At least one workstation open;
- WSAC = At least one workstation active;
- SGOP = Segment open.

![Diagram of GKS states and transitions]

Figure 12. Some transitions between operating states
The operating state value is contained in a global static variable that is initialised, before the first invocation of GKS, to the value GKCL. The operating states differ in so far as individual calls to GKS are allowed only in certain operating states, as indicated in the functional description in section 5.

The overall state of GKS is defined by a set of state variables having specific values. These state variables are characterized by the fact that they allow a complete description of the effects of the functions. The total set of GKS state variables contains the following subsets:

a) operating state;

b) GKS state list;

c) segment state list for every existing segment;

d) input queue;

e) workstation state list for every open workstation;

f) GKS error state list.

Certain functions cause these state subsets to be allocated, made available and cancelled. When these state subsets are allocated, they are initialised with default values. When initialising a workstation state list, some of the default values are taken from the workstation description table for that workstation type. There is a workstation description table for each workstation type supported by the GKS implementation. The variables of the state subsets are modified and inquired by invocations of GKS functions.

When an error condition is detected during execution of a GKS function, GKS calls the ERROR HANDLING procedure. During execution of the ERROR HANDLING procedure, GKS is in an error state. In this error state, GKS allows only inquiry functions, the ERROR LOGGING procedure and the EMERGENCY CLOSE GKS procedure to be executed and no modifications to any of the state lists except the error state list.

### 4.11.2 Inquiry functions

Inquiry functions return values directly from or derived from the various state lists and workstation description tables. The data types of the values and the default values of the entries are summarized in section 6.

The inquiry functions of GKS are designed in such a way that they do not cause any errors to be generated. Inquiry functions for values that may be logically unavailable have an output parameter, 'error indicator', that determines whether or not the other returned values are valid. The availability parameter is of type integer and, in the event of the other values not being available, returns an error number, which identifies the appropriate GKS error condition. The same error numbers are used as for non-inquiry functions and thus the standard list of error messages should be consulted. If GKS is not in the proper state, then the error number appropriate to this condition is the one returned, even if there are other reasons for the values being unavailable. If the values are available, zero is returned in the error indicator parameter.

For all values except zero the returned output values are implementation dependent. The description of each inquiry function lists the error indicator values that the function can return.

Some inquiry functions that retrieve values from the workstation state lists have an input parameter of type 'enumeration' that can take the following values:

a) SET: the values returned are those provided by the application program;

b) REALIZED: the values returned are those used by the workstation when the actual values are mapped to the available values in the workstation.
4.12 Error handling

For level \( m \), only the default actions of error handling are supported; that is, errors are printed on the error file specified in OPEN GKS. For other levels for each GKS function, a finite number of error situations is specified, any of which will cause the ERROR HANDLING procedure to be called. Every GKS implementation supports this error checking. The ERROR HANDLING procedure provides an interface between GKS and the application program. The ERROR HANDLING procedure, if provided by the application program, may interpret the information about the error and may store data in a data area for subsequent interpretation by the application program after return from the GKS function that caused the error.

The GKS error handling strategy is derived from the following classification of errors:

I errors resulting in a precisely defined reaction;
II errors resulting in an attempt to save the results of previous operations;
III errors which cause unpredictable results including the loss of information.

GKS recognizes three situations in which errors are detected:

A error detected in GKS procedures;
B error detected in procedures called from GKS (driver procedures, operating system procedures);
C error detected in other areas of the application program.

If errors are detected outside GKS (situation C), either the application program may regain control over the execution or program execution will be terminated abnormally. In the latter case, results are unpredictable (class III), and in the worst situation, all graphical information produced so far in this job may be lost. If, however, the application program obtains control, it may attempt to close GKS properly or at least attempt an emergency closure by calling the EMERGENCY CLOSE GKS procedure. Similarly, if the error occurs in procedures called by GKS and control is not returned properly to GKS, the effects are unpredictable.

The EMERGENCY CLOSE GKS procedure is an implementation dependent facility. Its purpose is to save as much of the graphical information produced as possible. The effects of this procedure on the workstations are left undefined in this standard. The EMERGENCY CLOSE GKS procedure may be called directly from the application program. It is also called from GKS itself as a standard error reaction to class II errors.

Finally, all errors that are listed explicitly as part of the definition of GKS functions belong to class I. Either they are detected within GKS itself (situation A) or a procedure called from GKS has returned control, to the corresponding GKS procedure, with the appropriate error information (situation B). In all these class I cases, GKS calls the ERROR HANDLING procedure. If a GKS function is called with more than one error condition applicable, at least one error is reported.

The application program may either provide its own ERROR HANDLING procedure or may use that provided as part of GKS. Any ERROR HANDLING procedure accepts the following information from GKS:

a) the identification of the error condition;

b) the identification of the GKS function that called the ERROR HANDLING procedure;

c) the error file.

The ERROR HANDLING procedure provided by GKS just calls the ERROR LOGGING procedure, using the same set of parameters. The latter performs the following actions:
d) prints an error message and GKS function identification on the error file;
e) returns to the calling procedure.

This two-stage calling of the error procedures allows the application program to supply its own ERROR HANDLING procedure, while still having access to services provided by the ERROR LOGGING procedure, as shown in the following example of an application program supplied ERROR HANDLING procedure.

Example

PROCEDURE ERROR HANDLING (error number, identification of GKS function, error file);
Interpret GKS function and error identification
in order to select the following cases:
CASE 'special treatment':
Interpret error parameters as passed from GKS;
Store information for application program in application supplied data area;
Return to calling GKS procedure;
CASE 'standard treatment':
Call ERROR LOGGING procedure with all the above parameters;
Return to calling GKS procedure;
END.

All GKS procedures perform the following actions after detecting an error condition:
f) set error state to ON;
g) call ERROR HANDLING procedure with appropriate parameters;
h) set error state to OFF;
i) Perform built-in error reaction (normally, a function causing an error has no effect; to accomplish this in some cases requires clean-up operations).

All GKS procedures check on entry (in the following order):
j) that GKS is in the correct state;
k) that the values of input parameters are valid.

At least the first error detected is reported except that, in the case of inquiry functions, the first error detected is returned via the error indicator.

The application program supplied ERROR HANDLING procedure has access to the set of GKS state variables. However, no modification of GKS state is possible during error handling, i.e. only GKS inquiry functions, the ERROR LOGGING procedure and the EMERGENCY CLOSE GKS procedure may be called by the application program supplied ERROR HANDLING procedure. This is achieved by setting the error state to ON prior to calling the ERROR HANDLING procedure from GKS and setting the error state to OFF afterwards. An inquiry function cannot generate an error.

Error numbers for errors originating in GKS functions can be found in section 5 under each GKS function.

4.12.1 Reserved errors
Unused error numbers less than 2000 are reserved for future standardization.

Error numbers 2000-3999 are reserved for language bindings.

Error numbers greater than or equal to 4000 are reserved for registration or future
standardization.

NOTE Error numbers are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When an error has been approved by the ISO Working Group on Computer Graphics, the error number will be assigned by the Registration Authority.

4.13 Special Interfaces between GKS and the Application Program

A uniform escape mechanism for allowing access to installation and hardware specific features (a 'standard way of being non-standard') is provided by means of the ESCAPE function. Although the use of this mechanism reduces the portability of the application program, it does so in an easily identifiable manner.

The ESCAPE function does not generate geometrical output; by contrast, the GENERALIZED DRAWING PRIMITIVE can generate geometrical output not otherwise generated by GKS.
5 GKS Functions

5.1 Notational Conventions
The heading of each function specifies:
a) the function’s name;
b) the GKS states in which the function may be used except that, for inquiry functions (see 5.9), only those states in which the inquiry function can return valid values is specified;
c) the GKS lowest level L at which the function is explicitly defined and required.

More information about levels and states can be found in 4.10 and 4.11 respectively. Appendices A.3 and A.4 contain lists of all functions according to levels and states respectively. The GKS functional capabilities are summarized in Appendix F.

The parameter lists indicate for each entry:
a) whether the entry is an input (In) or output (Out) parameter;
b) the name of the parameter;
c) for coordinate data, the coordinate system (WC, NDC, DC) used in the function call (coordinate systems are explained in 4.6);
d) either, for enumeration type data, the permitted values, or, for real and integer data, any restriction on their value range (for example, ‘>0’); the notation is explained in 6.1;
e) the data type, which is either of simple form (I, R, S, P, N, E) representing:

<table>
<thead>
<tr>
<th>I</th>
<th>integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>real</td>
</tr>
<tr>
<td>S</td>
<td>string</td>
</tr>
<tr>
<td>P</td>
<td>point</td>
</tr>
<tr>
<td>N</td>
<td>name</td>
</tr>
<tr>
<td>E</td>
<td>enumeration type</td>
</tr>
</tbody>
</table>

or is a compound based on one or more of the simple forms (for example: n×P) or is a compound, the content and structure of which are defined for some uses:

D    data record

The data types are explained in 6.1.
5.2 Control Functions

OPEN GKS

Parameters:
- In error file F
- In amount of memory units for buffer area I

Effect: GKS is set into the operating state GKOP = "GKS open". The GKS state list is allocated and initialised as indicated in 6.4. The GKS description table and the workstation description tables are made available.

The entry 'error file' in the GKS error state list is set to the value specified by the first parameter. The permitted buffer area which can be used by GKS for internal purposes is limited.

NOTE: Certain environments might not permit dynamic memory management. In this case, the buffer area may be limited in a static way to be described in the installation documentation.

References:
- 4.11
- 4.12

Errors:
1. GKS not in proper state: GKS shall be in the state GKCL
2. Specified error file is invalid

CLOSE GKS

Parameters: none

Effect: GKS is set into the operating state GKCL = "GKS closed". The GKS description table, GKS state list and the workstation description tables become unavailable. All GKS buffers are released and all GKS files are closed.

NOTE: GKS can be reopened by invoking the function OPEN GKS.

References:
- 4.11
- 4.12

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

OPEN WORKSTATION

Parameters:
- In workstation identifier N
- In connection identifier C
- In workstation type W

Effect: If GKS is in operating state GKOP, it is set into the state WSOP = 'at least one workstation open'. GKS requests the operating system to establish the specified connection for a workstation characterized in the workstation description table by the 'workstation type'. The workstation state list is allocated and initialised as indicated in 6.5. The
workstation identifier is added to the set of open workstations in the GKS state list.
OPEN WORKSTATION ensures that the display surface is clear, but does not clear the
surface needlessly.

NOTE: The connection identifier is given in a form suitable for the application program language.

An attempt to open a workstation with the same connection identifier and workstation type as one already
open causes error 26 to occur.

References:
4.5.2
4.11

Errors:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
21 Specified connection identifier is invalid
22 Specified workstation type is invalid
29 Specified workstation type does not exist
24 Specified workstation is open
26 Specified workstation cannot be opened
28 Workstation Independent Segment Storage is already open
42 Maximum number of simultaneously open workstations would be exceeded

CLOSE WORKSTATION

Parameters:

In workstation identifier

Effect: An implicit UPDATE WORKSTATION (with the parameter update regeneration flag
set to PERFORM) is performed for the specified workstation. The workstation state
list is deallocated. The workstation identifier is deleted from the set of open worksta¬
tions in the GKS state list and from the set of associated workstations in the segment
state list of every segment containing it. If the set of associated workstations of a seg¬
ment becomes empty, the segment is deleted. The input queue is flushed of all events
from all devices on the workstation being closed. If the ‘identification of one of the
logical input devices that caused an input queue overflow’ entry in the GKS error state
list refers to this workstation identifier, then all the contents of that entry become
undefined.

The connection to the workstation is released. GKS is set into operating state GKOP if
no workstations remain open. The display surface need not be cleared when CLOSE
WORKSTATION is invoked, but it may be cleared.

References:
4.5.2
4.8.5
4.11

Errors:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
29 Specified workstation is active
147 Input queue has overflowed
**ACTIVATE WORKSTATION**

Parameters:

- **In** workstation identifier

Effect: GKS is set into the operating state WSAC = "At least one workstation active". The specified workstation is marked active in the workstation state list. The workstation identifier is added to the set of active workstations in the GKS state list.

NOTE: Output primitives are sent to and segments are stored on all active workstations.

References:

- 4.5.2
- 4.11

Errors:

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

**DEACTIVATE WORKSTATION**

Parameters:

- **In** workstation identifier

Effect: The specified workstation is marked inactive in the workstation state list. The workstation identifier is deleted from the set of active workstations in the GKS state list. GKS is set into the operating state WSOP = "At least one workstation open" if no workstation remains active.

NOTE: While a workstation is inactive, primitives are not sent to it nor does it store new segments. Segments already stored on this workstation are retained.

References:

- 4.5.2
- 4.11

Errors:

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

**CLEAR WORKSTATION**

Parameters:

- **In** workstation identifier
- **In** control flag

Effect: The specified workstation is cleared. If the control flag is set to (CONDITIONALLY, ALWAYS), the specified workstation is cleared conditionally, otherwise it is cleared always.

References:

- 4.5.2
- 4.11
Effect: All of the following actions are executed in the given sequence:

a) All deferred actions for the specified workstation are executed (without intermediate clearing of the display surface).

b) The display surface is set to a clear state according to the control flag as follows:
   CONDITIONALLY:
   the display surface is cleared only if the 'display surface empty' entry in the workstation state list is NOTEMPTY.
   ALWAYS:
   the display surface is cleared.

c) If the 'workstation transformation update state' entry in the workstation state list is PENDING, the 'current workstation window' and 'current workstation viewport' entries in the workstation state list are assigned the values of the 'requested workstation window' and 'requested workstation viewport' entries; the 'workstation transformation update state' entry is set to NOTPENDING.

d) For all segments stored on the specified workstation, the workstation identifier is deleted from the 'set of associated workstations' in the segment state list. If the 'set of associated workstations' of a segment becomes 'empty', the segment is deleted. The 'set of stored segments for this workstation' in the workstation state list is set to 'empty'.

e) The 'new frame action necessary at update' entry in the workstation state list is set to NO.

f) The 'display surface empty' entry in the workstation state list is set to EMPTY.

References:

4.5.3  
4.5.4  
4.5.5  
4.7.1

Errors:

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

REDRAW ALL SEGMENTS ON WORKSTATION  
Parameters:  
In workstation identifier N  

Effect: All of the following actions are executed in the given sequence:

a) All deferred actions for the specified workstation are executed (without intermediate clearing of the display surface).

b) The display surface is cleared only if the 'display surface empty' entry in the workstation state list is NOTEMPTY. The entry is set to EMPTY.

c) If the 'workstation transformation update state' entry in the workstation state list is PENDING, the 'current workstation window' and 'current workstation viewport' entries in the workstation state list are assigned the values of the 'requested workstation window' and 'requested workstation viewport' entries; the 'workstation transformation update state' entry is set to NOTPENDING.
d) All visible segments stored for this workstation (i.e. contained in the 'set of stored segments for this workstation' in the workstation state list) are redisplayed. This action typically causes the 'display surface empty' entry in the workstation state list to be set to NOTEMPTY.

e) The 'new frame action necessary at update' entry in the workstation state list is set to NO.

References:
4.5.3
4.5.4
4.5.5
4.7

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
33 Specified workstation is of category MI
35 Specified workstation is of category INPUT
36 Specified workstation is Workstation Independent Segment Storage

UPDATE WORKSTATION

Parameters:
In  workstation identifier N
In  update regeneration flag (PERFORM,POSTPONE) E

Effect: All deferred actions for the specified workstation are executed (without intermediate clearing of the display surface). If the update regeneration flag is set to PERFORM and the 'new frame action necessary at update' entry in the workstation state list is YES, then the following actions are executed in the given sequence:

a) The display surface is cleared only if the 'display surface empty' entry in the workstation state list is NOTEMPTY. The entry is set to EMPTY.

b) If the 'workstation transformation update state' entry in the workstation state list is PENDING, the 'current workstation window' and 'current workstation viewport' entries in the workstation state list are assigned the values of the 'requested workstation window' and 'requested workstation viewport' entries; the 'workstation transformation update state' entry is set to NOTPENDING.

c) All visible segments stored on this workstation (i.e. contained in the 'set of stored segments for this workstation' in the workstation state list) are redisplayed. This action typically causes the 'display surface empty' entry in the workstation state list to be set to NOTEMPTY.

d) The 'new frame action necessary at update' entry in the workstation state list is set to NO.

NOTE: If the update regeneration flag is PERFORM, UPDATE WORKSTATION suspends the effect of SET DEFERRAL STATE. In that case, it is equivalent to the following sequence of functions.

INQUIRE WORKSTATION STATE;
save deferral state;
SET DEFERRAL STATE (ASAP,ALLOWED);
set deferral state to saved value;

If the value of the 'new frame action necessary at update' entry is NO or the update regeneration flag is
POSTPONE, UPDATE WORKSTATION merely initiates the transmission of blocked data. If the value of the entry 'new frame action necessary at update' is YES and the regeneration flag is PERFORM, UPDATE WORKSTATION behaves as REDRAW ALL SEGMENTS ON WORKSTATION.

The 'new frame action necessary at update' entry in a workstation state list is set to YES during deferred action generation if both of the following are true (see 4.5):

a) an action causing modification of the picture is actually deferred on that workstation;
b) the workstation display surface does not allow modification of the image without redrawing the whole picture (for example, plotter, storage tube display).

References:
4.5.3
4.5.4
4.5.5

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
33 Specified workstation is of category MI
35 Specified workstation is of category INPUT
36 Specified workstation is Workstation Independent Segment Storage

SET DEFERRAL STATE

Parameters:

- In workstation identifier (N)
- In deferral mode (ASAP, BNIG, BNIL, ASTI) (E)
- In implicit regeneration mode (SUPPRESSED, ALLOWED) (E)

Effect: The entries ‘deferral mode’ and ‘implicit regeneration mode’ for the specified workstation are set in the workstation state list. Depending on the new value of ‘deferral mode’, deferred output may be unblocked. If in the workstation state list, the new value of ‘implicit regeneration mode’ is ALLOWED and ‘new frame action necessary at update’ is YES, then an action equivalent to REDRAW ALL SEGMENTS is performed.

References:
4.5.3
4.5.5

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
33 Specified workstation is of category MI
35 Specified workstation is of category INPUT
36 Specified workstation is Workstation Independent Segment Storage
Control Functions

MESSAGE
Parameters:
In workstation identifier N
In message S
Effect: The message function:
   a) may display a message at an implementation dependent location on the workstation
      viewport or on some separate device associated with the workstation.
   b) does not alter the GKS state list.
   c) may affect the workstation in a purely local way (for example, requesting the opera¬
      tor to change paper). Possible effects on the execution of the application program or
      on subsequent commands sent to the workstation by GKS are stated explicitly in the
      implementation dependencies manual.

References:
4.5.6

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
86 Specified workstation is Workstation Independent Segment Storage

ESCAPE
Parameters:
In specific escape function identification N
In escape input data record D
Out escape output data record D
Effect: The specified non-standard specific escape function is invoked. The form of the escape
data record and which of them are used may vary for different functions. Also the
GKS states allowing the invocation of a specific escape function may be restricted. The
following rules govern the definition of a new specific escape function:
a) the GKS design concept (see Section 0 - Introduction)
b) the GKS state lists are not altered;
c) the function does not generate geometrical output;
d) any side effects are well documented.
Specific escape functions may apply to more than one workstation, for example all open
workstations or all active workstations. The escape input data record can include a
workstation identifier where this is required.

NOTE Examples of specific escape functions anticipated at present are:
a) support of raster devices allowing the display of more than one frame buffer;
b) use of rasterop hardware to manipulate data previously output by cell array.

Where the specific escape function identification is bound to an integer in a programming language,
specific escape function identifications greater than 0 are reserved for registration or future standardiza¬
tion and specific escape function identifications less than 0 are implementation dependent.
Specific escape function identifications are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a specific escape function has been approved by the ISO Working Group on Computer Graphics, the specific escape function identification will be assigned by the Registration Authority.

References:

4.13

Errors:

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

5.3 Output Functions

**POLYLINE**

Parameters:

- **In number of points**
- **In points**

Effect: A sequence of connected straight lines is generated, starting from the first point and ending at the last point. The current values of the polyline attributes, as given by the GKS state list (see 6.4), are bound to the primitive. The polyline attributes are listed in 4.4.2.

If, after the workstation transformation, all points coincide, no error is generated and whether anything is drawn is workstation dependent.

References:

- 4.4.1
- 4.4.2
- 4.4.3
- 4.5.3

Errors:

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

**POLYMARKER**

Parameters:

- **In number of points**
- **In points**

Effect: A sequence of markers is generated to identify all the given positions. The current values of the polymarker attributes, as given by the GKS state list (see 6.4), are bound to the primitive. The polymarker attributes are listed in 4.4.2.

NOTE: A marker is visible if and only if the marker position is within the clipping rectangle. The clipping of partially visible markers is workstation dependent.

References:

- 4.4.1
- 4.4.2
- 4.4.4
- 4.5.3

Errors:

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

Parameters:

In text position
In character string

Effect: A character string is generated. The current values of the text attributes, as given by the GKS state list (see 6.4), are bound to the primitive. The text attributes are listed in 4.4.2.

The text position is given in WC and transformed by the current normalization transformation.

If, after the workstation transformation, the height or width of a character is zero, no error is generated and whether anything is drawn is workstation dependent.

If the character string contains a control character (for example, characters outside the range 2/0 to 7/14 inclusive in ISO 646), the effect is workstation dependent. Either error 101 is generated or some visual effect may be generated or the character may be ignored. Even if error 101 occurs, the character string is displayed on all active workstations which do not generate error 101.

NOTE Text is clipped in a way that depends on the text precision as defined by the text font and precision currently selected (either via the text bundle or individually, depending upon the corresponding ASF).

References:

4.4.1
4.4.2
4.4.5
4.5.3

Errors:

5 GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
101 Invalid code in string

FILL AREA

Parameters:

In number of points
In points WC

Effect: A FILL AREA primitive is generated. The current values of the fill area attributes, as given by the GKS state list (see 6.4), are bound to the primitive. The fill area attributes are listed in 4.4.2.

The polygon defined by the points is filled according to the fill area interior style currently selected (either via the fill area bundle or individually, depending upon the corresponding ASF). The boundary is drawn for interior style HOLLOW, whereas, for other interior styles, the amount of the boundary that is drawn ensures that two regions which share a common edge appear without an apparent gap or overlap at that edge, to the extent that can reasonably be achieved.

If parts of the area are clipped, the resulting new boundaries become part of the area boundaries. Multiple subareas may be generated (see figure 13).
The interior of a polygon is defined in the following way (see figure 14).

For a given point, create a straight line starting at that point and going to infinity. If the number of intersections between the straight line and the polygon is odd, the point is within the polygon; otherwise it is outside. If the straight line passes a polygon vertex tangentially, the intersection count is not affected. If a point is within the polygon, it is included in the area to be filled subject to the rule for boundaries.

If, after the workstation transformation, all points coincide, no error is generated and whether anything is drawn is workstation dependent. If, after the workstation transformation, some or all lines in a bounding polygon have a line segment in common, no error is generated. Whether the resulting line segment is regarded as part of the boundary to be drawn or not is workstation dependent.
For PICK input, a FILL AREA primitive displayed with interior style HOLLOW may be picked by pointing at any point on the bounding polygon. A FILL AREA primitive displayed with interior style SOLID or PATTERN may be picked by pointing at any point inside the polygon. Pointing at a hole in the area does not identify that area. A FILL AREA primitive displayed with interior style HATCH may be picked by pointing at any point on any hatch line.

Errors:

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

CELL ARRAY

Parameters:

\[
\begin{array}{l}
\text{In} \quad \text{cell rectangle } (P,Q) \\
\text{In} \quad \text{dimensions of colour index array } DX,DY \\
\text{In} \quad \text{colour index array}
\end{array}
\]

Effect: A CELL ARRAY primitive is generated using the cell rectangle corners, the dimensions of the colour index array and the colour index array.

A rectangle, which is taken to be aligned with the world coordinate axes, is defined by the points P and Q. This rectangle is conceptually divided into a grid in DXxDY cells. Each cell has a width of \(|PX-QX|/DX\) and a height of \(|PY-QY|/DY\), where \((PX,PY)\) are the coordinates of the cornerpoint P and \((QX,QY)\) are the coordinates of the cornerpoint Q. The colour index array is oriented with respect to the rectangle by associating the four corners as follows: the \((1,1)\) element is associated with the cell having P at one corner; the \((DX,DY)\) element with the cell having Q at one corner; the \((1,DY)\) element with the cell having the point \((PX,QY)\) at one corner; the \((DX,1)\) element with the cell having the point \((QX,PY)\) at one corner. The colour of each cell is specified by the index of the corresponding element of the colour index array. If an index is not present in the colour table on a workstation, a workstation dependent index is used on that workstation.

The rectangular grid defined by P,Q,DX and DY is subject to all transformations, potentially transforming the rectangular cells into parallelograms. If part of a transformed cell is outside the window, the transformed cell is partially clipped. Mapping the transformed cells onto the pixels of a raster display (see figure 15) is performed by the following rules:

a) If the centrepoint of a pixel lies inside the parallelogram defined by the transformed rectangle, its colour is set.

b) The pixel is assigned the colour of the cell which contains the pixel's centrepoint. Thus, the pixel colour is selected by point sampling the transformed rectangle at the pixel centrepoint, not by area sampling or filtering.

If, after the workstation transformation, the four corner points are coincident or collinear, no error is generated and whether anything is drawn is workstation dependent.

References:

4.4.1
4.4.2
4.4.6
4.5.3
The minimal simulation required is to draw the transformed boundaries of the cell rectangle, using implementation dependent colour, linewidth and linetype.

References:
4.4.1
4.4.2
4.4.7
4.5.3

Errors:
5 GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
91 Dimensions of colour array are invalid

GENERALIZED DRAWING PRIMITIVE (GDP)

Parameters:
- In number of points
- In points
- In GDP identifier
- In GDP data record

Effect: A Generalized Drawing Primitive (GDP) of the type indicated by the GDP identifier is generated on the basis of the given points and the GDP data record. The current values of the entries in the GKS state list (see 6.4) for the sets of polyline, polymarker, text or fill area attributes are bound to the primitive. These attributes are listed in 4.4.2. When the GDP generates output at the workstation, zero or more of the sets of attributes are used. These are the sets of attributes most appropriate for the specified GDP.
function and are selected for the GDP as part of the definition of the GDP. (They are defined in the workstation description table.)

NOTE. The parameters are transmitted to the workstation and interpreted in a workstation dependent way. In this way special capabilities of the workstation can be addressed. Even if error 104 or error 105 occurs, the GDP is displayed on all active workstations capable of doing so. For example, some of the primitives anticipated at present are:

a) circle: points given are centre, peripheral point;

b) circular arc: points given are centre, start point, end point to be connected anticlockwise in world coordinates;

c) ellipse: points given are 2 focal points, peripheral point;

d) elliptic arc: points given are 2 focal points, start point, end point to be connected anticlockwise in world coordinates;

e) interpolating curve (for example, spline): points given are interpolated.

The recommended set of attributes to use for the above GDP examples would be the polyline attributes.

It should be emphasized that the points, specified as parameters, are transformed by GKS after the interpretation of the points (as defining, say, a spline curve or circle) is performed by the active workstations. For example, a GDP, which defines a circle, would appear as an ellipse when the transformation has differential scaling for the two axes. Each specific GDP definition defines how the transformation is applied to both the points and the shape of the GDP. Though the points cannot be clipped, the resulting output of the GDP is clipped against the clipping rectangle, if the 'clipping indicator' entry in the GKS state list is CLIP, and the workstation window. If a specific GDP is available on a workstation but is unable to be generated because the current transformations or clipping rectangle are such that the preceding conditions would be violated, error 105 occurs.

The GDP data record attribute list may contain additional data for each point (for example, vertex order for splines) which remain untransformed. These have to be defined for a specific GDP. In defining a new GDP, the GKS design concepts (see section 8) are not violated. The set of generalized drawing primitives implemented on a workstation may be empty.

Where the GDP identifier is bound to an integer in a programming language, GDP identifiers greater than 0 are reserved for registration or future standardization and GDP identifiers less than 0 are implementation dependent.

GDP identifiers are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a GDP has been approved by the ISO Working Group on Computer Graphics, the GDP identifier will be assigned by the Registration Authority.

References:

4.4.1
4.4.2
4.4.8
4.5.3
4.13

Errors:

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

5.4.1 Workstation Independent Primitive Attributes

SET POLYLINE INDEX

Parameters:
In polyline index (1..n) 1

Effect: The 'current polyline index' entry in the GKS state list is set to the value specified by the parameter. This value is used when creating subsequent POLYLINE output primitives.

References:
4.4.2
4.4.3

Errors:
8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
60 Polyline index is invalid

SET LINETYPE

Parameters:
In linetype (-n..-1,1..n) 1

Effect: The 'current linetype' entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent POLYLINE output primitives, created when the 'current linetype ASF' entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent POLYLINE output primitives, created when the 'current linetype ASF' entry in the GKS state list is BUNDLED.

Linetype values produce linetypes as indicated:

- <0 implementation dependent
- 1 solid line
- 2 dashed line
- 3 dotted line
- 4 dashed-dotted line
- ≥5 reserved for registration or future standardization

If the specified linetype is not available on a workstation, linetype 1 is used on that workstation.

NOTE Linetype values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a linetype has been approved by the ISO Working Group on Computer Graphics 1), the linetype value will be assigned by the Registration Authority.

References:
4.4.2
4.4.3

Errors:
8 \textit{GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP}
68 Linetype is equal to zero

\textbf{SET LINEWIDTH SCALE FACTOR}

\textit{GKOP, WSOP, WSAC, SGOP \quad L0a}

Parameters:
\begin{itemize}
\item \textbf{In} \text{linewidth scale factor} \geq 0 \quad R
\end{itemize}

Effect: The ‘current linewidth scale factor’ entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent POLYLINE output primitives, created when the ‘current linewidth scale factor’ entry in the GKS state list is \textit{INDIVIDUAL}. This value does not affect the display of subsequent POLYLINE output primitives, created when the ‘current linewidth scale factor’ entry in the GKS state list is \textit{BUNDLED}.

The linewidth scale factor is applied to the nominal linewidth on a workstation; the result is mapped by the workstation to the nearest available linewidth.

References:
4.4.2
4.4.3

Errors:
8 \textit{GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP}
65 Linewidth scale factor is less than zero

\textbf{SET POLYLINE COLOUR INDEX}

\textit{GKOP, WSOP, WSAC, SGOP \quad L1ma}

Parameters:
\begin{itemize}
\item \textbf{In} \text{polyline colour index} \quad (0..n) \quad I
\end{itemize}

Effect: The ‘current polyline colour index’ entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent POLYLINE output primitives, created when the ‘current polyline colour index’ entry in the GKS state list is \textit{INDIVIDUAL}. This value does not affect the display of subsequent POLYLINE output primitives, created when the ‘current polyline colour index’ entry in the GKS state list is \textit{BUNDLED}.

The colour index is a pointer into the colour tables of the workstations. If the specified colour index is not present in a workstation colour table, a workstation dependent colour index is used on that workstation.

References:
4.4.2
4.4.3

Errors:
8 \textit{GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP}
92 Colour index is less than zero
GKS Functions

**SET POLYMARKER INDEX**

Parameters:

- **In** polymer marker index $(1..n)$ 1

Effect: The ‘current polymer marker index’ entry in the GKS state list is set to the value specified by the parameter. This value is used when creating subsequent POLYMARKER output primitives.

References:

4.4.2
4.4.4

Errors:

- 8 **GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP**
- 66 **Polymer marker index is invalid**

**SET MARKER TYPE**

Parameters:

- **In** marker type $(-n..-1,1..n)$ 1

Effect: The ‘current marker type’ entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent POLYMARKER output primitives, created when the ‘current marker type ASF’ entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent POLYMARKER output primitives, created when the ‘current marker type ASF’ entry in the GKS state list is BUNDLED.

Marker type values produce centred symbols as indicated:

<table>
<thead>
<tr>
<th>Value</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;0</td>
<td>implementation dependent</td>
</tr>
<tr>
<td>1</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>+</td>
</tr>
<tr>
<td>3</td>
<td>*</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>X</td>
</tr>
<tr>
<td>≥6</td>
<td>reserved for registration or future standardization</td>
</tr>
</tbody>
</table>

Marker type 1 is always displayed as the smallest displayable dot. If the specified marker type is not available on a workstation, marker type 3 (*) is used on that workstation.

**NOTE** Marker type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a marker type has been approved by the ISO Working Group on Computer Graphics, the marker type value will be assigned by the Registration Authority.

References:

4.4.2
4.4.4

Errors:

- 8 **GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP**
- 69 **Marker type is equal to zero**

---

Output Attributes

GKS Functions

**SET MARKER SIZE SCALE FACTOR**

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>marker size scale factor</th>
</tr>
</thead>
</table>

Effect: The 'current marker size scale factor' entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent POLYMARKER output primitives, created when the 'current marker size scale factor ASF' entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent POLYMARKER output primitives, created when the 'current marker size scale factor ASF' entry in the GKS state list is BUNDLED.

The marker size scale factor is applied to the nominal marker size on a workstation; the result is mapped by the workstation to the nearest available marker size.

References:

4.4.2

4.4.4

Errors:

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

71 Marker size scale factor is less than zero

**SET POLYMARKER COLOUR INDEX**

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>polymarker colour index</th>
</tr>
</thead>
</table>

Effect: The 'current polymarker colour index' entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent POLYMARKER output primitives, created when the 'current polymarker colour index ASF' entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent POLYMARKER output primitives, created when the 'current polymarker colour index ASF' entry in the GKS state list is BUNDLED.

The colour index is a pointer into the colour tables of the workstations. If the specified colour index is not present in a workstation colour table, a workstation dependent colour index is used on that workstation.

References:

4.4.2

4.4.4

Errors:

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

92 Colour index is less than zero

**SET TEXT INDEX**

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>text index</th>
</tr>
</thead>
</table>

Effect: The 'current text index' entry in the GKS state list is set to the value specified by the parameter. This value is used when creating subsequent TEXT output primitives.
GKS Functions

References:

4.4.2
4.4.5

Errors:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
72 Text index is invalid

SET TEXT FONT AND PRECISION

Parameters:

In text font and precision (-n..-1,1..n;STRING,CHAR,STROKE) (I;E)

Effect: The ‘current text font and precision’ entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent TEXT output primitives, created when the ‘current text font and precision ASF’ entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent TEXT output primitives, created when the ‘current text font and precision ASF’ entry in the GKS state list is BUNDLED.

Text font and precision is a single text aspect; a particular text font can be available at some, but not necessarily all, precisions. Text font 1 contains a graphical representation of the characters defined in ISO 646 (see 4.4.5). Text fonts greater than 1 are reserved for registration or future standardization. Text fonts less than 0 are implementation dependent. The text precision value determines the fidelity with which the other text aspects are used. The values of text precision, in order of increasing fidelity, are STRING, CHAR and STROKE (see 4.4.5).

If the specified text font and precision is not available on a workstation, the value (1;STRING) is used on that workstation.

NOTE: Text font numbers are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a text font has been approved by the ISO Working Group on Computer Graphics, the text font number will be assigned by the Registration Authority.

References:

4.4.2
4.4.5

Errors:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
75 Text font is equal to zero

SET CHARACTER EXPANSION FACTOR

Parameters:

In character expansion factor

Output Attributes

Effect: The 'current character expansion factor' entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent TEXT output primitives, created when the 'current character expansion factor ASF' entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent TEXT output primitives, created when the 'current character expansion factor ASF' entry in the GKS state list is BUNDLED.

References:

4.4.2
4.4.5

Errors:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
77 Character expansion factor is less than or equal to zero

SET CHARACTER SPACING

Parameters:

In character spacing R

Effect. The 'current character spacing' entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent TEXT output primitives, created when the 'current character spacing ASF' entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent TEXT output primitives, created when the 'current character spacing ASF' entry in the GKS state list is BUNDLED.

References:

4.4.2
4.4.5

Errors:

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

SET TEXT COLOUR INDEX

Parameters:

In text colour index (0..n) I

Effect: The 'current text colour index' entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent TEXT output primitives, created when the 'current text colour index ASF' entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent TEXT output primitives, created when the 'current text colour index ASF' entry in the GKS state list is BUNDLED.

The colour index is a pointer into the colour tables of the workstations. If the specified colour index is not present in a workstation colour table, a workstation dependent colour index is used on that workstation.
GKS Functions

References:

4.4.2
4.4.5

Errors:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
92 Colour index is less than zero

SET CHARACTER HEIGHT

Parameters:

In character height WC >0 R

Effect: The 'current character height' entry in the GKS state list is set to the value specified by
the parameter. The 'current character width' entry in the GKS state list is also set to
the value specified by the parameter. These values are used when creating subsequent
TEXT output primitives.

References:

4.4.2
4.4.5

Errors:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
78 Character height is less than or equal to zero

SET CHARACTER UP VECTOR

Parameters:

In character up vector WC 2xR

Effect: The 'current character up vector' entry in the GKS state list is set to the value specified
by the parameter. The 'current character base vector' entry in the GKS state list is set
to a vector, of arbitrary length, at right angles in the clockwise direction to the value
specified by the parameter. These values are used when creating subsequent TEXT
output primitives.

References:

4.4.2
4.4.5

Errors:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
79 Length of character up vector is zero

SET TEXT PATH

Parameters:

In text path (RIGHT,LEFT,UP,DOWN) E
Output Attributes

Effect: The 'current text path' entry in the GKS state list is set to the value specified by the parameter. This value is used when creating subsequent TEXT output primitives.

NOTE: A change in the value of 'current text path' may make the value of the 'current text alignment' entry inappropriate.

References:
4.4.2
4.4.5

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

SET TEXT ALIGNMENT

Parameters:
In text alignment
(NORMAL, LEFT, CENTRE, RIGHT; NORMAL, TOP, CAP, HALF, BASE, BOTTOM) 2

Effect: The 'current text alignment' entry in the GKS state list is set to the value specified by the parameter. This value is used when creating subsequent TEXT output primitives. Text alignment has two components: horizontal and vertical.

References:
4.4.2
4.4.5

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

SET FILL AREA INDEX

Parameters:
In fill area index (1..n) 1

Effect: The 'current fill area index' entry in the GKS state list is set to the value specified by the parameter. This value is used when creating subsequent FILL AREA output primitives.

References:
4.4.2
4.4.6

Errors:
8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
80 Fill area index is invalid

SET FILL AREA INTERIOR STYLE

Parameters:
In fill area interior style (HOLLOW, SOLID, PATTERN, HATCH) E

References:
GKS Functions

Output Attributes

Effect: The 'current fill area interior style' entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent FILL AREA output primitives, created when the 'current fill area interior style ASF' entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent FILL AREA output primitives, created when the 'current fill area interior style ASF' entry in the GKS state list is BUNDLED.

The fill area interior style is used to determine in what style the area is filled and the possible values are: HOLLOW, SOLID, PATTERN and HATCH (see 4.4.6).

If the requested interior style is not available on a workstation, HOLLOW is used on that workstation.

References:

4.4.2
4.4.6

Errors:

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

SET FILL AREA STYLE INDEX

Parameters:

In  fill area style index (-n...l,n)  I

Effect: The 'current fill area style index' entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent FILL AREA output primitives, created when the 'current fill area style index ASF' entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent FILL AREA output primitives, created when the 'current fill area style index ASF' entry in the GKS state list is BUNDLED.

For interior styles HOLLOW and SOLID, the style index value is unused. For interior style PATTERN, the style index value is greater than 0 and is a pointer into the pattern tables of the workstations. For interior style HATCH, the style index value is non-zero and determines which of a number of hatch styles is used: hatch styles greater than 0 are reserved for registration or future standardization; hatch styles less than 0 are workstation dependent.

If the requested style index is not available on a particular workstation, style index 1 is used on that workstation. If style index 1 is not present on that workstation, the result is workstation dependent.

NOTE Hatch style values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a hatch style has been approved by the ISO Working Group on Computer Graphics, the hatch style value will be assigned by the Registration Authority.

References:

4.4.2
4.4.6

Errors:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
84 Style (pattern or hatch) index is equal to zero

SET FILL AREA COLOUR INDEX

Parameters:
In fill area colour index \((0..n)\)

Effect: The ‘current fill area colour index’ entry in the GKS state list is set to the value specified by the parameter. This value is used for the display of subsequent FILL AREA output primitives, created when the ‘current fill area colour index ASF’ entry in the GKS state list is INDIVIDUAL. This value does not affect the display of subsequent FILL AREA output primitives, created when the ‘current fill area colour index ASF’ entry in the GKS state list is BUNDLED.

The colour index is a pointer into the colour tables of the workstations. If the specified colour index is not present in a workstation colour table, a workstation dependent colour index is used on that workstation.

References:
4.4.2
4.4.6

Errors:
8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
92 Colour index is less than zero

SET PATTERN SIZE

Parameters:
In pattern size \((WC, SX, SY > 0)\)

Effect: The ‘current pattern width vector’ entry in the GKS state list is set to the vector \((SX, 0)\). The ‘current pattern height vector’ entry in the GKS state list is set to the vector \((0, SY)\). When the currently selected (either via the fill area bundle or individually, depending on the corresponding ASF) fill area interior style is PATTERN, these values are used, where possible, in conjunction with the ‘current pattern reference point’ entry in the GKS state list for displaying the FILL AREA output primitives.

References:
4.4.2
4.4.6

Errors:
8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
87 Pattern size value is not positive

SET PATTERN REFERENCE POINT

Parameters:
In reference point \((WC, P)\)

Effect: The ‘current pattern reference point’ entry in the GKS state list is set to the value specified by the parameter. When the currently selected (either via the fill area bundle or individually, depending upon the corresponding ASF) fill area interior style is PATTERN, this value is used, where possible, in conjunction with the ‘current pattern width vector’ and ‘current pattern height vector’ entries in the GKS state list for displaying the FILL AREA output primitives.
GKS Functions

References:

4.4.2
4.4.6

Errors:

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

---

SET ASPECT SOURCE FLAGS

Parameters:

In list of Aspect Source Flags (BUNDLED, INDIVIDUAL) 13xE

Effect: The Aspect Source Flags (ASFs) in the GKS state list are set to the values indicated by the parameter. The elements of the list of ASFs are arranged in the following order:

- linetype ASF
- linewidth scale factor ASF
- polyline colour index ASF
- marker type ASF
- marker size scale factor ASF
- polymarker colour index ASF
- text font and precision ASF
- character expansion factor ASF
- character spacing ASF
- text colour index ASF
- fill area interior style ASF
- fill area style index ASF
- fill area colour index ASF

References:

4.4.2

Errors:

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

---

SET PICK IDENTIFIER

Parameters:

In pick identifier N

Effect: The 'current pick identifier' entry in the GKS state list is set to the value specified by the parameter.

References:

4.4.2
4.7.1
4.8.1
4.8.4
Errors:
8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
97 Pick identifier is invalid

5.4.2 Workstation Attributes (Representations)

SET POLYLINE REPRESENTATION

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>N</td>
</tr>
<tr>
<td>polyline index</td>
<td>(1..n)</td>
</tr>
<tr>
<td>linetype</td>
<td>(-n..1,1..n)</td>
</tr>
<tr>
<td>linewidth scale factor</td>
<td>≥0</td>
</tr>
<tr>
<td>polyline colour index</td>
<td>(0..n)</td>
</tr>
</tbody>
</table>

Effect: In the polyline bundle table of the workstation state list, the given polyline index is associated with the specified parameters.

Linetype:
linetype values produce linetypes as indicated:

- <0 implementation dependent
- 1 solid line
- 2 dashed line
- 3 dotted line
- 4 dashed-dotted line
- ≥5 reserved for registration or future standardization

Linewidth scale factor:
a scale factor applied to the nominal linewidth. The result is mapped by the workstation to the nearest available linewidth.

Polyline colour index:
pointer into the colour table of the workstation.

The polyline bundle table in the workstation state list has predefined entries taken from the workstation description table; a number (see Table 2, in 4.10) are predefined for every workstation of category OUTPUT and OUTIN. Any table entry (including the predefined entries) may be redefined with this function.

When polylines are displayed, the polyline index refers to an entry in the polyline bundle table. If polylines are displayed with a polyline index that is not present in the polyline bundle table, polyline index 1 is used. Which of the aspects in the entry are used depends upon the setting of the corresponding ASFs (see 4.4.2).

NOTE Linetype values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a linetype has been approved by the ISO Working Group on Computer Graphics ¹, the linetype value will be assigned by the Registration Authority.

References:
4.4.3
4.5.3

¹ Currently, the Working Group on Computer Graphics is known as ISO/TC97/SC5/WG2.
Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
33 Specified workstation is of category MI
35 Specified workstation is of category INPUT
36 Specified workstation is Workstation Independent Segment Storage
60 Polyline index is invalid
63 Linetype is equal to zero
64 Specified linetype is not supported on this workstation
65 Linewidth scale factor is less than zero
93 Colour index is invalid

SET POLYMARKER REPRESENTATION

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>workstation identifier</td>
<td>Any</td>
<td>N</td>
</tr>
<tr>
<td>polymarker index</td>
<td>polymarker index</td>
<td>(1..n)</td>
<td>I</td>
</tr>
<tr>
<td>marker type</td>
<td>marker type</td>
<td>(-n...-1,1..n)</td>
<td>I</td>
</tr>
<tr>
<td>marker size scale factor</td>
<td>marker size scale factor</td>
<td>≥0</td>
<td>R</td>
</tr>
<tr>
<td>polymarker colour index</td>
<td>polymarker colour index</td>
<td>(0..n)</td>
<td>I</td>
</tr>
</tbody>
</table>

Effect: In the polymarker bundle table of the workstation state list, the given polymarker index is associated with the specified parameters.

Markertype: marker type values produce centred symbols as indicated:

- <0 implementation dependent
- 1
- 2 +
- 3 *
- 4 0
- 5 X
- ≥6 reserved for registration or future standardization

Marker type 1 is always displayed as the smallest displayable dot.

Marker scale factor: a scale factor applied to the nominal marker size. The result is mapped by the workstation to the nearest available marker size.

Polymarker colour index: a pointer into the colour table of the workstation.

The polymarker bundle table in the workstation state list has predefined entries taken from the workstation description table; a number (see Table 2, in 4.10) are predefined for every workstation of category OUTPUT and OUTIN. Any table entry (including the predefined entries) may be redefined with this function.

When polymarkers are displayed, the polymarker index refers to an entry in the polymarker bundle table. If polymarkers are displayed with a polymarker index that is not present in the polymarker bundle table, polymarker index 1 is used. Which of the aspects in the entry are used depends upon the setting of the corresponding ASFs (see 4.4.2).
NOTE: Marker type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a marker type has been approved by the ISO Working Group on Computer Graphics, the marker type value will be assigned by the Registration Authority.

References:

4.4.4
4.5.3

Errors:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
33 Specified workstation is of category MI
35 Specified workstation is of category INPUT
36 Specified workstation is Workstation Independent Segment Storage
66 Polymarker index is invalid
69 Marker type is equal to zero
70 Specified marker type is not supported on this workstation
71 Marker size scale factor is less than zero
93 Colour index is invalid

SET TEXT REPRESENTATION

Parameters:

In workstation identifier
In text index (1..n) I
In text font and precision (-n..-1,1..n;STRING,CHAR,STROKE) (I;E)
In character expansion factor >0 R
In character spacing R
In text colour index (0..n) I

Effect: In the text bundle table of the workstation state list, the given text index is associated with the specified parameters.

Text font and precision:
a single text aspect; a particular text font can be available at some, but not necessarily all, precisions. The text font value is used to select a particular font on this workstation. Text font 1 contains a graphical representation of the characters defined in ISO 646 (see 4.4.5). Text fonts greater than 1 are reserved for registration or future standardization. Text fonts less than 0 are implementation dependent. The text precision value determines the fidelity with which the other text aspects are used. The values of text precision, in order of increasing fidelity, are STRING, CHAR and STROKE (see 4.4.5).

Character expansion factor:
specifies the deviation of the width to height ratio of the characters from the ratio indicated by the font designer.

Character spacing:
specifies how much additional space is to be inserted between two adjacent character bodies. Character spacing is specified as a fraction of the font-nominal character height.

---

\[1\) Currently, the Working Group on Computer Graphics is known as ISO/TC97/SC5/WG2

Page 102
Text colour index:

a pointer into the colour table of the workstation.

The text bundle table in the workstation state list has predefined entries taken from the workstation description table; a number (see Table 2 in 4.10) are predefined for every workstation of category OUTPUT and OUTIN. Any table entry (including the predefined entries) may be redefined with this function.

When text is displayed, the text index refers to an entry in the text bundle table. If text is displayed with a text index that is not present in the text bundle table, text index 1 is used. Which of the aspects in the entry are used depends upon the setting of the corresponding ASFs (see 4.4.2).

NOTE. Text font numbers are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a text font has been approved by the ISO Working Group on Computer Graphics 1), the text font number will be assigned by the Registration Authority.

References:

4.4.5
4.5.3

Errors:

7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20  Specified workstation identifier is invalid
25  Specified workstation is not open
38  Specified workstation is of category MI
35  Specified workstation is of category INPUT
86  Specified workstation is Workstation Independent Segment Storage
72  Text index is invalid
75  Text font is equal to zero
76  Requested text font is not supported for the specified precision on this workstation
77  Character expansion factor is less than or equal to zero
93  Colour index is invalid

SET FILL AREA REPRESENTATION

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>In workstation identifier</td>
<td>N</td>
</tr>
<tr>
<td>In fill area index</td>
<td>(1..n)</td>
</tr>
<tr>
<td>In fill area interior style</td>
<td>(HOLLOW, SOLID, PATTERN, HATCH)</td>
</tr>
<tr>
<td>In fill area style index</td>
<td>(-n..-1, 1..n)</td>
</tr>
<tr>
<td>In fill area colour index</td>
<td>(0..n)</td>
</tr>
</tbody>
</table>

Effect: In the fill area bundle table of the workstation state list, the given fill area index is associated with the specified parameters.

Fill area interior style:

is used to determine in what style the area is filled and the possible values are: HOLLOW, SOLID, PATTERN and HATCH (see 4.4.6).

Fill area style index:

For interior styles HOLLOW and SOLID, this value is unused. For interior style PATTERN, this value is greater than 0 and is a pointer into the pattern table of the workstation. For interior style HATCH, this value is non-zero and determines which of a number of hatch styles is used: hatch styles greater than 0 are reserved

1) Currently, the Working Group on Computer Graphics is known as ISO/TC97/SC5/WG2
for registration or future standardization; hatch styles less than 0 are workstation dependent.

Fill area colour index:
pointer into the colour table of the workstation.

The fill area bundle table in the workstation state list has predefined entries taken from the workstation description table; a number (see Table 2 in 4.10) are predefined for every workstation of category OUTPUT and OUTIN. Any table entry (including the predefined entries) may be redefined with this function.

When fill area is displayed, the current fill area index refers to an entry in the fill area bundle table. If fill areas are displayed with a fill area index that is not present in the fill area bundle table, fill area index 1 is used. Which of the aspects in the entry are used depends upon the setting of the corresponding ASFs (see 4.4.2).

NOTE Hatch style values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a hatch style has been approved by the ISO Working Group on Computer Graphics 1), the hatch style value will be assigned by the Registration Authority.

References:

4.4.6
4.5.3

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
83 Specified workstation is of category MI
85 Specified workstation is of category INPUT
86 Specified workstation is Workstation Independent Segment Storage
80 Fill area index is invalid
88 Specified fill area interior style is not supported on this workstation
85 Specified pattern index is invalid
86 Specified hatch style is not supported on this workstation
93 Colour index is invalid

SET PATTERN REPRESENTATION

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>In workstation identifier</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>In pattern index</td>
<td>(1..n)</td>
<td>1</td>
</tr>
<tr>
<td>In dimensions of pattern array DX,DY</td>
<td>(1..n)</td>
<td>2x1</td>
</tr>
<tr>
<td>In pattern array</td>
<td>(0..n)</td>
<td>n x n x 1</td>
</tr>
</tbody>
</table>

Effect: In the pattern table of the workstation state list, the given pattern index is associated with the specified parameters.

A grid of DX x DY cells is specified. The colour is given individually for each cell by a colour index, a pointer into the colour table of the workstation. The arrangement of cells is described in 4.4.6.

If the workstation supports interior style PATTERN, the pattern table in the workstation state list has predefined entries taken from the workstation description table; a number (see Table 2 in 4.10) are predefined for every workstation supporting interior style PATTERN. Any table entry (including the predefined entries) may be redefined with this function.

When a fill area is displayed, if the currently selected (either via the fill area bundle or individually, depending upon the corresponding ASF) interior style is PATTERN, the currently selected style index refers to an entry in the pattern table. If fill areas are displayed with a pattern index that is not present in the pattern table, pattern index 1 will be used. If pattern index 1 is not present (i.e. interior style PATTERN is not supported for this workstation), the result is workstation dependent.

References:

4.4.6
4.5.3

Errors:

7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20  Specified workstation identifier is invalid
25  Specified workstation is not open
33  Specified workstation is of category MI
35  Specified workstation is of category INPUT
36  Specified workstation is Workstation Independent Segment Storage
85  Specified pattern index is invalid
90  Interior style PATTERN is not supported on this workstation
91  Dimensions of colour array are invalid
93  Colour index is invalid

SET COLOUR REPRESENTATION  WSOP,WSAC,SGOP  Lma

Parameters:

In  workstation identifier  N
In  colour index  (0..n)  1
In  colour (red,green,blue intensities)  [0,1]  3xR

Effect: In the colour table of the workstation state list, the given colour index is associated with the specified colour. The colour is mapped by the workstation to the nearest available.

The colour table in the workstation state list has predefined entries taken from the workstation description table; at least indices 0 and 1 are predefined for every workstation of category OUTPUT and OUTIN. Any table entry (including the predefined entries) may be redefined with this function.

When output primitives are displayed, the colour index refers to an entry in the colour table. If output primitives are displayed with a colour index that is not present in the colour table, a workstation dependent colour index will be used. The background colour is defined by colour index 0.

NOTE: On monochrome workstations, the intensity is computed from the colour values in a workstation dependent way.
References:

4.4.2
4.4.9
4.5.3

Errors:

7  GKS not in proper state; GKS shall be in one of the states WSOP, WSAC or SGOP
20  Specified workstation identifier is invalid
25  Specified workstation is not open
33  Specified workstation is of category MI
35  Specified workstation is of category INPUT
36  Specified workstation is Workstation Independent Segment Storage
93  Colour index is invalid
96  Colour is outside range [0,1]
5.5 Transformation Functions

5.5.1 Normalization Transformation

SET WINDOW

Parameters:

- Transformation number
  - In: (1..n) 1
- Window limits 
  - In: XMIN<XMAX, YMIN<YMAX WC 4 xR

Effect: The window limits entry of the specified normalization transformation in the GKS state list is set to the value specified by the parameter.

References:

- 4.6.1
- 4.8.4

Errors:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 50 Transformation number is invalid
- 51 Rectangle definition is invalid

SET VIEWPORT

Parameters:

- Transformation number
  - In: (1..n) 1
- Viewport limits 
  - In: XMIN<XMAX, YMIN<YMAX ND C 4 xR

Effect: The viewport limits entry of the specified normalization transformation in the GKS state list is set to the value specified by the parameter. If the 'current normalization transformation number' entry in the GKS state list is the same as the specified transformation number, the 'clipping rectangle' entry in the GKS state list is set to the specified viewport limits.

References:

- 4.6.1
- 4.8.4

Errors:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 50 Transformation number is invalid
- 51 Rectangle definition is invalid
- 52 Viewport is not within the Normalized Device Coordinate unit square

SET VIEWPORT INPUT PRIORITY

Parameters:

- Transformation number
  - In: (0..n) 1
- Reference transformation number
  - In: (0..n) 1
- Relative priority
  - In: (HIGHER, LOWER) E
Transformation Functions

Effect: The viewport input priority of the specified normalization transformation in the GKS state list is set to the next higher or next lower priority relative to the reference transformation according to the specified relative priority. If the specified transformation number is the same as the reference transformation number, the function has no effect.

References:
4.6.4
4.8.4

Errors:
8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
50 Transformation number is invalid

SELECT NORMALIZATION TRANSFORMATION GKOP,WSOP,WSAC,SGOP Lma

Parameters:
In transformation number (0..n) I

Effect: The 'current normalization transformation number' entry in the GKS state list is set to the value specified by the parameter. The 'clipping rectangle' entry in the GKS state list is set to the viewport limits of the specified transformation number.

References:
4.6.1

Errors:
8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
50 Transformation number is invalid

SET CLIPPING INDICATOR GKOP,WSOP,WSAC,SGOP Lma

Parameters:
In clipping indicator (CLIP,NOCLIP) E

Effect: The 'clipping indicator' entry in the GKS state list is set to the value specified by the parameter.

References:
4.6.2
4.7.4
4.7.6
4.8.4

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

SET WORKSTATION WINDOW

Parameters:

- **In** workstation identifier
- **In** workstation window limits \( X_{\text{MIN}} < X_{\text{MAX}}, Y_{\text{MIN}} < Y_{\text{MAX}} \) \( \text{NDC} \)

Effect: The ‘requested workstation window’ entry in the workstation state list of the specified workstation is set to the value specified by the parameter.

If the ‘dynamic modification accepted for workstation transformation’ entry in the workstation description table is set to IMM, or if the ‘display surface empty’ entry in the workstation state list is set to EMPTY, then the ‘current workstation window’ entry in the workstation state list is set to the value specified by the parameter and the ‘workstation transformation update state’ entry is set to NOTPENDING. Otherwise the ‘workstation transformation update state’ entry in the workstation state list is set to PENDING and the ‘current workstation window’ entry is not changed.

References:
4.6.3

Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20  Specified workstation identifier is invalid
25  Specified workstation is not open
33  Specified workstation is of category MI
36  Specified workstation is Workstation Independent Segment Storage
51  Rectangle definition is invalid
53  Workstation window is not within the Normalized Device Coordinate unit square

SET WORKSTATION VIEWPORT

Parameters:

- **In** workstation identifier
- **In** workstation viewport limits \( X_{\text{MIN}} < X_{\text{MAX}}, Y_{\text{MIN}} < Y_{\text{MAX}} \) \( \text{DC} \)

Effect: The ‘requested workstation viewport’ entry in the workstation state list of the specified workstation is set to the value specified by the parameter.

If the ‘dynamic modification accepted for workstation transformation’ entry in the workstation description table is set to IMM, or if the ‘display surface empty’ entry in the workstation state list is set to EMPTY, then the ‘current workstation viewport’ entry in the workstation state list is set to the value specified by the parameter and the ‘workstation transformation update state’ entry is set to NOTPENDING. Otherwise the ‘workstation transformation update state’ entry in the workstation state list is set to PENDING and the ‘current workstation viewport’ entry is not changed.

References:
4.6.3
Errors:

7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20  Specified workstation identifier is invalid
25  Specified workstation is not open
38  Specified workstation is of category MI
36  Specified workstation is Workstation Independent Segment Storage
51  Rectangle definition is invalid
54  Workstation viewport is not within the display space
5.6 Segment functions

5.6.1 Segment manipulation functions

CREATE SEGMENT

Parameters:

In    segment name N

Effect: GKS is set into the operating state SGOP = 'Segment open'. The segment state list is set up and initialised as indicated in 6.7. The segment name is recorded as the 'name of the open segment' in the GKS state list (see 6.4). All subsequent output primitives until the next CLOSE SEGMENT will be collected into this segment. The segment name is entered in the 'set of stored segments for this workstation' in the workstation state list (see 6.5) for every active workstation. All active workstations are included in the 'set of associated workstations' of the segment state list of the newly opened segment. The segment name is entered into the 'set of segment names in use' in the GKS state list. Primitive attributes are not affected.

References:

4.7.1

Errors:

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

CLOSE SEGMENT

Parameters:

none

Effect: GKS is put into the operating state WSAC = 'At least one workstation active'. Primitives may no longer be added to the previously open segment. The 'name of the open segment' in the GKS state list (see 6.4) becomes unavailable for inquiry.

References:

4.7.1

Errors:

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

RENAME SEGMENT

Parameters:

In    old segment name
In    new segment name

Effect: Each occurrence of old segment name in the 'set of stored segments for this workstation' in each workstation state list (see 6.5) and in the 'set of segment names in use' in the GKS state list is replaced by new segment name. If old segment name is the name of the open segment, the 'name of the open segment' in the GKS state list is set to new segment name.
NOTE The old segment name may be reused by the application program.

References:

4.7.1

Errors:

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

DELETE SEGMENT

Parameters:

In segment name

Effect: The segment is deleted. The segment name is removed from each 'set of stored segments for this workstation' (in the workstation state lists (see 6.5)) which contains it and from the 'set of segment names in use' in the GKS state list. The segment's state list is cancelled.

NOTE The segment name may be reused by the application program.

References:

4.5.3
4.7.1

Errors:

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

DELETE SEGMENT FROM WORKSTATION

Parameters:

In workstation identifier
In segment name

Effect: The segment is deleted from the specified workstation. The segment name is removed from the 'set of stored segments for this workstation' in the workstation state list (see 6.5). The workstation identifier is removed from the 'set of associated workstations' in the segment state list (see 6.7). If the 'set of associated workstations' becomes empty, the segment is deleted, i.e. the DELETE SEGMENT function is performed.

References:

4.5.3
4.7.1
GKS Functions

Segment functions

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

ASSOCIATE SEGMENT WITH WORKSTATION  WSOP, WSAC  L2a
Parameters:
   In  workstation identifier  N
   In  segment name  N
Effect: The segment is sent to the specified workstation in the same way as if the workstation were active when the segment was created. Clipping rectangles are copied unchanged. The segment name is added to the 'set of stored segments for this workstation' in the workstation state list (see 6.5). The workstation identifier is included in the 'set of associated workstations' in the segment state list (see 6.7).
NOTE: If the specified segment is not present in the Workstation Independent Segment Storage, an error occurs. If the segment is already associated with the specified workstation, the function has no effect.

References:
4.5.3
4.7.1
4.7.6

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

COPY SEGMENT TO WORKSTATION  WSOP, WSAC  L2a
Parameters:
   In  workstation identifier  N
   In  segment name  N
Effect: The primitives in the segment are sent to the specified workstation after segment transformation and clipping at the clipping rectangle stored with each primitive. The primitives are not stored in a segment.
NOTE. If the specified segment is not present in the Workstation Independent Segment Storage, an error occurs. The specified workstation cannot be the Workstation Independent Segment Storage.

All primitives keep the values of the primitive attributes (for example, polyline index, character path, pick identifier), that were assigned to them when they were created, for their whole lifetime (see 4.7.1). In particular, when segments are copied, the values of the primitive attributes within the copied segments are unchanged.

References:

4.5.3
4.7.1
4.7.5

Errors:

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

INSERT SEGMENT

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>segment name</td>
<td>N</td>
<td>In</td>
</tr>
<tr>
<td>transformation matrix</td>
<td>2×3×R</td>
<td>In</td>
</tr>
</tbody>
</table>

Effect: Having been transformed as described below, the primitives contained in the segment are copied either (in state SGOP) into the open segment or (in state WSAC) into the stream of primitives outside segments.

In both cases the transformed primitives are sent to all active workstations. The coordinates of the primitives contained in the inserted segment are transformed, firstly, by the segment transformation of the inserted segment, and, secondly, by applying the following matrix multiplication to them:

\[
\begin{bmatrix}
    x' \\
    y'
\end{bmatrix} =
\begin{bmatrix}
    M_{11} & M_{12} & M_{13} \\
    M_{21} & M_{22} & M_{23}
\end{bmatrix}
\begin{bmatrix}
    x \\
    y \\
    1
\end{bmatrix}
\]

The original coordinates are \((x, y)\), the transformed coordinates are \((x', y')\), both in NDC. The values \(M_{13}\) and \(M_{23}\) of the transformation matrix are NDC coordinates, the other values are unitless. For geometric attributes which are vectors (for example, CHARACTER UP VECTOR), the values \(M_{13}\) and \(M_{23}\) are ignored.

The insert transformation (conceptually) takes place in NDC space. Other than the segment transformation, attributes of the inserted segment are ignored.

All clipping rectangles in the inserted segment are ignored. Each primitive processed is assigned a new clipping rectangle which is the clipping rectangle in the GKS state list if the ‘clipping indicator’ entry in the GKS state list is CLIP and is \([0,1] \times [0,1]\) if the ‘clipping indicator’ entry in the GKS state list is NOCLIP. All primitives processed by a single invocation of INSERT SEGMENT receive the same clipping rectangle.
NOTE: If the specified segment is not in the Workstation Independent Segment Storage or is the open segment, an error occurs.

All primitives keep the values of the primitive attributes (for example, POLYLINE INDEX, TEXT PATH, PICK IDENTIFIER), that were assigned to them when they were created, for their whole lifetime (see 4.7.1). In particular, when segments are inserted, the values of the primitive attributes within the inserted segments are unchanged. The values of primitive attributes in the GKS state list, that are used in the creation of subsequent primitives within the segment into which the insertion takes place, are not changed by that insertion.

References:

4.5.3
4.7.6

Errors:

5 GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
27 Workstation Independent Segment Storage is not open
120 Specified segment name is invalid
124 Specified segment does not exist on Workstation Independent Segment Storage
125 Specified segment is open

5.6.2 Segment attributes

SET SEGMENT TRANSFORMATION WSOP, WSAC, SGOP

Parameters:

In segment name N
In transformation matrix 2×3×R

Effect: The 'segment transformation matrix' entry in the segment state list of the named segment is set to the value specified by the parameter. When a segment is displayed, the coordinates of its primitives are transformed by applying the following matrix multiplication to them:

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} = \begin{bmatrix}
  M_{11} & M_{12} & M_{13} \\
  M_{21} & M_{22} & M_{23}
\end{bmatrix} \times \begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

The original coordinates are \((x, y)\), the transformed coordinates are \((x', y')\), both in NDC. The values \(M_{13}\) and \(M_{23}\) of the transformation matrix are in NDC coordinates, the other values are unitless. For geometric attributes which are vectors (for example, CHARACTER UP VECTOR), the values \(M_{13}\) and \(M_{23}\) are ignored.

This function can be used to transform a segment stored on a workstation. The transformation applies to all workstations where the specified segment is stored even if they are not all active.

The segment transformation (conceptually) takes place in NDC space. The segment transformation will be stored in the segment state list and will not affect the contents of the segment. The segment transformation is not cumulative, i.e. it always applies to the segment as originally created.

NOTE: Applying the same segment transformation twice to a segment gives identical results. The identity transformation shows the segment in its original geometrical appearance.
Segment functions

References:

4.5.3
4.7.3

Errors:

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

SET VISIBILITY

Parameters:

In  segment name
In  visibility  (VISIBLE,INVISIBLE) E

Effect: The ‘visibility’ entry in the segment state list of the named segment is set to the value specified by the parameter.

References:

4.5.3
4.7.2
4.8.4

Errors:

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

SET HIGHLIGHTING

Parameters:

In  segment name
In  highlighting  (NORMAL,HIGHLIGHTED) E

Effect: The ‘highlighting’ entry in the segment state list of the named segment is set to the value specified by the parameter. If the segment is marked as HIGHLIGHTED and VISIBLE, the primitives in it are highlighted in an implementation dependent manner.

References:

4.5.3
4.7.2

Errors:

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

Parameters:
- In segment name N
- In segment priority [0,1] R

Effect: The 'segment priority' entry in the segment state list of the named segment is set to the value specified by the parameter. Segment priority affects the display of segments and pick input if segments overlap, in which case GKS gives precedence to segments with higher priority. If segments with the same priority overlap, the result is implementation dependent.

NOTE The use of segment priority applies only to workstations where the entry 'number of segment priorities supported' in the workstation description table is greater than 1 or equal to 0 (indicating that a continuous range of priorities is supported).

If 'number of segment priorities supported' is greater than 1, the range [0,1] for segment priority is mapped onto the range 1 to 'number of segment priorities supported' for a specific workstation before being used by a device driver. If 'number of segment priorities supported' is equal to 0, the implementation allows all values of segment priority to be differentiated.

This feature is intended to address appropriate hardware capabilities only. It cannot be used to force software checking of interference between segments on non-raster displays.

The segment priority is also used for picking segments. When overlapping or intersecting segments are picked, the segment with higher priority is delivered as a result of the pick input primitive. All workstations having pick input provide this mechanism.

References:
- 4.5.3
- 4.7.2
- 4.8.4

Errors:
- 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 120 Specified segment name is invalid
- 122 Specified segment does not exist
- 126 Segment priority is outside the range [0,1]

SET DETECTABILITY

Parameters:
- In segment name N
- In detectability (UNDETECTABLE, DETECTABLE) E

Effect: The 'detectability' entry in the segment state list of the named segment is set to the value specified by the parameter. If the segment is marked as DETECTABLE and VISIBLE, the primitives in it are available for pick input. DETECTABLE but INVISIBLE segments cannot be picked.

References:
- 4.7.2
- 4.8.4
Errors:

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

5.7.1 Initialization of Input Devices

INITIALISE LOCATOR

Parameters:

- In workstation identifier
- In locator device number
- In initial normalization transformation number
- In initial locator position
- In prompt and echo type
- In echo area XMIN<XMAX,YMIN<YMAX
- In locator data record

Effect: The initial locator position, initial normalization transformation number, prompt and echo type, echo area and locator data record are stored in the workstation state list entry for the specified LOCATOR device.

For some LOCATOR prompt and echo types, two positions are required. One of the positions, which remains fixed during the input operation, is the initial locator position. The other position is the current locator position that varies dynamically as the operator uses the LOCATOR.

Prompt and echo type:

1. designates the current position of the LOCATOR using an implementation-defined technique.
2. crosshair, i.e. designate the current position of the LOCATOR using a vertical line and a horizontal line spanning over the display surface or the workstation viewport intersecting at the current locator position.
3. designate the current position of the LOCATOR using a tracking cross.
4. designate the current position of the LOCATOR using a rubber band line connecting the initial locator position given by this function and the current locator position.
5. designate the current position of the LOCATOR using a rectangle. The diagonal of the rectangle is the line connecting the initial locator position given by this function and the current locator position.
6. display a digital representation of the current position of the LOCATOR in LOCATOR device dependent coordinates within the echo area.
7. reserved for registration or future standardization.

NOTE LOCATOR prompt and echo type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a LOCATOR prompt and echo type has been approved by the ISO Working Group on Computer Graphics, the LOCATOR prompt and echo type value will be assigned by the Registration Authority.

References:

4.8.2
4.8.6

Data Record

Prompt and echo type = 4

Input Functions

1. attribute control flag
   
   if attribute control flag = SPECIFIED
   
   2. linetype ASF
   3. linewidth scale factor ASF
   4. polyline colour index ASF
   5. polyline index
   6. linetype index
   7. linetype scale factor
   8. polyline colour index

   if attribute control flag = CURRENT, the current polyline attributes at LOCATOR initialization are used.

Prompt and echo type = 5

1. polyline/fill area control flag
2. attribute control flag

   if attribute control flag = SPECIFIED and polyline/fill area control flag = POLYLINE
   
   3. linetype ASF
   4. linewidth scale factor ASF
   5. polyline colour index ASF
   6. polyline index
   7. linetype index
   8. linewidth scale factor
   9. polyline colour index

   if attribute control flag = SPECIFIED and polyline/fill area control flag = FILL AREA

3. fill area interior style ASF
4. fill area style index ASF
5. fill area colour index ASF
6. fill area index
7. fill area interior style
8. fill area style index
9. fill area colour index

   If attribute control flag = CURRENT and polyline/fill area control flag = POLYLINE, then the current polyline attributes at LOCATOR initialization will be used. If attribute control flag = CURRENT and polyline/fill area control flag = FILL AREA, then the current fill area attributes at LOCATOR initialization will be used.
Errors:

7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
38 Specified workstation is neither of category INPUT nor of category OUTIN
51 Rectangle definition is invalid
60 Polyline index is invalid
68 Linetype is equal to zero
65 Linewidth scale factor is less than zero
84 Fill area index is invalid
92 Colour index is less than zero
140 Specified input device is not present on workstation
141 Input device is not in REQUEST mode
144 Specified prompt and echo type is not supported on this workstation
145 Echo area is outside display space
146 Contents of input data record are invalid
152 Initial value is invalid

INITIALISE STROKE

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>In  workstation identifier</td>
<td>N</td>
</tr>
<tr>
<td>In  stroke device number</td>
<td>N</td>
</tr>
<tr>
<td>In  initial normalization transformation number</td>
<td>I</td>
</tr>
<tr>
<td>In  number of points in initial stroke</td>
<td>I</td>
</tr>
<tr>
<td>In  points in initial stroke</td>
<td>WC</td>
</tr>
<tr>
<td>In  prompt and echo type</td>
<td>-n..1,n</td>
</tr>
<tr>
<td>In  echo area XMIN&lt;XMAX, YMIN&lt;YMAX</td>
<td>DC</td>
</tr>
<tr>
<td>In  stroke data record</td>
<td>D</td>
</tr>
</tbody>
</table>

Effect: The initial stroke, initial normalization transformation number, prompt and echo type, echo area and stroke data record are stored in the workstation state list entry for the specified STROKE device.

For all prompt and echo types, the first entry in the stroke data record is the input buffer size which is an integer in the range (1..n). This is compared against an implementation defined 'maximum input buffer size' for this device (contained in the workstation description table). If the requested buffer size is greater, the 'maximum input buffer size' is substituted in the stored data record. If the initial stroke is longer than the buffer size, an error is issued.

When a STROKE measure process comes into existence, it obtains a buffer of the current input buffer size. The initial stroke is copied into the buffer, and the editing position is placed at the initial buffer editing position within it. Replacement of points begins at this initial position. If the initial buffer editing position cannot be specified in the stroke data record, the value 1 is used.

Prompt and echo types:

<0 prompting and echoing is STROKE device dependent.
1  display the current stroke using an implementation defined technique.
2 display a digital representation of the current stroke position within the echo area.
3 display a marker at each point of the current stroke.
4 display a line joining successive points of the current stroke.
5 reserved for registration or future standardization.

If the operator enters more points than the current input buffer size, the additional points are lost. The operator should be informed of this situation.

Stroke data record entries for variables such as intervals in X, Y and time may be provided to constrain the number of points delivered.

NOTE. For all prompt and echo types, the stroke data record may contain an initial buffer editing position, which may range from 1 to length of initial stroke plus 1.

STROKE prompt and echo type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a STROKE prompt and echo type has been approved by the ISO Working Group on Computer Graphics\(^1\), the STROKE prompt and echo type value will be assigned by the Registration Authority.

References:

4.8.2
4.8.6

Data Record

\begin{verbatim}
1 input buffer size (l.n)
2 editing position (l.n)
3 x,y interval WC > 0 2XR
4 time interval (seconds) R

Prompt and echo type = 3

5 attribute control flag (CURRENT, SPECIFIED) E

if attribute control flag = SPECIFIED

6 marker type ASF E
7 marker size scale factor ASF (BUNDLED, INDIVIDUAL) E
8 polymarker colour index ASF (BUNDLED, INDIVIDUAL) E
9 polymarker index (l.n) I
10 marker type index (-n...1.n) I
11 marker size scale factor >0 R
12 polymarker colour index (0.n) I

Prompt and echo type = 4

5 attribute control flag (CURRENT, SPECIFIED) E

if attribute control flag = SPECIFIED

6 linetype ASF (BUNDLED, INDIVIDUAL) E
7 linewidth scale factor ASF (BUNDLED, INDIVIDUAL) E
8 polyline colour index ASF (BUNDLED, INDIVIDUAL) E
9 polyline index (l.n) I
10 linetype index (-n...1.n) I
11 linewidth scale factor (0.n) I
12 polyline colour index R
\end{verbatim}

\(^1\) Currently, the Working Group on Computer Graphics is known as ISO/TC97/SC5/WG2.
Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
88 Specified workstation is neither of category INPUT nor of category OUTIN
51 Rectangle definition is invalid
60 Polyline index is invalid
68 Linetype is equal to zero
66 Polymarker index is invalid
65 Linewidth scale factor is less than zero
67 A representation for the specified polymarker index has not been defined on this workstation
69 Marker type is equal to zero
92 Colour index is less than zero
140 Specified input device is not present on workstation
141 Input device is not in REQUEST mode
144 Specified prompt and echo type is not supported on this workstation
145 Echo area is outside display space
146 Contents of input data record are invalid
152 Initial value is invalid
158 Number of points in the initial stroke is greater than the buffer size

INITIALISE VALUATOR

Parameters:
In workstation identifier N
In valuator device number I
In initial value R
In prompt and echo type (-n..-1,1..n) I
In echo area XMIN<XMAX,YMIN<YMAX DC 4 XR
In valuator data record D

Effect: The initial value, prompt and echo type, echo area and valuator data record are stored in the workstation state list entry for the specified VALUATOR device. For all VALUATOR prompt and echo types, the valuator data record includes, in the first two positions, a low value and a high value, in that order, specifying the range. The values from the device will be scaled linearly to the specified range.

Prompt and echo types:
<0prompting and echoing is VALUATOR device dependent.
1 designate the current VALUATOR value using an implementation defined technique.
2 display a graphical representation of the current VALUATOR value within the echo area (for example, a dial or a pointer).
3 display a digital representation of the current VALUATOR value within the echo area.
≥4reserved for registration or future standardization.

NOTE VALUATOR prompt and echo type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a VALUATOR prompt and echo type has been approved by the ISO Working Group on Computer Graphics, the VALUATOR prompt and echo

type value will be assigned by the Registration Authority.

References:
4.8.2
4.8.6

Data Record:

1  low value of valuator range  
2  high value of valuator

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

INITIALISE CHOICE

Parameters:

In  workstation identifier  
In  choice device number  (1..n)  I
In  initial status  (OK,NOCHOICE)  E
In  initial choice number  (1..n)  I
In  prompt and echo type  (-n..-1,1..n)  I
In  echo area XMIN<XMAX,YMIN<YMAX  DC  4XR
In  choice data record

Effect: The initial status, initial choice number, prompt and echo type, echo area and choice data record are stored in the workstation state list entry for the specified CHOICE device.

Prompt and echo types:

<0prompting and echoing is CHOICE device dependent.

1  designate the current CHOICE number using an implementation defined technique.

2  the physical input devices that are most commonly used to implement a CHOICE logical input device normally have a built-in prompting capability. This prompt and echo type allows the application program to invoke this prompting capability. If the value of the i-th element of 'prompt array' in the choice data record is OFF, prompting of the i-th alternative of the specified choice input device is turned off. An ON value indicates that prompting for that alternative is turned on. The first entry in the choice data record is the number of choice alternatives. This is compared against an implementation defined 'maximum number of choice alternatives' for this device (contained in the workstation description table). If the maximum value is exceeded, an error is issued. The second entry in the choice data record is the 'prompt array'.
3 allow the operator to indicate a CHOICE number by selecting, using an appropriate technique, one of a set of CHOICE strings. The CHOICE strings are contained in the choice data record and are displayed within the echo area. The logical input value is the number of the string selected. The first entry in the choice data record is the number of choice strings. This is compared against an implementation defined 'maximum number of choice alternatives' for this device (contained in the workstation description table). If the maximum value is exceeded, an error is issued. The second entry in the choice data record is the 'array of choice strings'.

4 allow the operator to indicate a CHOICE number by selecting, via an alphanumeric keyboard, one of a set of CHOICE strings. The CHOICE strings are contained in the choice data record and may be displayed in the echo area as a prompt. The string typed in by the operator is echoed in the echo area. The logical input value is the number of the first string in the array that has been typed in by the operator. The first entry in the choice data record is the number of choice strings. This is compared against an implementation defined 'maximum number of choice alternatives' for this device (contained in the workstation description table). If the maximum value is exceeded, an error is issued. The second entry in the choice data record is the 'array of choice strings'.

5 the segment named by the choice data record is interpreted during execution of INITIALISE CHOICE for later use as a prompt of the specified CHOICE device. It will be displayed within the echo area by mapping the unit square [0,1] x [0,1] of NDC space onto the echo area. The pick identifiers in the segment are mapped to CHOICE numbers in a CHOICE device dependent fashion. Picking these primitives selects the corresponding CHOICE value. After the interpretation, no logical connection between the specified segment and the specified CHOICE device exists. The first entry in the choice data record is the segment name.

≥6 reserved for registration or future standardization.

NOTE CHOICE prompt and echo type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a CHOICE prompt and echo type has been approved by the ISO Working Group on Computer Graphics, the CHOICE prompt and echo type value will be assigned by the Registration Authority.

References:

4.8.2
4.8.6

Data Record:

Prompt and echo type = 2

- 1 number of choice alternatives
- 2 array of prompts

Prompt and echo type = 3

- 1 number of choice strings
- 2 array of strings

Prompt and echo type = 4

- 1 number of choice strings
- 2 array of strings

Input Functions

Prompt and echo type = 5

1 segment name N
2 number of choice alternatives (1..n) I
3 array of pick identifiers 0 \*N

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

INITIALISE PICK

WSOP,WSAC,SGOP L1b

Parameters:

- workstation identifier N
- pick device number (1..n) I
- initial status (OK,NOPICK) E
- initial segment N
- initial pick identifier N
- prompt and echo type (-n..-1,1..n) I
- echo area XMIN<XMAX,YMIN<YMAX DC 4xR
- pick data record D

Effect: The initial status, initial segment, initial pick identifier, prompt and echo type, echo area and the pick data record are stored in the workstation state list entry for the specified PICK device.

Prompt and echo types:

$<0$prompting and echoing is PICK device dependent.

1 use an implementation-defined technique that at least highlights the ‘picked’ primitive for a short period of time.

2 echo the contiguous group of primitives within the segment with the same pick identifier as the ‘picked’ primitive, or all primitives of the segment with the same pick identifier as the ‘picked’ primitive.

3 echo the whole segment containing the ‘picked’ primitive.

$\geq4$reserved for registration or future standardization.

NOTE: PICK prompt and echo type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a PICK prompt and echo type has been approved by the ISO Working Group on Computer Graphics, the PICK prompt and echo type value will be assigned by the

Data Record:
No predefined entries for Prompt and Echo types 1 through 3.

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
37 Specified workstation is not of category OUTIN
51 Rectangle definition is invalid
140 Specified input device is not present on workstation
141 Input device is not in REQUEST mode
144 Specified prompt and echo type is not supported on this workstation
145 Echo area is outside display space
146 Contents of input data record are invalid
152 Initial value is invalid

INITIALISE STRING

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>N</td>
</tr>
<tr>
<td>string device number</td>
<td>I</td>
</tr>
<tr>
<td>initial string</td>
<td>S</td>
</tr>
<tr>
<td>prompt and echo type</td>
<td>I</td>
</tr>
<tr>
<td>echo area XMIN&lt;XMAX,YMIN&lt;YMAX</td>
<td>DC</td>
</tr>
<tr>
<td>string data record</td>
<td>D</td>
</tr>
</tbody>
</table>

Effect: The initial string, prompt and echo type, echo area and string data record are stored in the workstation state list entry for the specified STRING device.

For all prompt and echo types, the first entry of the string data record is the input buffer size, which is an integer in the range (1..n). This is compared against an implementation defined 'maximum input buffer size' for this device (contained in the workstation description table). If the requested buffer size is greater, the 'maximum input buffer size' is substituted in the stored record. If the initial string is longer than the buffer size, an error is issued.

For all prompt and echo types, the second entry of the string data record is an initial cursor position, which may range from 1 to the length of the initial string plus 1.

When a STRING measure process comes into existence, it obtains a buffer of the current input buffer size. The initial string is copied into the buffer, and the cursor is placed at the initial cursor position within it. Replacement of characters begins at this cursor position.

Prompt and echo types:
- 1 displaying the current STRING value within the echo area.
- 2 reserved for registration or future standardization.
NOTE: If the operator enters more characters than the current input buffer size, the additional characters are lost.

STRING prompt and echo type values are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When a STRING prompt and echo type has been approved by the ISO Working Group on Computer Graphics\(^1\), the STRING prompt and echo type value will be assigned by the Registration Authority.

References:

4.8.2
4.8.6

Data Record

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Input buffer size (^{1.n})</td>
</tr>
<tr>
<td>2</td>
<td>Initial cursor position (^{1.n})</td>
</tr>
</tbody>
</table>

Errors:

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

5.7.2 Setting Mode of Input Devices

**SET LOCATOR MODE**

WSOP,WSAC,SGOP Lmb

Parameters:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>workstation identifier</td>
</tr>
<tr>
<td>In</td>
<td>locator device number (^{1..n})</td>
</tr>
<tr>
<td>In</td>
<td>operating mode (REQUEST, SAMPLE, EVENT)</td>
</tr>
<tr>
<td>In</td>
<td>echo switch (ECHO, NOECHO)</td>
</tr>
</tbody>
</table>

Effect: The given LOCATOR device is set to the specified operating mode and its echoing state is set to ECHO or NOECHO. Depending on the specified operating mode, an interaction with the given device may begin or end. The input device state defined by 'operating mode' and 'echo switch' is stored in the workstation state list for the given LOCATOR device.

References:

4.8.1
4.8.3

\(^1\) Currently, the Working Group on Computer Graphics is known as ISO/TC97/SC5/WG2.
GKS Functions

Input Functions

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

SET STROKE MODE

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>N</td>
</tr>
<tr>
<td>stroke device number</td>
<td>(1..n)</td>
</tr>
<tr>
<td>operating mode</td>
<td>(REQUEST,SAMPLE,EVENT)</td>
</tr>
<tr>
<td>echo switch</td>
<td>(ECHO,NOECHO)</td>
</tr>
</tbody>
</table>

Effect: The given STROKE device is set to the specified operating mode and its echoing state is set to ECHO or NOECHO. Depending on the specified operating mode, an interaction with the given device may begin or end. The input device state defined by 'operating mode' and 'echo switch' is stored in the workstation state list for the given STROKE device.

References:
  4.8.1
  4.8.3

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

SET VALUATOR MODE

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>N</td>
</tr>
<tr>
<td>valuator device number</td>
<td>(1..n)</td>
</tr>
<tr>
<td>operating mode</td>
<td>(REQUEST,SAMPLE,EVENT)</td>
</tr>
<tr>
<td>echo switch</td>
<td>(ECHO,NOECHO)</td>
</tr>
</tbody>
</table>

Effect: The given VALUATOR device is set to the specified operating mode and its echoing state is set to ECHO or NOECHO. Depending on the specified operating mode, an interaction with the given device may begin or end. The input device state defined by 'operating mode' and 'echo switch' is stored in the workstation state list for the given VALUATOR device.

References:
  4.8.1
  4.8.3
Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
148 EVENT and SAMPLE input mode are not available at this level of GKS

SET CHOICE MODE

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation</td>
<td>N</td>
</tr>
<tr>
<td>device number</td>
<td>(1..n)</td>
</tr>
<tr>
<td>operating mode</td>
<td>(REQUEST, SAMPLE, EVENT)</td>
</tr>
<tr>
<td>echo switch</td>
<td>(ECHO, NOECHO)</td>
</tr>
</tbody>
</table>

Effect: The given CHOICE device is set to the specified operating mode and its echoing state is set to ECHO or NOECHO. Depending on the specified operating mode, an interaction with the given device may begin or end. The input device state defined by ‘operating mode’ and ‘echo switch’ is stored in the workstation state list for the given CHOICE device.

References:
4.8.1
4.8.3

Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
148 EVENT and SAMPLE input mode are not available at this level of GKS

SET PICK MODE

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation</td>
<td>N</td>
</tr>
<tr>
<td>device number</td>
<td>(1..n)</td>
</tr>
<tr>
<td>operating mode</td>
<td>(REQUEST, SAMPLE, EVENT)</td>
</tr>
<tr>
<td>echo switch</td>
<td>(ECHO, NOECHO)</td>
</tr>
</tbody>
</table>

Effect: The given PICK device is set to the specified operating mode and its echoing state is set to ECHO or NOECHO. Depending on the specified operating mode, an interaction with the given device may begin or end. The input device state defined by ‘operating mode’ and ‘echo switch’ is stored in the workstation state list for the given PICK device.

References:
4.8.1
4.8.3
GKS Functions

Errors:

1. GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
2. Specified workstation identifier is invalid
3. Specified workstation is not open
4. Specified workstation is not of category OUTIN
5. Specified input device is not present on workstation
6. EVENT and SAMPLE input mode are not available at this level of GKS

SET STRING MODE

Parameters:

| In workstation identifier | N |
| In string device number | (1..n) I |
| In operating mode | (REQUEST,SAMPLE,EVENT) E |
| In echo switch | (ECHO,NOECHO) E |

Effect: The given STRING device is set to the specified operating mode and its echoing state is set to ECHO or NOECHO. Depending on the specified operating mode, an interaction with the given device may begin or end. The input device state defined by 'operating mode' and 'echo switch' is stored in the workstation state list for the given STRING device.

References:

4.8.1
4.8.3

Errors:

1. GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
2. Specified workstation identifier is invalid
3. Specified workstation is not open
4. Specified workstation is neither of category INPUT nor of category OUTIN
5. Specified input device is not present on workstation
6. EVENT and SAMPLE input mode are not available at this level of GKS

5.7.8 Request Input Functions

REQUEST LOCATOR

Parameters:

| In workstation identifier | N |
| In locator device number | (1..n) I |
| Out status | (OK,NONE) E |
| Out normalization transformation number | (0..n) I |
| Out locator position | WC P |

Effect: GKS performs a REQUEST on the specified LOCATOR device. If the break facility is invoked by the operator, the status NONE is returned; otherwise OK is returned together with the logical input value which is the current measure of the LOCATOR device. This measure consists of a locator position in world coordinates and the normalization transformation number, which was used in the conversion to world coordinates. The locator position is within the window of the normalization transformation.
REQUEST STROKE

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
<th>WSOP, WSAC, SGOP</th>
<th>Lmb</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>status</td>
<td>(OK, NONE)</td>
<td>E</td>
</tr>
<tr>
<td>stroke device number</td>
<td>normalization transformation number</td>
<td>(0..n)</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td>number of points</td>
<td>(0..n)</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td>points in stroke</td>
<td>WC</td>
<td>(n \times p)</td>
</tr>
</tbody>
</table>

Effect: GKS performs a REQUEST on the specified STROKE device. If the break facility is invoked by the operator, the status NONE is returned; otherwise OK is returned together with the logical input value which is the current measure of the STROKE device. This consists of a sequence of not more than 'input buffer size' (in the stroke data record) points in world coordinates, and the normalization transformation number, which was used in the conversion to world coordinates. The points in the stroke all lie within the window of the normalization transformation.

NOTE: If an operator enters more points than the stroke input buffer size (in the workstation state list) allows, the additional points are lost. The operator should be informed of this situation.

References:

4.6.5
4.8.1
4.8.2
4.8.3
4.8.4

Errors:

7 \( GKS \) not in proper state: \( GKS \) shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
141 Input device is not in REQUEST mode
GKS Functions

REQUEST VALUATOR

Parameters:

- In workstation identifier: N
- In valuator device number: (1..n) I
- Out status: (OK, NONE) E
- Out value: R

Effect: GKS performs a REQUEST on the specified VALUATOR device. If the break facility is invoked by the operator, the status NONE is returned; otherwise OK is returned together with the logical input value which is the current measure of the VALUATOR device. The value delivered is in the range specified in the workstation state list entry (for this device) in the data record.

Errors:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
88 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
141 Input device is not in REQUEST mode

REQUEST CHOICE

Parameters:

- In workstation identifier: N
- In choice device number: (1..n) I
- Out status: (OK, NOCHOICE, NONE) E
- Out choice number: (1..n) I

Effect: GKS performs a REQUEST on the specified CHOICE device. If the break facility is invoked by the operator, the status NONE is returned; if the measure of the CHOICE device indicates no choice, status NOCHOICE is returned; otherwise OK is returned together with a choice number which is set according to the current measure of the CHOICE device.

References:

4.8.1
4.8.2
4.8.3
4.8.4
REQUEST PICK

Parameters:

In workstation identifier N
In pick device number (1..n) I
Out status (OK,NOPICK,NONE) E
Out segment name N
Out pick identifier N

Effect: GKS performs a REQUEST on the specified PICK device. If the break facility is invoked by the operator, the status NONE is returned; if the measure of the PICK device indicates no pick, status NOPICK is returned; otherwise OK is returned together with a segment name and a pick identifier which are set according to the current measure of the PICK device. The pick identifier is associated with the primitive, within the segment, that was picked.

References:
4.8.1
4.8.2
4.8.3
4.8.4

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
37 Specified workstation is not of category OUTIN
140 Specified input device is not present on workstation
141 Input device is not in REQUEST mode

REQUEST STRING

Parameters:

In workstation identifier N
In string device number (1..n) I
Out status (OK,NONE) E
Out character string S

Effect: GKS performs a REQUEST on the specified STRING device. If the break facility is invoked by the operator, the status NONE is returned; otherwise OK is returned together with the logical input value which is the current measure of the STRING device.
NOTE: The length of the returned string is less than or equal to the buffer size specified in the workstation state list entry (for this device) in the data record.

References:

4.8.1
4.8.2
4.8.3
4.8.4

Errors:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
141 Input device is not in REQUEST mode

5.7.4 Sample Input Functions

SAMPLE LOCATOR

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>workstation identifier</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>locator device number</td>
<td>(1..n)</td>
</tr>
<tr>
<td>Out</td>
<td>normalization transformation number</td>
<td>(0..n)</td>
</tr>
<tr>
<td>Out</td>
<td>locator position</td>
<td>WC</td>
</tr>
</tbody>
</table>

Effect: The logical input value, which is the current measure of the specified LOCATOR device, is returned. The measure consists of a locator position in world coordinates and the normalization transformation number, which was used in the conversion to world coordinates. The locator position is within the window of the normalization transformation.

References:

4.6.4
4.8.1
4.8.3
4.8.4

Errors:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
142 Input device is not in SAMPLE mode

SAMPLE STROKE

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>workstation identifier</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>stroke device number</td>
<td>(1..n)</td>
</tr>
<tr>
<td>Out</td>
<td>normalization transformation number</td>
<td>(0..n)</td>
</tr>
<tr>
<td>Out</td>
<td>number of points</td>
<td>(0..n)</td>
</tr>
</tbody>
</table>
Out points in stroke WC n XP

Effect: The logical input value, which is the current measure of the specified STROKE device, is returned. The measure consists of a sequence of points in world coordinates, and the normalization transformation number which was used in the conversion to world coordinates. The points in the stroke all lie within the window of the normalization transformation.

NOTE: If an operator enters more points than the stroke input buffer size (in the workstation state list) allows, the additional points are lost. It is anticipated that the operator would be informed of this situation.

References:
4.6.5
4.8.1
4.8.3
4.8.4

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
88 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
142 Input device is not in SAMPLE mode

SAMPLE VALUATOR

Parameters:
| In   | workstation identifier | N |
| In   | valuator device number  | (1..n)  |
| Out  | value                   | R |

Effect: The logical input value, which is the current measure of the specified VALUATOR device, is returned. The value delivered is in the range specified in the workstation state list entry (for this device) in the data record.

References:
4.8.1
4.8.3
4.8.4

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
88 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
142 Input device is not in SAMPLE mode

SAMPLE CHOICE

Parameters:
| In   | workstation identifier | N |
| In   | choice device number    | (1..n) |

References:
GKS Functions

Out status (OK, NOCHOICE) E
Out choice number (1..n) I

Effect: If the current measure of the specified CHOICE device is indicating no choice, status NOCHOICE is returned; otherwise OK is returned together with a choice number which is set according to the current measure of the CHOICE device.

References:
4.8.1
4.8.3
4.8.4

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
142 Input device is not in SAMPLE mode

SAMPLE PICK

Parameters:
In workstation identifier N
In pick device number (1..n) I
Out status (OK, NOPICK) E
Out segment name N
Out pick identifier N

Effect: If the current measure of the specified PICK device is indicating no pick, status NOPICK is returned; otherwise OK is returned together with a segment name and a pick identifier which are set according to the current measure of the PICK device. The pick identifier is associated with the primitive, within the segment, that was picked.

References:
4.8.1
4.8.3
4.8.4

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
37 Specified workstation is not of category OUTIN
140 Specified input device is not present on workstation
142 Input device is not in SAMPLE mode

SAMPLE STRING

Parameters:
In workstation identifier N
In string device number (1..n) I
Out character string S

WSOP, WSAC, SGOP L1c
WSOP, WSAC, SGOP Lmc

Page 137
Effect: The logical input value, which is the current measure of the specified STRING device, is returned.

NOTE The length of the returned string is less than or equal to the buffer size specified in the workstation state list entry (for this device) in the data record.

References:

4.8.1
4.8.3
4.8.4

Errors:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation
142 Input device is not in SAMPLE mode

5.7.5 Event Input Functions

AWAIT EVENT

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>In timeout (seconds)</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Out workstation identifier</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>Out input class</td>
<td>E</td>
<td>(NONE, LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)</td>
</tr>
<tr>
<td>Out logical input device n</td>
<td>I</td>
<td>(1..n)</td>
</tr>
</tbody>
</table>

Effect: If the input queue is empty, GKS is set into a wait state until an input event is written into the queue or the time specified in the timeout parameter has elapsed. If a timeout occurs and there is still no entry in the queue, a NONE value is returned for input class. If there is at least one entry in the queue, the oldest event report is moved from the event queue to the current event report in the GKS state list. The workstation identifier, the input class, and the logical input device number are returned and the corresponding values are made available for subsequent interrogation by the GET <input class> functions.

NOTE The operation is performed even if error 147 has occurred.

A timeout of zero causes an immediate inspection of the queue, and a NONE value for input class is returned if the queue is empty.

Some operating systems may not provide a reliable timeout facility. In this case a timeout different from zero may never cause a timeout at all.

References:

4.8.1
4.8.2
4.8.3
4.8.5

Errors:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
147 Input queue has overflowed
151 Timeout is invalid
**FLUSH DEVICE EVENTS**

Parameters:
- In workstation identifier \( N \)
- In input class (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING) \( E \)
- In logical input device number \( (1..n) \)

Effect: All entries in the input queue from the specified logical input device are removed.

NOTE: The operation is performed even if error 147 has occurred.

References:
- 4.8.5

Errors:
- 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20 Specified workstation identifier is invalid
- 25 Specified workstation is not open
- 38 Specified workstation is neither of category INPUT nor of category OUTIN
- 140 Specified input device is not present on workstation
- 147 Input queue has overflowed

**GET LOCATOR**

Parameters:
- Out normalization transformation number \( (0..n) \)
- Out locator position \( WC \)

Effect: The LOCATOR logical input value in the current event report is returned. This consists of a locator position in world coordinates and the normalization transformation number, which was used in the conversion to world coordinates.

References:
- 4.6.4
- 4.8.4
- 4.8.5

Errors:
- 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 150 No input value of the correct class is in the current event report

**GET STROKE**

Parameters:
- Out normalization transformation number \( (0..n) \)
- Out number of points \( (0..n) \)
- Out points in stroke \( WC )\n
Effect: The STROKE logical input value from the current event report is returned. This consists of a sequence of points in world coordinates, and the normalization transformation number, which was used in the conversion to world coordinates. The points in the stroke all lie within the window of the normalization transformation.
Input Functions

NOTE. The number of points in the stroke is less than or equal to the stroke buffer size specified in the workstation state list for this device.

References:
4.8.4
4.8.5

Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
150  No input value of the correct class is in the current event report

GET VALUATOR
Parameters:
Out value
Effect: The VALUATOR logical input value in the current event report is returned. The value delivered is in the range specified in the workstation state list entry (for the device) in the data record.

References:
4.8.4
4.8.5

Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
150  No input value of the correct class is in the current event report

GET CHOICE
Parameters:
Out status
Out choice number
Effect: The CHOICE logical input value in the current event report is returned. This consists of a CHOICE status and a choice number.

References:
4.8.4
4.8.5

Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
150  No input value of the correct class is in the current event report

GET PICK
Parameters:
Out status
Out segment name
Out pick identifier

Page 140
Effect: The PICK logical input value in the current event report is returned. This consists of a PICK status, a segment name and the pick identifier associated with the primitive within the segment that was picked.

References:
4.8.4
4.8.5

Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
150 No input value of the correct class is in the current event report

GET STRING

Parameters:  
Out  character string S

Effect: The STRING logical input value in the current event report is returned.

NOTE The length of the returned string is less than or equal to the buffer size specified in the workstation state list entry (for this device) in the data record, at the time the event was queued.

References:
4.8.4
4.8.5

Errors:
7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
150 No input value of the correct class is in the current event report
5.8 Metafile Functions

WRITE ITEM TO GKSM

Parameters:
- In workstation identifier
- In item type
- In item data record length
- In item data record

Effect: An item containing non-graphical data provided by the application program is written to the GKSM. The parameters 'item data record' and 'item data record length' define the data to be output whilst 'item type' specifies their type.

NOTE: This function can only be used to transfer non-graphical information to GKSM. Graphical data are sent automatically after a workstation of category MO has been activated.

Errors:
- 5 GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
- 20 Specified workstation identifier is invalid
- 30 Specified workstation is not active
- 32 Specified workstation is not of category MO
- 160 Item type is not allowed for user items
- 161 Item length is invalid

GET ITEM TYPE FROM GKSM

Parameters:
- In workstation identifier
- Out item type
- Out item data record length

Effect: GKS inspects the type of the current item and the length of its data record in the GKSM and returns the type and length back to the application program.

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

Parameters:
- In workstation identifier
- In maximum item data record length
- Out item data record

Effect: GKS returns the current item on the GKSM back to the application program and then makes the next item in the metafile the current item. If the item data record length is greater than 'maximum item data record length', the excess parts of the item are lost.

NOTE: By specifying 'maximum item data record length' = 0, the current item can be skipped.

Any program which makes use of the access that this function provides to the content of GKSM items is using information that is not part of the standard, viz. the format and content of metafile items.

References:

4.9

Errors:
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
84 Specified workstation is not of category MI
162 No item is left in GKS Metafile input
169 Metafile item is invalid
165 Content of item data record is invalid for the specified item type
166 Maximum item data record length is invalid

INTERPRET ITEM

Parameters:
- In item type
- In item data record length
- In item data record

Effect: The supplied item is interpreted. This causes appropriate changes in the set of GKS state variables (see 4.9) and the generation of appropriate graphical output, as determined by the metafile specification.

NOTE: Apart from errors noted below, other GKS errors may occur as a result of interpreting the item.

References:

4.9

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

5.9.1 Introduction

Inquiry functions return values from the various state lists. The data types of the values and the default values of the state list entries are summarized in section 6. Errors detected by inquiry functions are reported through an error indicator parameter, see 4.11.2. The error handling procedure is not called. The list of states in the function heading indicates those states in which the inquiry function can return valid values. Some inquiry functions that retrieve values from the workstation state lists have an input parameter of type 'Enumeration' that can take the following values:

- **a)** SET: the values returned are those provided by the application program.
- **b)** REALIZED: the values returned are those used by the workstation when the actual values are mapped to the available values in the workstation.

Inquiries for predefined representations in the workstation description table (see 5.9.6) have no such parameter unlike the corresponding inquiries for the representations in the workstation state list (see 5.9.5). The values of predefined representations are available on the workstation. Thus all values returned from a predefined representation are such that, if used by an application program to set a representation, a subsequent inquiry for that representation in the workstation state list would return the same values whether SET or REALIZED was specified.

5.9.2 Inquiry Function for Operating State Value

**INQUIRE OPERATING STATE VALUE**

Parameters:

- **Out** operating state value $(\text{GKCL}, \text{GKOP}, \text{WSOP}, \text{WSAC}, \text{SGOP})$ E

Effect: The operating state of GKS is returned.

References:

- 4.11.1
- 4.11.2

Errors:

- *none*
5.9.3 Inquiry Functions for GKS Description Table

INQUIRE LEVEL OF GKS

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out error indicator</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Out level of GKS</td>
<td>E</td>
<td>((ma, mb, mc, 0a, 0b, 0c, 1a, 1b, 1c, 2a, 2b, 2c))</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:

4.10
4.11.2

Errors:

none

INQUIRE LIST OF AVAILABLE WORKSTATION TYPES

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out error indicator</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Out number of available workstation types</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Out list of available workstation types</td>
<td>n x N</td>
<td></td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:

4.5.1
4.11.2

Errors:

none
INQUIRE WORKSTATION MAXIMUM NUMBERS

Parameters:

- Out error indicator I
- Out maximum number of simultaneously open workstations (1..n) I
- Out maximum number of simultaneously active workstations (1..n) I
- Out maximum number of workstations associated with segment (1..n) I

Effect: If the inquired information is available, the error indicator is returned as 0 and the values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:

4.5
4.11.2

Errors:

none

INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER

Parameters:

- Out error indicator I
- Out maximum normalization transformation number (1..n) I

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:

4.6.1
4.11.2

Errors:

none

5.9.4 Inquiry functions for GKS state list
INQUIRE SET OF OPEN WORKSTATIONS

**Parameters:**
- Out error indicator
- Out number of open workstations (0..n)
- Out set of open workstations

**Effect:**
- If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
- If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:
  - 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP

**References:**
- 4.5.2
- 4.11.2

**Errors:**
- none

---

INQUIRE SET OF ACTIVE WORKSTATIONS

**Parameters:**
- Out error indicator
- Out number of active workstations (0..n)
- Out set of active workstations

**Effect:**
- If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
- If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:
  - 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP

**References:**
- 4.5.2
- 4.11.2

**Errors:**
- none
INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES  

**Parameters:**

- Out error indicator
- Out current polyline index
- Out current polymarker index
- Out current text index
- Out current character height
- Out current character up vector
- Out current character width
- Out current character base vector
- Out current text path
- Out current text alignment
- Out current fill area index
- Out current pattern width vector
- Out current pattern height vector
- Out current pattern reference point

**Effect:** If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

**References:**

4.4.2

4.11.2

**Errors:**

none

---

INQUIRE CURRENT PICK IDENTIFIER VALUE  

**Parameters:**

- Out error indicator
- Out current pick identifier

**Effect:** If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

**References:**

4.4.2

4.11.2

**Errors:**

none
INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES

Parameters:
- Out error indicator
- Out current linetype \((-n..-1,1..n)\)
- Out current linewidth scale factor \(\geq 0\)
- Out current polyline colour index \((0..n)\)
- Out current marker type \((-n..-1,1..n)\)
- Out current marker size scale factor \(\geq 0\)
- Out current polymarker colour index \((0..n)\)
- Out current text font and precision \((-n..-1,1..n;\text{STRING,CHAR,STROKE})\)
- Out current character expansion factor \(>0\)
- Out current character spacing
- Out current text colour index \((0..n)\)
- Out current fill area interior style \((\text{HOLLOW, SOLID, PATTERN, HATCH})\)
- Out current fill area style index \((-n..-1,1..n)\)
- Out current fill area colour index \((0..n)\)
- Out current list of aspect source flags \((\text{BUNDLED, INDIVIDUAL})\)

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:
- 4.4.2
- 4.11.2

Errors:
- none

INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER

Parameters:
- Out error indicator
- Out current normalization transformation number \((0..n)\)
Inquiry Functions

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:
4.6.1
4.11.2

Errors:
none

INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS

GKOP, WSOP, WSAC, SGOP

Parameters:
Out error indicator I
Out list of transformation numbers n x 1

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters. The transformation numbers are returned in a list, which is ordered by viewport input priority, starting with the highest priority transformation number.
If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:
4.6.1
4.11.2

Errors:
none

INQUIRE NORMALIZATION TRANSFORMATION

GKOP, WSOP, WSAC, SGOP

Parameters:
In normalization transformation number (0..n) I
Out error indicator I
Out window limits WC 4 x R
Out viewport limits NDC 4 x R

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

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

50  Transformation number is invalid

References:

4.6.1
4.11.2

Errors:

none

**INQUIRE CLIPPING**

GKOP, WSOP, WSAC, SGOP  

Parameters:

- Out error indicator  
- Out clipping indicator  (CLIP, NOCLIP)  
- Out clipping rectangle  NDC  

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:

4.6.2
4.11.2

Errors:

none

**INQUIRE NAME OF OPEN SEGMENT**

SGOP  

Parameters:

- Out error indicator  
- Out name of open segment  

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:

4.6.2
4.11.2

Errors:

none
Inquiry Functions

References:

4.7.1
4.11.2

Errors:

none

INQUIRE SET OF SEGMENT NAMES IN USE

Parameters:

Out error indicator I
Out number of segment names (0..n) I
Out set of segment names in use n x N

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:

4.7.1
4.11.2

Errors:

none

INQUIRE MORE SIMULTANEOUS EVENTS

Parameters:

Out error indicator I
Out more simultaneous events (NOMORE, MORE) E

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to the following error number to indicate the reason for non-availability:

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

References:

4.8.5
4.11.2

Errors:

none
5.9.5 Inquiry functions for workstation state list

INQUIRE WORKSTATION CONNECTION AND TYPE  
WSOP, WSAC, SGOP  

Parameters:

- In  workstation identifier  N
- Out error indicator  I
- Out connection identifier  N
- Out workstation type  N

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

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

References:
- 4.5.2
- 4.11.2

Errors: none

INQUIRE WORKSTATION STATE  
WSOP, WSAC, SGOP  

Parameters:

- In  workstation identifier  N
- Out error indicator  I
- Out workstation state (INACTIVE, ACTIVE)  E

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20 Specified workstation identifier is invalid
- 25 Specified workstation is not open
- 33 Specified workstation is of category MI
- 35 Specified workstation is of category INPUT

References:
- 4.5.2
- 4.11.2

Errors: none
INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES

Parameters:

- **In** workstation identifier
- **Out** error indicator
- **Out** deferral mode: (ASAP, BNIG, BNIL, ASTI)
- **Out** implicit regeneration mode: (SUPPRESSED, ALLOWED)
- **Out** display surface empty: (EMPTY, NOTEMPTY)
- **Out** new frame action necessary at update: (NO, YES)

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20 Specified workstation identifier is invalid
- 25 Specified workstation is not open
- 33 Specified workstation is of category MI
- 35 Specified workstation is of category INPUT
- 36 Specified workstation is Workstation Independent Segment Storage

References:

4.5.3
4.5.4
4.11.2

Errors:

none

INQUIRE LIST OF POLYLINE INDICES

Parameters:

- **In** workstation identifier
- **Out** error indicator
- **Out** number of polyline bundle table entries: (5..n)
- **Out** list of defined polyline indices: (1..n)

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20 Specified workstation identifier is invalid
- 25 Specified workstation is not open
- 33 Specified workstation is of category MI
- 35 Specified workstation is of category INPUT
- 36 Specified workstation is Workstation Independent Segment Storage
INQUIRE POLYLINE REPRESENTATION

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>In, N</td>
</tr>
<tr>
<td>polyline index</td>
<td>In, (1..n)</td>
</tr>
<tr>
<td>type of returned values</td>
<td>In, (SET,REALIZED)</td>
</tr>
<tr>
<td>error indicator</td>
<td>Out, I</td>
</tr>
<tr>
<td>linetype</td>
<td>Out, (-n..-1,1..n)</td>
</tr>
<tr>
<td>linewidth scale factor</td>
<td>Out, ≥0</td>
</tr>
<tr>
<td>polyline colour index</td>
<td>Out, (0..n)</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the specified polyline index is not present in the polyline bundle table on the workstation and the specified type of returned values is REALIZED, the representation for polyline index 1 is returned.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
33 Specified workstation is of category MI
35 Specified workstation is of category INPUT
36 Specified workstation is Workstation Independent Segment Storage
60 Polyline index is invalid
61 A representation for the specified polyline index has not been defined on this workstation

References:

4.4.3
4.11.2

Errors:

none
INQUIRE LIST OF POLYMARKER INDICES

Parameters:
- In  workstation identifier  N
- Out error indicator  I
- Out number of polymarker bundle table entries  \((5..n)\)  I
- Out list of defined polymarker indices  \((1..n)\)  nX1

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20  Specified workstation identifier is invalid
- 25  Specified workstation is not open
- 33  Specified workstation is of category MI
- 35  Specified workstation is of category INPUT
- 36  Specified workstation is Workstation Independent Segment Storage

References:

- 4.4.4
- 4.11.2

Errors:
- none

INQUIRE POLYMARKER REPRESENTATION

Parameters:
- In  workstation identifier  N
- In  polymarker index  \((1..n)\)  I
- In  type of returned values  \((SET,REALIZED)\)  E
- Out error indicator  I
- Out marker type  \((-n..-1,1..n)\)  I
- Out marker size scale factor  \(\geq 0\)  R
- Out polymarker colour index  \((0..n)\)  I

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the specified polymarker index is not present in the polymarker bundle table on the workstation and the specified type of returned values is REALIZED, the representation for polymarker index 1 is returned.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20  Specified workstation identifier is invalid
- 25  Specified workstation is not open
- 33  Specified workstation is of category MI
- 35  Specified workstation is of category INPUT
- 36  Specified workstation is Workstation Independent Segment Storage
- 66  Polymarker index is invalid
67 A representation for the specified polymarker index has not been defined on this workstation.

References:
4.4.4
4.11.2

Errors:
none

**INQUIRE LIST OF TEXT INDICES**

Parameters:
- In workstation identifier
- Out error indicator
- Out number of text bundle table entries
- Out list of defined text indices

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters. If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:
- 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20 Specified workstation identifier is invalid
- 25 Specified workstation is not open
- 33 Specified workstation is of category MI
- 35 Specified workstation is of category INPUT
- 36 Specified workstation is Workstation Independent Segment Storage

References:
4.4.5
4.11.2

Errors:
none

**INQUIRE TEXT REPRESENTATION**

Parameters:
- In workstation identifier
- In text index
- In type of returned values
- Out error indicator
- Out text font and precision
- Out character expansion factor
- Out character spacing
- Out text colour index

References:
4.4.5
4.11.2

Errors:
none
Inquiry Functions

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the specified text index is not present in the text bundle table on the workstation and the specified type of returned values is REALIZED, the representation for text index 1 is returned.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
33 Specified workstation is of category MI
35 Specified workstation is of category INPUT
36 Specified workstation is Workstation Independent Segment Storage
72 Text index is invalid
73 A representation for the specified text index has not been defined on this workstation

References:
4.4.5
4.11.2

Errors:
none

INQUIRE TEXT EXTENT

Parameters:
In workstation identifier
In text position
In character string
Out error indicator
Out concatenation point
Out text extent parallelogram

WSOP, WSAC, SGOP Lma

N WC WC
P P
S I

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

The extent of the specified character string is computed using the text font and precision, character expansion factor and character spacing currently selected (either via the bundle or individually, depending upon the corresponding ASFs) and the current text attributes (CHARACTER HEIGHT, CHARACTER WIDTH, CHARACTER UP VECTOR, CHARACTER BASE VECTOR, TEXT PATH, TEXT ALIGNMENT). If the current text index is not present in the text bundle table, text index 1 is used.

At precisions STRING and CHAR, the text extent parallelogram is an approximation of that defined in 4.4.5, being the minimum which completely encloses the character bodies of the displayed string (see figure 16). For UP and DOWN text paths, the widest character body in the font is enclosed. The parallelogram is returned as four corner points in anticlockwise order. If, at STROKE precision, the CHARACTER WIDTH VECTOR and CHARACTER BASE VECTOR are perpendicular, the text extent parallelogram is a rectangle.
The concatenation point can be used as the origin of a subsequent TEXT output primitive for the concatenation of character strings, where meaningful. For certain combinations of TEXT PATH and TEXT ALIGNMENT, concatenation is not meaningful and the returned concatenation point is the same as the text position.

If TEXT PATH is RIGHT or LEFT, the concatenation point is displaced from the text position, in a direction determined by the horizontal component of TEXT ALIGNMENT. If this component is LEFT, the direction is to the right; if it is CENTRE, the displacement is zero; if it is RIGHT, the direction is to the left. Unless the horizontal component of TEXT ALIGNMENT is CENTRE, the magnitude of the displacement is the width of the text extent parallelogram plus one additional character spacing. (The width of the text extent parallelogram is the length of the sides parallel to the CHARACTER BASE VECTOR.)

If TEXT PATH is UP or DOWN, the concatenation point is displaced from the text position in a direction determined by the vertical component of TEXT ALIGNMENT. If this component is TOP or CAP, the direction is down; if it is HALF, the displacement is zero; if it is BASE or BOTTOM, the direction is up. Unless the vertical component of TEXT ALIGNMENT is HALF, the magnitude of the displacement is the height of the text extent parallelogram plus an additional character spacing. (The height of the text extent parallelogram is the length of the sides parallel to the CHARACTER UP VECTOR.)

Control characters in the character string have a workstation dependent effect consistent with their treatment in the TEXT function (see 5.3).
Figure 16. Examples of replies to INQUIRE TEXT EXTENT with different text attributes.
If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- **7 GKS not in proper state:** GKS shall be in one of the states WSOP, WSAC or SGOP
- **20 Specified workstation identifier is invalid**
- **25 Specified workstation is not open**
- **39 Specified workstation is neither of category OUTPUT nor of category OUTIN**
- **101 Invalid code in string**

References:

4.4.5
4.11.2

Errors:

*none*

**INQUIRE LIST OF FILL AREA INDICES**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>WSOP, WSAC, SGOP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In</strong> workstation identifier</td>
<td></td>
</tr>
<tr>
<td><strong>Out</strong> error indicator</td>
<td></td>
</tr>
<tr>
<td><strong>Out</strong> number of fill area bundle table entries</td>
<td>(5..n)</td>
</tr>
<tr>
<td><strong>Out</strong> list of defined fill area indices</td>
<td>(1..n)</td>
</tr>
</tbody>
</table>

**Effect:** If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- **7 GKS not in proper state:** GKS shall be in one of the states WSOP, WSAC or SGOP
- **20 Specified workstation identifier is invalid**
- **25 Specified workstation is not open**
- **33 Specified workstation is of category MI**
- **35 Specified workstation is of category INPUT**
- **36 Specified workstation is Workstation Independent Segment Storage**

References:

4.4.6
4.11.2

Errors:

*none*
INQUIRE FILL AREA REPRESENTATION

Parameters:

- In  workstation identifier
- In  fill area index
- In  type of returned values
- Out error indicator
- Out fill area interior style
- Out fill area style index
- Out fill area colour index

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the specified fill area index is not present in the fill area bundle table on the workstation and the specified type of returned values is REALIZED, the representation for fill area index 1 is returned.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

1. GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
2. Specified workstation identifier is invalid
3. Specified workstation is not open
4. Specified workstation is of category MI
5. Specified workstation is of category INPUT
6. Specified workstation is Workstation Independent Segment Storage
7. Fill area index is invalid
8. A representation for the specified fill area index has not been defined on this workstation

Errors:
- none

References:

4.4.6
4.11.2

---

INQUIRE LIST OF PATTERN INDICES

Parameters:

- In  workstation identifier
- Out error indicator
- Out number of pattern table entries
- Out list of pattern indices

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

7. GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
2. Specified workstation identifier is invalid
2. Specified workstation is not open
3. Specified workstation is of category MI
INQUIRE PATTERN REPRESENTATION

Parameters:

- **In** workstation identifier
- **In** pattern index: \((1..n)\)
- **In** type of returned values: \((\text{SET,REALIZED})\)
- **Out** error indicator
- **Out** pattern array dimensions: \((1..n)\)
- **Out** pattern array: \(n \times n \times 1\)

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the specified pattern index is not present in the pattern table on the workstation and the specified type of returned values is REALIZED, the representation for pattern index 1 is returned (pattern index 1 is present if interior style PATTERN is supported on the workstation).

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- **7** GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- **20** Specified workstation identifier is invalid
- **25** Specified workstation is not open
- **35** Specified workstation is of category MI
- **36** Specified workstation is of category INPUT
- **85** Specified pattern index is invalid
- **88** A representation for the specified pattern index has not been defined on this workstation
- **90** Interior style PATTERN is not supported on this workstation

References:

- **4.4.6**
- **4.11.2**

Errors:

- **none**
INQUIRE LIST OF COLOUR INDICES

Parameters:

- In  workstation identifier N
- Out error indicator I
- Out number of colour table entries (2..n) I
- Out list of colour indices (0..n) n x I

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20  Specified workstation identifier is invalid
25  Specified workstation is not open
33  Specified workstation is of category MI
35  Specified workstation is of category INPUT
36  Specified workstation is Workstation Independent Segment Storage

References:

4.4.9
4.11.2

Errors:

none

INQUIRE COLOUR REPRESENTATION

Parameters:

- In  workstation identifier N
- In  colour index (0..n) I
- In  type of returned values (SET,REALIZED) E
- Out error indicator I
- Out colour (red,green,blue intensities) [0,1] 3 x R

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the specified colour index is not present in the colour table on the workstation and the specified type of returned values is REALIZED, the representation of the workstation dependent colour index, that would be used if output primitives were displayed with the specified colour index, is returned.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20  Specified workstation identifier is invalid
25  Specified workstation is not open
33  Specified workstation is of category MI
35  Specified workstation is of category INPUT
36  Specified workstation is Workstation Independent Segment Storage
93  Colour index is invalid
94  A representation for the specified colour index has not been defined on this workstation
GKS Functions

References:
4.4.9
4.11.2

Errors:
none

INQUIRE WORKSTATION TRANSFORMATION

Parameters:
- In  workstation identifier
- Out error indicator
- Out workstation transformation update state (NOTPENDING, PENDING)
- Out requested workstation window
- Out current workstation window
- Out requested workstation viewport
- Out current workstation viewport

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

The workstation transformation update state is PENDING if a workstation transformation change has been requested but not yet provided.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
33 Specified workstation is of category MI
86 Specified workstation is Workstation Independent Segment Storage

References:
4.6.3
4.11.2

Errors:
none
INQUIRE SET OF SEGMENT NAMES ON WORKSTATION

Parameters:

- In  workstation identifier  
- Out  error indicator  
- Out  number of segment names  
- Out  set of stored segments for this workstation  

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20  Specified workstation identifier is invalid
- 25  Specified workstation is not open
- 33  Specified workstation is of category MI
- 35  Specified workstation is of category INPUT

References:

4.5.2
4.7.1
4.11.2

Errors:

none

INQUIRE LOCATOR DEVICE STATE

Parameters:

- In  workstation identifier  
- In  locator device number  
- In  type of returned values  
- Out  error indicator  
- Out  operating mode  
- Out  echo switch  
- Out  initial normalization transformation number  
- Out  initial locator position  
- Out  prompt and echo type  
- Out  echo area  
- Out  locator data record  

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7  GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20  Specified workstation identifier is invalid
- 25  Specified workstation is not open
- 38  Specified workstation is neither of category INPUT nor of category OUTIN
- 140  Specified input device is not present on workstation
INQUIRE STROKE DEVICE STATE

Parameters:

| In  | workstation identifier | N   |
| In  | stroke device number    | (1..n) | I   |
| In  | type of returned values | SET,REALIZED | E   |
| Out | error indicator         | I   |
| Out | operating mode          | (REQUEST,SAMPLE,EVENT) | E   |
| Out | echo switch             | (ECHO,NOECHO) | E   |
| Out | initial normalization transformation number | (0..n) | I   |
| Out | initial number of points | (0..n) | I   |
| Out | initial points in stroke | WC | n XP |
| Out | prompt and echo type    | (-n..-1,1..n) | I   |
| Out | echo area               | DC | 4 XR |
| Out | stroke data record      | D   |

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
88 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation

References:

4.8
4.11.2

Errors:

none
### INQUIRE VALUATOR DEVICE STATE

**Parameters:**

| In workstation identifier | N |
| In valuator device number | (1..n) |
| Out error indicator | I |
| Out operating mode | (REQUEST,SAMPLE,EVENT) |
| Out echo switch | (ECHO,NOECHO) |
| Out initial value | R |
| Out prompt and echo type | (-n..-1,1..n) |
| Out echo area | DC |
| Out valuator data record | D |

**Effect:** If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- **7 GKS not in proper state:** GKS shall be in one of the states WSOP, WSAC or SGOP
- **20 Specified workstation identifier is invalid**
- **25 Specified workstation is not open**
- **88 Specified workstation is neither of category INPUT nor of category OUTIN**
- **140 Specified input device is not present on workstation**

**References:**

4.8
4.11.2

**Errors:**

*none*

### INQUIRE CHOICE DEVICE STATE

**Parameters:**

| In workstation identifier | N |
| In choice device number | (1..n) |
| Out error indicator | I |
| Out operating mode | (REQUEST,SAMPLE,EVENT) |
| Out echo switch | (ECHO,NOECHO) |
| Out initial status | (OK,NOCHOICE) |
| Out initial choice number | (1..n) |
| Out prompt and echo type | (-n..-1,1..n) |
| Out echo area | DC |
| Out choice data record | D |

**Effect:** If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- **7 GKS not in proper state:** GKS shall be in one of the states WSOP, WSAC or SGOP
- **20 Specified workstation identifier is invalid**
- **25 Specified workstation is not open**
- **88 Specified workstation is neither of category INPUT nor of category OUTIN**
GKS Functions

GKS Functions

140 Specified input device is not present on workstation

References:
4.8
4.11.2

Errors:
none

INQUIRE PICK DEVICE STATE

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>error indicator</td>
</tr>
<tr>
<td>pick device number</td>
<td>operating mode</td>
</tr>
<tr>
<td>type of returned values</td>
<td>initial status</td>
</tr>
<tr>
<td></td>
<td>initial segment</td>
</tr>
<tr>
<td></td>
<td>initial pick identifier</td>
</tr>
<tr>
<td></td>
<td>prompt and echo type</td>
</tr>
<tr>
<td></td>
<td>echo area</td>
</tr>
<tr>
<td></td>
<td>pick data record</td>
</tr>
</tbody>
</table>

Values:

<table>
<thead>
<tr>
<th>In parameter</th>
<th>Output parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation identifier</td>
<td>N</td>
</tr>
<tr>
<td>pick device number</td>
<td>I</td>
</tr>
<tr>
<td>type of returned values</td>
<td>E</td>
</tr>
<tr>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>operating mode</td>
<td>E</td>
</tr>
<tr>
<td>initial status</td>
<td>N</td>
</tr>
<tr>
<td>initial segment</td>
<td>N</td>
</tr>
<tr>
<td>initial pick identifier</td>
<td>I</td>
</tr>
<tr>
<td>prompt and echo type</td>
<td>(-n..-1,1..n)</td>
</tr>
<tr>
<td>echo area</td>
<td>DC</td>
</tr>
<tr>
<td>pick data record</td>
<td>D</td>
</tr>
</tbody>
</table>

Effect:
If the inquired information is available, the error indicator is returned as 0 and values
are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters
are implementation dependent and the error indicator is set to one of the following
error numbers to indicate the reason for non-availability:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
37 Specified workstation is not of category OUTIN
140 Specified input device is not present on workstation

References:
4.8
4.11.2

Errors:
none
INQUIRE STRING DEVICE STATE

Parameters:
- In workstation identifier N
- In string device number (1..n) I
- Out error indicator I
- Out operating mode (REQUEST, SAMPLE, EVENT) E
- Out echo switch (ECHO, NOECHO) E
- Out initial string S
- Out prompt and echo type (-n..-1, 1..n) I
- Out echo area DC 4 XR
- Out string data record D

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20 Specified workstation identifier is invalid
- 25 Specified workstation is not open
- 38 Specified workstation is neither of category INPUT nor of category OUTIN
- 140 Specified input device is not present on workstation

References:
4.8
4.11.2

Errors:
none

5.9.6 Inquiry functions for workstation description table

INQUIRE WORKSTATION CATEGORY

Parameters:
- In workstation type N
- Out error indicator I
- Out workstation category (OUTPUT, INPUT, OUTIN, WISS, MO, MI) E

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 22 Specified workstation type is invalid
- 23 Specified workstation type does not exist

References:
4.5.1
4.11.2
Errors: none

INQUIRE WORKSTATION CLASSIFICATION   GKOP, WSOP, WSAC, SGOP   L0a
Parameters:
  In  workstation type  N
  Out error indicator  I
  Out vector, raster, or other type  (VECTOR, RASTER, OTHER)  E
Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:
  8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
  22 Specified workstation type is invalid
  23 Specified workstation type does not exist
  39 Specified workstation is neither of category OUTPUT nor of category OUTIN
References:
4.5.1
4.11.2
Errors: none

INQUIRE DISPLAY SPACE SIZE   GKOP, WSOP, WSAC, SGOP   Lma
Parameters:
  In  workstation type  N
  Out error indicator  I
  Out device coordinate units  (METRES, OTHER)  E
  Out maximum display surface size in device coordinate units  DC > 0  2xR
  Out maximum display surface size in raster units  (1..n)  2xR
Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:
  8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
  22 Specified workstation type is invalid
  23 Specified workstation type does not exist
  31 Specified workstation is of category MO
  33 Specified workstation is of category MI
  36 Specified workstation is Workstation Independent Segment Storage
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES

**Parameters:**

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
<th>Parameters</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In</td>
<td>workstation type</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Out</td>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>Out</td>
<td>Out</td>
<td>polyline bundle representation</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>changeable</td>
<td>E</td>
</tr>
<tr>
<td>Out</td>
<td>Out</td>
<td>polymarker bundle representation</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>changeable</td>
<td>E</td>
</tr>
<tr>
<td>Out</td>
<td>Out</td>
<td>text bundle representation</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>changeable</td>
<td>E</td>
</tr>
<tr>
<td>Out</td>
<td>Out</td>
<td>fill area bundle representation</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>changeable</td>
<td>E</td>
</tr>
<tr>
<td>Out</td>
<td>Out</td>
<td>pattern representation</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>changeable</td>
<td>E</td>
</tr>
<tr>
<td>Out</td>
<td>Out</td>
<td>colour representation</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>changeable</td>
<td>E</td>
</tr>
<tr>
<td>Out</td>
<td>Out</td>
<td>workstation transformation</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>changeable</td>
<td>E</td>
</tr>
</tbody>
</table>

**Effect:** If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

IRG means that implicit regeneration is necessary; IMM means the action is performed immediately.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

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

22 Specified workstation type is invalid

23 Specified workstation type does not exist

39 Specified workstation is neither of category OUTPUT nor of category OUTIN

References:

4.5.1
4.5.3
4.11.2

Errors:

none
INQUIRE DEFAULT DEFERRAL STATE VALUES  GKOP, WSOP, WSAC, SGOP  

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>workstation type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out</td>
<td>error indicator</td>
</tr>
<tr>
<td>Out</td>
<td>default value for deferral mode (ASAP,BNIG,BNIL,ASTI)</td>
</tr>
<tr>
<td>Out</td>
<td>default value for implicit regeneration mode (SUPPRESSED, ALLOWED)</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 22 Specified workstation type is invalid
- 28 Specified workstation type does not exist
- 89 Specified workstation is neither of category OUTPUT nor of category OUTIN

References:

- 4.5.1
- 4.5.3
- 4.11.2

Errors:

- none

INQUIRE POLYLINE FACILITIES  GKOP, WSOP, WSAC, SGOP  

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>workstation type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out</td>
<td>error indicator</td>
</tr>
<tr>
<td>Out</td>
<td>number of available linetypes</td>
</tr>
<tr>
<td>Out</td>
<td>list of available linetypes</td>
</tr>
<tr>
<td>Out</td>
<td>number of available linewidths</td>
</tr>
<tr>
<td>Out</td>
<td>nominal linewidth</td>
</tr>
<tr>
<td>Out</td>
<td>range of linewidths (minimum, maximum)</td>
</tr>
<tr>
<td>Out</td>
<td>number of predefined polyline indices</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the number of available linewidths is returned as 0, the workstation supports a continuous range of linewidths.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 22 Specified workstation type is invalid
- 28 Specified workstation type does not exist
- 89 Specified workstation is neither of category OUTPUT nor of category OUTIN
INQUIRE PREDEFINED POLYLINE REPRESENTATION

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>In workstation type</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>In predefined polyline index</td>
<td>(1..n)</td>
<td>I</td>
</tr>
<tr>
<td>Out error indicator</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Out linetype</td>
<td>(-n..-1,1..n)</td>
<td>I</td>
</tr>
<tr>
<td>Out linewidth scale factor</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Out polyline colour index</td>
<td>(0..n)</td>
<td>I</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 22 Specified workstation type is invalid
- 23 Specified workstation type does not exist
- 89 Specified workstation is neither of category OUTPUT nor of category OUTIN
- 60 Polyl ine index is invalid
- 62 A representation for the specified polyline index has not been predefined on this workstation

References:

4.5.1
4.11.2

Errors:
none
INQUIRE POLYMARKER FACILITIES

Parameters:

<table>
<thead>
<tr>
<th>In workstation type</th>
<th>Out error indicator</th>
<th>Out number of available marker types</th>
<th>Out list of available marker types</th>
<th>Out number of available marker sizes</th>
<th>Out nominal marker size</th>
<th>Out range of marker sizes (minimum, maximum)</th>
<th>Out number of predefined polymarker indices</th>
</tr>
</thead>
<tbody>
<tr>
<td>In N</td>
<td>Out I</td>
<td>Out (5..n)</td>
<td>Out (-n..-1, 1..n)</td>
<td>Out (0..n)</td>
<td>Out DC &gt;0</td>
<td>Out DC &gt;0 x2</td>
<td>Out (0.5..n)</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the number of available marker sizes is returned as 0, the workstation supports a continuous range of marker sizes.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

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

22 Specified workstation type is invalid

23 Specified workstation type does not exist

89 Specified workstation is neither of category OUTPUT nor of category OUTIN

References:

4.5.1

4.11.2

Errors:

none

INQUIRE PREDEFINED POLYMARKER REPRESENTATION

Parameters:

<table>
<thead>
<tr>
<th>In workstation type</th>
<th>In predefined polymarker index</th>
<th>Out error indicator</th>
<th>Out marker type</th>
<th>Out marker size scale factor</th>
<th>Out polymarker colour index</th>
</tr>
</thead>
<tbody>
<tr>
<td>In N</td>
<td>Out (1..n)</td>
<td>Out I</td>
<td>Out (-n..-1, 1..n)</td>
<td>Out R</td>
<td>Out (0..n)</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

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

22 Specified workstation type is invalid

23 Specified workstation type does not exist

89 Specified workstation is neither of category OUTPUT nor of category OUTIN

Errors:

none
66 Polymarker index is invalid
68 A representation for the specified polymarker index has not been predefined on this workstation

References:
4.5.1
4.11.2

Errors:
none

INQUIRE TEXT FACILITIES

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
<th>Out</th>
<th>Out</th>
<th>Out</th>
<th>Out</th>
<th>Out</th>
<th>Out</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation type</td>
<td>error indicator</td>
<td>number of text font and precision pairs</td>
<td>list of text font and precision pairs</td>
<td>number of available character heights</td>
<td>range of character heights (minimum, maximum)</td>
<td>number of available character expansion factors</td>
<td>range of character expansion factors (minimum, maximum)</td>
<td>number of predefined text indices</td>
</tr>
<tr>
<td>N</td>
<td>I</td>
<td>(1..n)</td>
<td>(-n..-1,1..n;STRING,CHAR,STROKE)</td>
<td>(0..n)</td>
<td>DC &gt;0</td>
<td>(0..n)</td>
<td>&gt;0</td>
<td>(0,2..n)</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the number of available character heights is returned as 0, the workstation supports a continuous range of character heights. If the number of available character expansion factors is returned as 0, the workstation supports a continuous range of character expansion factors. If the available character heights and character expansion factors vary between fonts, the character heights and character expansion factors returned are for font 1.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
89 Specified workstation is neither of category OUTPUT nor of category OUTIN

References:
4.5.1
4.11.2

Errors:
none
**INQUIRE PREDEFINED TEXT REPRESENTATION**

**Parameters:**

- **In workstation type**  
- **In predefined text index** (1..n)  
- **Out error indicator**  
- **Out text font and precision** (-n..-1,1..n;STRING,CHAR,STROKE) (I;E)  
- **Out character expansion factor** >0  
- **Out character spacing**  
- **Out text colour index** (0..n)

**Effect:** If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 22 Specified workstation type is invalid
- 23 Specified workstation type does not exist
- 39 Specified workstation is neither of category OUTPUT nor of category OUTIN

**References:**

4.5.1
4.11.2

**Errors:**

none

---

**INQUIRE FILL AREA FACILITIES**

**Parameters:**

- **In workstation type**  
- **Out error indicator**  
- **Out number of available fill area interior styles** (1..n)  
- **Out list of available fill area interior styles** (HOLLOW,SOLID,PATTERN,HATCH) nXE  
- **Out number of available hatch styles** (0..n)  
- **Out list of available hatch styles** (-n..-1,1..n) nXd  
- **Out number of predefined fill area indices** (0,5..n)

**Effect:** If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 22 Specified workstation type is invalid
- 23 Specified workstation type does not exist
- 39 Specified workstation is neither of category OUTPUT nor of category OUTIN
INQUIRE PREDEFINED FILL AREA REPRESENTATION

GKOP, WSOP, WSAC, SGOP

Parameters:

| In workstation type                        | N  |
| In predefined fill area index              | (1..n) | I |
| Out error indicator                        | I  |
| Out fill area interior style               | (HOLLOW, SOLID, PATTERN, HATCH) | E |
| Out fill area style index                  | (-n..-1, 1..n) | I |
| Out fill area colour index                 | (0..n) | I |

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

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

References:

4.5.1
4.11.2

Errors:

none
INQUIRE PATTERN FACILITIES

Parameters:

In  workstation type  
Out error indicator  
Out number of predefined pattern indices

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8  GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
89 Specified workstation is neither of category OUTPUT nor of category OUTIN

References:

4.5.1
4.11.2

Errors:

none

INQUIRE PREDEFINED PATTERN REPRESENTATION

Parameters:

In  workstation type  
In predefined pattern index  
Out error indicator  
Out pattern array dimensions  
Out pattern array

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8  GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
89 Specified workstation is neither of category OUTPUT nor of category OUTIN
85 Specified pattern index is invalid
89 A representation for the specified pattern index has not been predefined on this workstation
90 Interior style PATTERN is not supported on this workstation
Inquiry Functions

References:

4.5.1
4.11.2

Errors:

none

**INQUIRE COLOUR FACILITIES**

Parameters:

<table>
<thead>
<tr>
<th></th>
<th>workstation type</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>Out</td>
<td>number of available colours or intensities</td>
<td>(0,2..n)</td>
</tr>
<tr>
<td>Out</td>
<td>colour available</td>
<td>(COLOUR,MONOCHROME)</td>
</tr>
<tr>
<td>Out</td>
<td>number of predefined colour indices</td>
<td>(2..n)</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the number of available colours or intensities is returned as 0, the workstation supports a continuous range of colours or intensities.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
39 Specified workstation is neither of category OUTPUT nor of category OUTIN

References:

4.5.1
4.11.2

Errors:

none

**INQUIRE PREDEFINED COLOUR REPRESENTATION**

Parameters:

<table>
<thead>
<tr>
<th></th>
<th>workstation type</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>predefined colour index</td>
<td>(0..n)</td>
</tr>
<tr>
<td>Out</td>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>Out</td>
<td>colour (red, green, blue intensities)</td>
<td>[0,1]</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:
GKS Functions

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
39 Specified workstation is neither of category OUTPUT nor of category OUTIN
95 A representation for the specified colour index has not been predefined on this workstation

References:
4.5.1
4.11.2

Errors:
none

INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES
GKOP, WSOP, WSAC, SGOP

Parameters:

<table>
<thead>
<tr>
<th>Parameter Description</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation type</td>
<td>N</td>
</tr>
<tr>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>number of available generalized drawing primitives</td>
<td>(0..n)</td>
</tr>
<tr>
<td>list of GDP identifiers</td>
<td>n XN</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
39 Specified workstation is neither of category OUTPUT nor of category OUTIN

References:
4.5.1
4.11.2

Errors:
none
INQUIRE GENERALIZED DRAWING PRIMITIVE

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>workstation type</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>GDP identifier</td>
<td>N</td>
</tr>
<tr>
<td>Out</td>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>Out</td>
<td>number of sets of attributes used</td>
<td>(0..n)</td>
</tr>
<tr>
<td>Out</td>
<td>list of sets of attributes used</td>
<td>n×E</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 22 Specified workstation type is invalid
- 23 Specified workstation type does not exist
- 39 Specified workstation is neither of category OUTPUT nor of category OUTIN
- 41 Specified workstation type is not able to generate the specified generalized drawing primitive

References:

4.5.1
4.11.2

Errors:
none

INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>workstation type</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out</td>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>Out</td>
<td>maximum number of polyline bundle table entries</td>
<td>(0,5..n)</td>
</tr>
<tr>
<td>Out</td>
<td>maximum number of polymarker bundle table entries</td>
<td>(0,5..n)</td>
</tr>
<tr>
<td>Out</td>
<td>maximum number of text bundle table entries</td>
<td>(0,2..n)</td>
</tr>
<tr>
<td>Out</td>
<td>maximum number of fill area bundle table entries</td>
<td>(0,5..n)</td>
</tr>
<tr>
<td>Out</td>
<td>maximum number of pattern indices</td>
<td>(0..n)</td>
</tr>
<tr>
<td>Out</td>
<td>maximum number of colour indices</td>
<td>(2..n)</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
- 22 Specified workstation type is invalid
- 23 Specified workstation type does not exist
- 39 Specified workstation is neither of category OUTPUT nor of category OUTIN
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED

GKOP, WSOP, WSAC, SGOP

Parameters:

In  workstation type     N
Out error indicator      I
Out number of segment priorities supported (0..n) I

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the number of segment priorities supported is returned as 0, the workstation supports an infinite number of segment priorities.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8  GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
28 Specified workstation type does not exist
88 Specified workstation is neither of category OUTPUT nor of category OUTIN

References:

4.5.1
4.7.2
4.11.2

Errors:

none
INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES
GKOP, WSOP, WSAC, SGOP

Parameters:

In  workstation type  N
Out error indicator  I
Out segment transformation changeable (IRG, IMM) E
Out visibility changeable from 'visible' to 'invisible' (IRG, IMM) E
Out visibility changeable from 'invisible' to 'visible' (IRG, IMM) E
Out highlighting changeable (IRG, IMM) E
Out segment priority changeable (IRG, IMM) E
Out adding primitives to the open segment (IRG, IMM) E
Out segment deletion immediately visible (IRG, IMM) E

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

IRG means that implicit regeneration is necessary; IMM means the action is performed immediately.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8  GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22  Specified workstation type is invalid
28  Specified workstation type does not exist
89  Specified workstation is neither of category OUTPUT nor of category OUTIN

References:

4.5.1
4.5.3
4.11.2

Errors:
none

INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES
GKOP, WSOP, WSAC, SGOP

Parameters:

In  workstation type  N
Out error indicator  I
Out number of locator devices (0..n) I
Out number of stroke devices (0..n) I
Out number of valuator devices (0..n) I
Out number of choice devices (0..n) I
Out number of pick devices (0..n) I
Out number of string devices (0..n) I

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:
GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
38 Specified workstation is neither of category INPUT nor of category OUTIN

References:
4.5.1
4.8.1
4.11.2

Errors:
none

INQUIRE DEFAULT LOCATOR DEVICE DATA  GKOP, WSOP, WSAC, SGOP  Lmb

Parameters:

- In workstation type
- In logical input device number
- Out error indicator
- Out default initial locator position
- Out number of available prompt and echo types
- Out list of available prompt and echo types
- Out default echo area
- Out default locator data record

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation

References:
4.5.1
4.8.1
4.8.6
4.11.2

Errors:
none
INQUIRE DEFAULT STROKE DEVICE DATA  GKOP, WSOP, WSAC, SGOP  Lmb

Parameters:

In  workstation type  N
In  logical input device number  (1..n)  I
Out  error indicator  I
Out  maximum input buffer size  (64..n)  I
Out  number of available prompt and echo types  (1..n)  I
Out  list of available prompt and echo types  (-n..-1,1..n)  n Xl
Out  default echo area  DC  4 Xr
Out  default stroke data record  D

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist
88 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation

References:

4.5.1
4.8.1
4.8.6
4.11.2

Errors:

none

INQUIRE DEFAULT VALUATOR DEVICE DATA  GKOP, WSOP, WSAC, SGOP  Lmb

Parameters:

In  workstation type  N
In  logical input device number  (1..n)  I
Out  error indicator  I
Out  default initial value  R
Out  number of available prompt and echo types  (1..n)  I
Out  list of available prompt and echo types  (-n..-1,1..n)  n Xl
Out  default echo area  DC  4 Xr
Out  default valuator data record  D

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
23 Specified workstation type does not exist

none
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation

References:
4.5.1
4.8.1
4.8.6
4.11.2

Errors:
none

INQUIRE DEFAULT CHOICE DEVICE DATA

Parameters:

<table>
<thead>
<tr>
<th>In</th>
<th>workstation type</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>logical input device number</td>
<td>(1..n)</td>
</tr>
<tr>
<td>Out</td>
<td>error indicator</td>
<td></td>
</tr>
<tr>
<td>Out</td>
<td>maximum number of choice alternatives</td>
<td>(1..n)</td>
</tr>
<tr>
<td>Out</td>
<td>number of available prompt and echo types</td>
<td>(1..n)</td>
</tr>
<tr>
<td>Out</td>
<td>list of available prompt and echo types</td>
<td>(-n..-1,1..n)</td>
</tr>
<tr>
<td>Out</td>
<td>default echo area</td>
<td>DC</td>
</tr>
<tr>
<td>Out</td>
<td>default choice data record</td>
<td>D</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

Errors: If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
22 Specified workstation type is invalid
29 Specified workstation type does not exist
38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation

References:
4.5.1
4.8.1
4.8.6
4.11.2

Errors:
none
### INQUIRE DEFAULT PICK DEVICE DATA

**Parameters:**
- **In** workstation type
- **In** logical input device number \((1..n)\)
- **Out** error indicator \(I\)
- **Out** number of available prompt and echo types \((1..n)\)
- **Out** list of available prompt and echo types \((-n..-l,l..n)\)
- **Out** default echo area \(4XR\)
- **Out** default pick data record \(D\)

**Effect:**
- If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
- If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:
  - 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
  - 22 Specified workstation type is invalid
  - 23 Specified workstation type does not exist
  - 38 Specified workstation is neither of category INPUT nor of category OUTIN
  - 140 Specified input device is not present on workstation

**References:**
- 4.5.1
- 4.8.1
- 4.8.6
- 4.11.2

**Errors:**
- none

### INQUIRE DEFAULT STRING DEVICE DATA

**Parameters:**
- **In** workstation type
- **In** logical input device number \((1..n)\)
- **Out** error indicator \(I\)
- **Out** maximum string buffer size \((72..n)\)
- **Out** number of available prompt and echo types \((1..n)\)
- **Out** list of available prompt and echo types \((-n..-l,l..n)\)
- **Out** default echo area \(4XR\)
- **Out** default string data record \(D\)

**Effect:**
- If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.
- If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:
  - 8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP
  - 22 Specified workstation type is invalid
  - 23 Specified workstation type does not exist
  - 38 Specified workstation is neither of category INPUT nor of category OUTIN
140 Specified input device is not present on workstation

References:
4.5.1
4.8.1
4.8.6
4.11.2

Errors:
none

5.9.7 Inquiry functions for segment state list

INQUIRE SET OF ASSOCIATED WORKSTATIONS WSOP,WSAC,SGOP L1a

Parameters:

<table>
<thead>
<tr>
<th>Par.</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>segment name</td>
<td>N</td>
</tr>
<tr>
<td>Out</td>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>Out</td>
<td>number of associated workstations</td>
<td>(1..n)</td>
</tr>
<tr>
<td>Out</td>
<td>set of associated workstation identifiers</td>
<td>n x N</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

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

References:
4.7
4.11.2

Errors:
none

INQUIRE SEGMENT ATTRIBUTES WSOP,WSAC,SGOP L1a

Parameters:

<table>
<thead>
<tr>
<th>Par.</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>In</td>
<td>segment name</td>
<td>N</td>
</tr>
<tr>
<td>Out</td>
<td>error indicator</td>
<td>I</td>
</tr>
<tr>
<td>Out</td>
<td>segment transformation matrix</td>
<td>2 x 3 x R</td>
</tr>
<tr>
<td>Out</td>
<td>visibility (VISIBLE,INVISIBLE)</td>
<td>E</td>
</tr>
<tr>
<td>Out</td>
<td>highlighting (NORMAL,HIGHLIGHTED)</td>
<td>E</td>
</tr>
<tr>
<td>Out</td>
<td>segment priority [0,1]</td>
<td>R</td>
</tr>
<tr>
<td>Out</td>
<td>detectability (UNDETECTABLE,DETECTABLE)</td>
<td>E</td>
</tr>
</tbody>
</table>

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:
Inquiry Functions

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

References:
4.7
4.11.2

Errors:
one

5.9.8 Pixel inquiries

INQUIRE PIXEL ARRAY DIMENSIONS

Parameters:

\[
\begin{array}{ll}
\text{In} & \text{workstation identifier} \\
\text{In} & \text{2 points P,Q} \\
\text{Out} & \text{error indicator} \\
\text{Out} & \text{dimensions of pixel array}
\end{array}
\]

Parameters:
\[
\begin{array}{ll}
\text{WSOP, WSAC, SGOP} & \text{LOa} \\
\text{WS} & \text{N} \\
\text{WC} & \text{2XP} \\
\text{I} & \text{1} \\
\text{(1..n)} & \text{2X1}
\end{array}
\]

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

The points P, Q define a rectangle. By transforming P and Q by the current normalization and workstation transformations, the rectangle is mapped onto the display surface. The number of columns and the number of rows of pixels, whose positions lie within the rectangle, are returned. For this calculation no clipping is applied.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
20 Specified workstation identifier is invalid
25 Specified workstation is not open
39 Specified workstation is neither of category OUTPUT nor of category OUT1N

References:
4.11.2

Errors:
one
INQUIRE PIXEL ARRAY

Parameters:

- In workstation identifier
- In point P
- In dimensions of colour index array DX, DY
- Out error indicator
- Out presence of invalid values
- Out colour index array

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

By transforming P by the current normalization and workstation transformations, it is mapped onto a pixel of the display surface. The colour indices of the array of pixels, whose upper left corner is this pixel (associated with the (1,1) element), are returned in the colour index array. The orientation of the array is such that the first dimension increases as the X device coordinate increases and the second dimension increases as the Y device coordinate decreases. If the colour index corresponding to a particular pixel cannot be ascertained (for example, the point P was transformed such that the position of the pixel is not on the display surface), the value -1 (i.e. invalid) is assigned for that cell.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- 7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- 20 Specified workstation identifier is invalid
- 25 Specified workstation is not open
- 89 Specified workstation is neither of category OUTPUT nor of category OUTIN
- 40 Specified workstation has no pixel store readback capability
- 91 Dimensions of colour array are invalid

References:
4.11.2

Errors:
none

INQUIRE PIXEL

Parameters:

- In workstation identifier
- In point P
- Out error indicator
- Out colour index

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

By transforming P by the current normalization and workstation transformations, it is mapped onto a pixel of the display surface. The colour index of this pixel is returned. If a colour index cannot be ascertained (for example, the point P was transformed such that the position of the pixel is not on the display surface), the value -1 (i.e. invalid) is returned.
If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- **7** GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- **20** Specified workstation identifier is invalid
- **25** Specified workstation is not open
- **39** Specified workstation is neither of category OUTPUT nor of category OUTIN
- **40** Specified workstation has no pixel store readback capability

References:

4.11.2

Errors:

none

5.9.9 Inquiry function for GKS error state list

**INQUIRE INPUT QUEUE OVERFLOW**

Parameters:

- Out error indicator
- Out workstation identifier
- Out input class (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
- Out logical input device number (1..n)

Effect: If the inquired information is available, the error indicator is returned as 0 and values are returned in the output parameters.

If the input queue has overflowed since OPEN GKS or the last invocation of INQUIRE INPUT QUEUE OVERFLOW, the identification of the logical input device that caused the overflow is returned. The entry is removed from the error state list.

If the inquired information is not available, the values returned in the output parameters are implementation dependent and the error indicator is set to one of the following error numbers to indicate the reason for non-availability:

- **7** GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
- **148** Input queue has not overflowed since GKS was opened or the last invocation of INQUIRE INPUT QUEUE OVERFLOW
- **149** Input queue has overflowed, but associated workstation has been closed

References:

4.8.5
4.11.2

Errors:

none
5.10 Utility Functions

EVALUATE TRANSFORMATION MATRIX GKOP, WSOP, WSAC, SGOP L1a

Parameters:

- In fixed point WC or NDC P
- In shift vector WC or NDC 2xR
- In rotation angle in radians (positive if anticlockwise) R
- In scale factors 2xR
- In coordinate switch (WC,NDC) E
- Out segment transformation matrix 2x3xR

Effect: The transformation defined by fixed point, shift vector, rotation angle, and scale factors is evaluated and the result is put in the output segment transformation matrix (for use, for example, by INSERT SEGMENT and SET SEGMENT TRANSFORMATION). The coordinate switch determines whether the shift vector and fixed point are given in WC coordinates or NDC coordinates. If WC coordinates are used, the shift vector and the fixed point are transformed by the current normalization transformation. The order of transformation is: scale, rotate (both relative to the specified fixed point), and shift. The elements $M_{13}$ and $M_{23}$ of the resulting 2x3 transformation matrix are in NDC coordinates; the other elements are unitless.

References:

4.7.3

Errors:

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

ACCUMULATE TRANSFORMATION MATRIX GKOP, WSOP, WSAC, SGOP L1a

Parameters:

- In segment transformation matrix 2x3xR
- In fixed point WC or NDC P
- In shift vector WC or NDC 2xR
- In rotation angle in radians (positive if anticlockwise) R
- In scale factors 2xR
- In coordinate switch (WC,NDC) E
- Out segment transformation matrix 2x3xR

Effect: The transformation defined by fixed point, shift vector, rotation angle, and scale factors is composed with the input segment transformation matrix and the result is returned in the output segment transformation matrix (for use, for example, by INSERT SEGMENT and SET SEGMENT TRANSFORMATION). The coordinate switch determines whether the shift vector and fixed point are given in WC coordinates or NDC coordinates. If WC coordinates are used, the shift vector and the fixed point are transformed by the current normalization transformation. The order of transformation is: specified input matrix, scale, rotate (both relative to the specified fixed point), and shift. The elements $M_{13}$ and $M_{23}$ of the 2x3 input matrix and the resulting 2x3 transformation matrix are in NDC coordinates; the other elements are unitless.

References:

4.7.3
Errors:

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

**EMERGENCY CLOSE GKS**

Parameters:

none

Effect: GKS is emergency closed (see 4.12). The following actions are performed (if possible):

a) CLOSE SEGMENT (if open);

b) UPDATE for all open workstations;

c) DEACTIVATE all active workstations;

d) CLOSE all open workstations;

e) CLOSE GKS.

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

References:

4.12

Errors:

none

---

**ERROR HANDLING**

Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>error number as listed in section 5.</td>
</tr>
<tr>
<td>N</td>
<td>identification of the GKS procedure called by the application program which caused the error detection</td>
</tr>
<tr>
<td>F</td>
<td>error file</td>
</tr>
</tbody>
</table>

NOTE: The last parameter has been defined in OPEN GKS.

Effect: The ERROR HANDLING procedure is called by GKS in any of the error situations listed in section 5. The standard procedure just calls the ERROR LOGGING procedure with the same parameters.

NOTE: The ERROR HANDLING procedure may be replaced by an application program supplied procedure to allow specific reaction to some error situations.

References:

4.12

Errors:

none
ERROR LOGGING

Parameters:
- In error number as listed in section 5.
- In identification of the GKS procedure called by the application program which caused the error detection.
- In error file.

NOTE
The last parameter has been defined in OPEN GKS.

Effect
The ERROR LOGGING procedure:

a) prints an error message and GKS function identification on the error file,

b) returns to the calling procedure.

References:

4.12

Errors:

none
6 GKS data structures

6.1 Notation and data types
In this section, the contents of the GKS data structures are listed.

The information for each entry includes:

a) the name of the entry;
b) the coordinate system (if appropriate);
c) the permitted values;
d) the data type;
e) the initial value (if appropriate).

The notation used to express the data type, coordinate system and permitted values is also used to describe the parameters of the GKS functions in section 5.

The data type can be a simple type, which is one of the following:

- **I** integer: whole number
- **R** real: floating point number
- **S** string: number of characters and character sequence
- **P** point: 2 real values specifying the x- and y-coordinates of a location in WC, NDC or DC space
- **N** name: identification (used for error file, workstation identifier, connection identifier, workstation type, specific escape function identification, GDP identifier, pick identifier, segment name and identification of a GKS function). In a programming language, not all these instances of the name data type need be bound to the same data type in the language.
- **E** enumeration type: a data type comprising a set of values. The set is defined by enumerating the identifiers which denote the values. This type could be mapped, for example, onto scalar types in Pascal, or onto integers in FORTRAN.

Alternatively, the data type can be a combination of simple types, thus:

f) a vector of values, for example, 2×R
g) a matrix of values, for example, 2×3×R
h) a list of values of one type: the type can be a simple type or a vector, for example, n×I and n×4×R
i) an array of values of simple type, for example, n×n×I
j) an ordered pair of different types, for example, (I;E)

or it can be:

- **D** data record: a compound data type, the content and structure of which are not defined in this standard.

An occurrence of n merely indicates a variable integer value and does not necessarily relate to other occurrences of n.

How these data types are represented in a given implementation is dependent on the features of the programming language and on the capabilities of the system. Each language dependent layer
has to map the GKS data types onto the data types available in the programming language.

For coordinate data, the relevant coordinate system is indicated:

k) WC : world coordinate system;
l) NDC : normalized device coordinate system;
m) DC : device coordinate systems.

See 4.3 for more information about coordinate systems in GKS. It should be pointed out that different coordinate systems may be used in a function call and in the state lists to describe the same entry.

Permitted values can be specified by:

n) a condition, for example, >0 or [0,1]; the latter implies that the value lies between 0 and 1 inclusively;
o) a standard range of integer values, for example, (1..4);
p) a range of integer values in which the maximum is determined by implementation or other constraints, for example (32..n). An occurrence of n does not necessarily imply any relationship with other occurrences of n; n merely denotes a variable integer in this context;
q) a list of values which constitute an enumeration type, for example, (SUPPRESSED,ALLOWED);
r) an ordered list of any of the above.

Initial values, if present, occur on the last column of the data structure lists. The following abbreviations occur:
s) undef: undefined value;
t) id: implementation dependent;
u) w.d.t: initial value taken from workstation description table.

If no initial value is given, the value is set by the relevant GKS function.
6.2 Operating state

Operating state value (static variable)

(GKCL, GKOP, WSOP, WSA, SGOP) E GKCL
### 6.3 GKS description table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level of GKS</td>
<td>(ea, eb, ec, 0a, 0b, 0c, 1a, 1b, 1c, 2a, 2b, 2c)</td>
<td>E</td>
</tr>
<tr>
<td>Number of available workstation types</td>
<td>(1, n)</td>
<td>I</td>
</tr>
<tr>
<td>List of available workstation types</td>
<td></td>
<td>n x N</td>
</tr>
<tr>
<td>Maximum number of simultaneously open workstations</td>
<td>(1, n)</td>
<td>I</td>
</tr>
<tr>
<td>Maximum number of simultaneously active workstations</td>
<td>(1, n)</td>
<td>I</td>
</tr>
<tr>
<td>Maximum number of workstations associated with a segment</td>
<td>(1, n)</td>
<td>I</td>
</tr>
<tr>
<td>Maximum normalization transformation number</td>
<td>(1, n)</td>
<td>I</td>
</tr>
</tbody>
</table>
6.4  GKS state list

set of open workstations
set of active workstations

current polyline index (1..n)

current linetype (-n..-1,1..n)

current linewidth scale factor ≥0

current polyline colour index (0..n)

current linetype ASF (BUNDLED,INDIVIDUAL)

current linewidth scale factor ASF (BUNDLED,INDIVIDUAL)

current polyline colour index ASF (BUNDLED,INDIVIDUAL)

current polymarker index (1..n)

current marker type (-n..-1,1..n)

current marker size scale factor ≥0

current polymarker colour index (0..n)

current marker type ASF (BUNDLED,INDIVIDUAL)

current marker size scale factor ASF (BUNDLED,INDIVIDUAL)

current polymarker colour index ASF (BUNDLED,INDIVIDUAL)

current text index (1..n)

current text font and precision (-n..-1,1..n,STRING,CHAR,STROKE)

(current character expansion factor >0

current text colour index (0..n)

(current text font and precision ASF (BUNDLED,INDIVIDUAL)

(current character expansion factor ASF (BUNDLED,INDIVIDUAL)

(current text colour index ASF (BUNDLED,INDIVIDUAL)

(current character height WC >0

current character up vector WC 2XR 0,1

current character width WC >0

current character base vector WC 2XR 1,0

current text path (RIGHT,LEFT,UP,DOWN) (NORMAL,LEFT,CENTRE,RIGHT,NORMAL,TOPTOP,CAP,HALF,BASE,BOTTOM) 2XE (NORMAL,NORMAL)

(current fill area index (1..n)

(current fill area interior style (HOLLOW,SOLID,PATTERN,HATCH)

(current fill area style index (-n..-1,1..n)

(current fill area colour index (0..n)

(current fill area interior style ASF (BUNDLED,INDIVIDUAL)

(current fill area style index ASF (BUNDLED,INDIVIDUAL)

(current fill area colour index ASF (BUNDLED,INDIVIDUAL)

(current pattern width vector WC 2XR 1,0

(current pattern height vector WC 2XR 0,1

(current pattern reference point WC P (0,0)

language binding dependent

current normalization transformation number (0..n)

list of normalization transformations ordered by

viewport input priority (initially in numerical order with 0 highest),

for every entry:

normalization transformation number (0..n)

entry number

window WC 4XR 0,1,0,1

viewport NDC 4XR 0,1,0,1
GKS state list

clipping indicator
clipping rectangle
name of open segment
set of segment names in use
set of segment state lists (one state list for every segment; see 6.7)
input queue (one entry for each event report)
for every entry:
  workstation identifier
  device number
  last of group of simultaneous events
  (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
input class
  (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
  (CLIP, NOCLIP)
  leaf
  name of open segment
  set of segment names in use
  set of segment state lists (one state list for every segment; see 6.7)
  input queue (one entry for each event report)
  for every entry:
    workstation identifier
    device number
    last of group of simultaneous events
    (a single event is indicated by LAST)
GKS data structures

6.5 Workstation state list

One workstation state list exists for every open workstation. For workstations of category MO, the values marked w.d.t in the following list are actually implementation dependent because the workstation description table does not contain the corresponding entries.

Entries in this group exist for all workstation categories
workstation identifier N
connection identifier N
workstation type N

the above 3 entries are initialized by OPEN WORKSTATION

Entries in this group do not exist for workstations of category INPUT and MI

workstation state (ACTIVE, INACTIVE) E
set of stored segments for this workstation n X N

Entries in this group do not exist for workstations of categories INPUT, WISS and MI

deferral mode (ASAP, BNIG, BNIL, ASTI) E
implicit regeneration mode (SUPPRESSED, ALLOWED) E
display surface empty (EMPTY, NOTEMPTY) E
new frame action necessary at update (NO, YES) E

number of polyline bundle table entries (5.n) I

for every entry:
polyline index (1.n) I
linetype (-n -1, 1.n) I
linewidth scale factor \( \geq 0 \) R
polyline colour index (0.n) I

number of polymarker bundle table entries (5.n) I

for every entry:
polymarker index (1.n) I
marker type (-n -1, 1.n) I
marker size scale factor \( \geq 0 \) R
polymarker colour index (0.n) I

number of text bundle table entries (2.n) I

for every entry:
text index (1.n) I

for text font and precision (-n -1, 1.n; STRING, CHAR, STROKE) (1,E) I
character expansion factor \( > 0 \) R
character spacing R

for text colour index (0.n) I

number of fill area bundle table entries (5.n) I

for every entry:
fill area index (1.n) I

for fill area interior style (HOLLOW, SOLID, PATTERN, HATCH) (E) I
### Workstation State List

- **Fill area style index**: 
  - \((-n, l, n)\)
  - \((0, n)\)
- **Fill area colour index**: 
  - \((0, n)\)
- **Number of pattern table entries**: 
  - \((0, n)\)
- **Table of pattern representations**:
  - For every entry:
    - **Pattern index**: 
      - \((1, n)\)
    - **Pattern array dimensions**: 
      - \((0, n)\)
    - **Pattern array**: 
      - \((n \times n)\)
- **Number of colour table entries**: 
  - \((2, n)\)
- **Table of colour representations**:
  - For every entry:
    - **Colour index**: 
      - \((0, n)\)
    - **Colour (red, green, blue intensities)**: 
      - \([0, 1] \times 3\)
- **Workstation transformation update state**: 
  - (NOTPENDING, PENDING)
  - **Requested workstation window**: 
    - NDC
  - **Current workstation window**: 
    - NDC
  - **Requested workstation viewport**: 
    - DC
  - **Current workstation viewport**: 
    - DC
  - Where \((x_d, y_d)\) is the display space size from the \(w.d.t\)

### GKS Data Structures

- **Input buffer size**: 
  - \((1, 1)\)
- **Requested workstation window**: 
  - NDC
  - \(4 \times R\)
  - \(0, 1, 0, 1\)
- **Current workstation window**: 
  - NDC
  - \(4 \times R\)
  - \(0, 1, 0, 1\)
- **Requested workstation viewport**: 
  - DC
  - \(4 \times R\)
  - \(0, x_d, 0, y_d\)
- **Current workstation viewport**: 
  - DC
  - \(4 \times R\)
  - \(c, x_d, 0, y_d\)

**Entries in this group do not exist for workstations of categories OUTPUT, WISS, MO and MI**

For every logical input device of class LOCATOR:

- **Locator device number**: 
  - \((1, n)\)
- **Operating mode**: 
  - \((\text{REQUEST, SAMPLE, EVENT})\)
- **Echo switch**: 
  - \((\text{ECHO, NOECHO})\)
- **Initial normalization transformation number**: 
  - \((0, n)\)
- **Initial locator position**: 
  - WC
  - \((-n, -1, 1, n)\)
- **Prompt and echo type**: 
  - DC
  - \(4 \times R\)
  - \[I, 0, 1\]
- **Echo area**: 
  - DC
  - \(4 \times R\)
  - \[I, 0, 1\]

For every logical input device of class STROKE:

- **Stroke device number**: 
  - \((1, n)\)
- **Operating mode**: 
  - \((\text{REQUEST, SAMPLE, EVENT})\)
- **Echo switch**: 
  - \((\text{ECHO, NOECHO})\)
- **Initial normalization transformation number**: 
  - \((0, n)\)
- **Initial number of points**: 
  - \((0, n)\)
- **Initial points in stroke**: 
  - WC
  - \([n \times P]\)
  - \[\text{empty}\]
- **Prompt and echo type**: 
  - DC
  - \((-n, -1, 1, n)\)
  - \[I, 0, 1\]
- **Echo area**: 
  - DC
  - \(4 \times R\)
  - \[I, 0, 1\]

**Input buffer size**: 
- \((1, n)\)
GKS data structures

for every logical input device of class VALUATOR:

valuator device number
operating mode
(REQUEST,SAMPLE,EVENT)
echo switch
(ECHO,NOECHO)
initial value
prompt and echo type
(-n ..-1,1..n)
echo area
DC
valuator data record containing at least:
low value
high value

for every logical input device of class CHOICE:

choice device number
operating mode
(REQUEST,SAMPLE,EVENT)
echo switch
(ECHO,NOECHO)
initial status
(OK,NOCHOICE)
initial choice number
(-n ..-1,1..n)
echo area
DC
choice data record

for every logical input device of class PICK:

pick device number
operating mode
(REQUEST,SAMPLE,EVENT)
echo switch
(ECHO,NOECHO)
initial status
(OK,NOPICK)
initial segment
N
initial pick identifier
N
prompt and echo type
(-n ..-1,1..n)
echo area
DC
pick data record

for every logical input device of class STRING:

string device number
operating mode
(REQUEST,SAMPLE,EVENT)
echo switch
(ECHO,NOECHO)
initial string
prompt and echo type
(-n ..-1,1..n)
echo area
DC
string data record containing at least:
input buffer size
(1..n)
initial cursor position
(1..n)
6.6 Workstation description table

There are three special categories of GKS workstation:

a) WISS (Workstation Independent Segment Storage);

b) MO (GKSM Output);

c) MI (GKSM Input).

For levels m, 0 and 1 there is no WISS; for level 2 there is exactly one WISS.

There may be a number of different workstation types for MO and MI to accommodate different metafile formats. These special workstations have a restricted workstation description table.

Further workstation types may be assigned and are implementation dependent, for example:

d) storage tube type 1;

e) flat bed plotter 1.

The workstation description tables cannot be changed by the application program. There is only one workstation description table for each workstation type available in a given implementation.

<table>
<thead>
<tr>
<th>Entries in this group exist for all workstation categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>workstation type</td>
</tr>
<tr>
<td>workstation category</td>
</tr>
<tr>
<td>(OUTPUT,INPUT,OUTIN,WISS,MO,MI)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Entries in this group do not exist for workstations of categories WISS, MO and MI</th>
</tr>
</thead>
<tbody>
<tr>
<td>device coordinate units</td>
</tr>
<tr>
<td>display space size</td>
</tr>
<tr>
<td>(visible area of the display surface or available area on tablet for workstations of category INPUT)</td>
</tr>
<tr>
<td>in device coordinate units</td>
</tr>
<tr>
<td>in raster units</td>
</tr>
<tr>
<td>(for vector displays, for example, the raster units give the highest possible resolution, for raster displays, the number of columns and lines of the raster array)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Entries in this group do not exist for workstations of categories INPUT, WISS, MO and MI</th>
</tr>
</thead>
<tbody>
<tr>
<td>raster or vector display</td>
</tr>
<tr>
<td>(VECTOR = vector display, RASTER = raster device, OTHER = other device, for example, vector+ raster)</td>
</tr>
<tr>
<td>dynamic modification accepted for</td>
</tr>
<tr>
<td>polyline bundle representation</td>
</tr>
<tr>
<td>polymarker bundle representation</td>
</tr>
<tr>
<td>text bundle representation</td>
</tr>
<tr>
<td>fill area bundle representation</td>
</tr>
<tr>
<td>pattern representation</td>
</tr>
<tr>
<td>colour representation</td>
</tr>
<tr>
<td>workstation transformation</td>
</tr>
<tr>
<td>where.</td>
</tr>
<tr>
<td>IRG: implicit regeneration necessary (may be deferred)</td>
</tr>
<tr>
<td>IMM: performed immediately</td>
</tr>
<tr>
<td>default value for:</td>
</tr>
<tr>
<td>deferral mode</td>
</tr>
<tr>
<td>implicit regeneration mode</td>
</tr>
</tbody>
</table>
### GKS data structures

<table>
<thead>
<tr>
<th>Description</th>
<th>Expression</th>
<th>ID</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of available linetypes</td>
<td>(4..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>List of available linetypes</td>
<td>(-n..-1..n.)</td>
<td>n X</td>
<td>id</td>
</tr>
<tr>
<td>Number of available linewidths</td>
<td>(0..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Nominal linewidth</td>
<td>DC &gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Minimum linewidth</td>
<td>DC &gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Maximum linewidth</td>
<td>DC &gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Number of predefined polyline indices (bundles)</td>
<td>(5..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Table of predefined polyline bundles, for every entry:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linetype (within range of predefined colour indices)</td>
<td>(-n..-1..n.)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Linewidth scale factor</td>
<td>R</td>
<td>id</td>
<td></td>
</tr>
<tr>
<td>Polyl ine colour index (within range of predefined colour indices)</td>
<td>(0..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Number of available marker types</td>
<td>(5..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>List of available marker types</td>
<td>(-n..-1..n.)</td>
<td>n X</td>
<td>id</td>
</tr>
<tr>
<td>Number of available marker sizes</td>
<td>(0..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Nominal marker size</td>
<td>DC &gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Minimum marker size</td>
<td>DC &gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Maximum marker size</td>
<td>DC &gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Number of predefined polymarker indices (bundles)</td>
<td>(5..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Table of predefined polymarker bundles, for every entry:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Marker type (within range of predefined colour indices)</td>
<td>(-n..-1..n.)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Marker size scale factor</td>
<td>R</td>
<td>id</td>
<td></td>
</tr>
<tr>
<td>Polymarker colour index (within range of predefined colour indices)</td>
<td>(0..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Number of text font and precision pairs</td>
<td>(1..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>List of text font and precision pairs</td>
<td>(-n..-1..n.;STRING,CHAR,STROKE)</td>
<td>n X 1,E</td>
<td>id</td>
</tr>
<tr>
<td>Number of available character expansion factors</td>
<td>(0..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Minimum character expansion factor</td>
<td>&gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Maximum character expansion factor</td>
<td>&gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Maximum character expansion factor (if the available character expansion factors vary between fonts, these values are for font 1)</td>
<td>&gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Number of available character heights</td>
<td>(0..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Minimum character height</td>
<td>DC &gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Maximum character height</td>
<td>DC &gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Number of predefined text indices (bundles)</td>
<td>(2..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Table of predefined text bundles, for every entry:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text font and precision</td>
<td>(-n..-1..n.;STRING,CHAR,STROKE)</td>
<td>1,E</td>
<td>id</td>
</tr>
<tr>
<td>Character expansion factor</td>
<td>&gt;0</td>
<td>R</td>
<td>id</td>
</tr>
<tr>
<td>Character spacing</td>
<td>R</td>
<td>id</td>
<td></td>
</tr>
<tr>
<td>Text colour index (within range of predefined colour indices)</td>
<td>(0..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>Number of available fill area interior styles</td>
<td>(1..4)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>List of available fill area interior styles</td>
<td>(HOLLOW,SOLID,PATTERN,HATCH)</td>
<td>n X E</td>
<td>id</td>
</tr>
<tr>
<td>Number of available hatch styles</td>
<td>(0..n)</td>
<td>1</td>
<td>id</td>
</tr>
<tr>
<td>List of available hatch styles</td>
<td>(-n..-1..n.)</td>
<td>n X</td>
<td>id</td>
</tr>
</tbody>
</table>
Workstation description table

<table>
<thead>
<tr>
<th>Structure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of predefined fill area indices(bundles)</td>
<td>(5..n)</td>
</tr>
<tr>
<td>table of predefined fill area bundles,</td>
<td>I</td>
</tr>
<tr>
<td>for every entry:</td>
<td>id</td>
</tr>
<tr>
<td>fill area interior style</td>
<td>(HOLLOW, SOLID, PATTERN, HATCH)</td>
</tr>
<tr>
<td>fill area style index</td>
<td>I</td>
</tr>
<tr>
<td>(for interior style PATTERN is within range of predefined pattern indices)</td>
<td>id</td>
</tr>
<tr>
<td>(for interior style HATCH is within range of available hatch styles)</td>
<td></td>
</tr>
<tr>
<td>fill area colour index (within range of predefined colour indices)</td>
<td>(0..n)</td>
</tr>
<tr>
<td>number of predefined pattern indices (representations)</td>
<td>(0..n)</td>
</tr>
<tr>
<td>table of predefined pattern representations,</td>
<td></td>
</tr>
<tr>
<td>for every entry:</td>
<td>id</td>
</tr>
<tr>
<td>pattern array dimensions</td>
<td>(0..n)</td>
</tr>
<tr>
<td>pattern array</td>
<td>2X1</td>
</tr>
<tr>
<td>number of available colours or intensities</td>
<td>I</td>
</tr>
<tr>
<td>(a value of 0 indicates that a continuous range of colours is supported)</td>
<td>id</td>
</tr>
<tr>
<td>colour available</td>
<td>(COLOUR, MONOCHROME)</td>
</tr>
<tr>
<td>number of predefined colour indices (representations)</td>
<td>(1..n)</td>
</tr>
<tr>
<td>table of predefined colour representations,</td>
<td></td>
</tr>
<tr>
<td>for every entry:</td>
<td>id</td>
</tr>
<tr>
<td>colour(red, green, blue intensities)</td>
<td>[0,1]</td>
</tr>
<tr>
<td>number of available generalized drawing primitives</td>
<td>3XR</td>
</tr>
<tr>
<td>list of available generalized drawing primitives (may be empty),</td>
<td>I</td>
</tr>
<tr>
<td>for every GDP:</td>
<td>id</td>
</tr>
<tr>
<td>GDP identifier</td>
<td>N</td>
</tr>
<tr>
<td>number of sets of attributes used</td>
<td>(0..n)</td>
</tr>
<tr>
<td>list of sets of attributes used</td>
<td>POLYLINE, POLYMARKER, TEXT, FILL AREA</td>
</tr>
<tr>
<td>maximum number of polyline bundle table entries</td>
<td>(5..n)</td>
</tr>
<tr>
<td>maximum number of polymarker bundle table entries</td>
<td>(5..n)</td>
</tr>
<tr>
<td>maximum number of text bundle table entries</td>
<td>(5..n)</td>
</tr>
<tr>
<td>maximum number of fill area bundle table entries</td>
<td>(5..n)</td>
</tr>
<tr>
<td>maximum number of pattern indices</td>
<td>(0..n)</td>
</tr>
<tr>
<td>maximum number of colour indices</td>
<td>(0..n)</td>
</tr>
<tr>
<td>number of segment priorities supported</td>
<td></td>
</tr>
<tr>
<td>(a value of 0 indicates that a continuous range of priorities is supported)</td>
<td></td>
</tr>
<tr>
<td>dynamic modification accepted for:</td>
<td>id</td>
</tr>
<tr>
<td>segment transformation</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td>visibility (visible → invisible)</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td>visibility (invisible → visible)</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td>highlighting</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td>segment priority</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td>adding primitives to open segment overlapping segment of higher priority</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td>delete segment</td>
<td>(IRG, IMM)</td>
</tr>
<tr>
<td>where:</td>
<td>id</td>
</tr>
<tr>
<td>IRG: implicit regeneration necessary (may be deferred)</td>
<td></td>
</tr>
<tr>
<td>IMM: performed immediately</td>
<td></td>
</tr>
</tbody>
</table>

Page 208
Entries in this group do not exist for workstations of categories OUTPUT, WISS, MO and MI

for every logical input device of class LOCATOR:

- **locator device number**
- **default initial locator position**
- **number of available prompt and echo types**
- **list of available prompt and echo types**
- **default echo area**
- **default locator data record**

for every logical input device of class STROKE:

- **stroke device number**
- **maximum input buffer size**
- **number of available prompt and echo types**
- **list of available prompt and echo types**
- **default echo area**
- **default stroke data record containing at least:**
- **input buffer size**

for every logical input device of class VALUATOR:

- **valuator device number**
- **default initial value**
- **number of available prompt and echo types**
- **list of available prompt and echo types**
- **default echo area**
- **default valuator data record containing at least:**
- **low value**
- **high value**

for every logical input device of class CHOICE:

- **choice device number**
- **maximum number of choice alternatives**
- **number of available prompt and echo types**
- **list of available prompt and echo types**
- **default echo area**
- **default choice data record**

for every logical input device of class PICK:

- **pick device number**
- **number of available prompt and echo types**
- **list of available prompt and echo types**
- **default echo area**
- **default pick data record**

for every logical input device of class STRING:

- **string device number**
- **maximum input buffer size**
- **number of available prompt and echo types**
- **list of available prompt and echo types**
- **default echo area**
- **default string data record containing at least:**
- **input buffer size**
- **initial cursor position**
6.7 Segment state list
One segment state list exists for the open segment and for each stored segment.

- **Segment name**
- **Set of associated workstations**
- **Segment transformation matrix**
  - (the elements $M_{13}$ and $M_{23}$ are in NDC coordinates and the other elements are unitless)
- **Visibility**
- **Highlighting**
- **Segment priority**
- **Detectability**

<table>
<thead>
<tr>
<th>Active workstations at create segment</th>
<th>N</th>
<th>n x N</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 x 3 x R</td>
<td>1,0,0</td>
<td></td>
</tr>
<tr>
<td>0,1,0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(Visible, Invisible)</th>
<th>E</th>
<th>Visible</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Normal, Highlighted)</td>
<td>E</td>
<td>Normal</td>
</tr>
<tr>
<td>[0,1]</td>
<td>R</td>
<td>0</td>
</tr>
<tr>
<td>(Undetectable, Detectable)</td>
<td>E</td>
<td>Undetectable</td>
</tr>
</tbody>
</table>
6.8 GKS error state list

| error state | (OFF,ON) | E | OFF |
| error file | | N | id |

Identification of one of the logical input devices that caused an input queue overflow:

| workstation identifier | | N | undef |
| input class | (LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING) | E | undef |
| logical input device number | (1..n) | I | undef |
## Appendix A  Function lists

NOTE. This appendix is not part of the Standard, but provides additional information.

### A.1 Alphabetic

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCUMULATE TRANSFORMATION MATRIX</td>
<td>193</td>
</tr>
<tr>
<td>ACTIVATE WORKSTATION</td>
<td>76</td>
</tr>
<tr>
<td>ASSOCIATE SEGMENT WITH WORKSTATION</td>
<td>113</td>
</tr>
<tr>
<td>AWAIT EVENT</td>
<td>138</td>
</tr>
<tr>
<td>CELL ARRAY</td>
<td>85</td>
</tr>
<tr>
<td>CLEAR WORKSTATION</td>
<td>76</td>
</tr>
<tr>
<td>CLOSE GKS</td>
<td>74</td>
</tr>
<tr>
<td>CLOSE SEGMENT</td>
<td>111</td>
</tr>
<tr>
<td>CLOSE WORKSTATION</td>
<td>75</td>
</tr>
<tr>
<td>COPY SEGMENT TO WORKSTATION</td>
<td>113</td>
</tr>
<tr>
<td>CREATE SEGMENT</td>
<td>111</td>
</tr>
<tr>
<td>DEACTIVATE WORKSTATION</td>
<td>76</td>
</tr>
<tr>
<td>DELETE SEGMENT</td>
<td>112</td>
</tr>
<tr>
<td>DELETE SEGMENT FROM WORKSTATION</td>
<td>112</td>
</tr>
<tr>
<td>EMERGENCY CLOSE GKS</td>
<td>195</td>
</tr>
<tr>
<td>ERROR HANDLING</td>
<td>195</td>
</tr>
<tr>
<td>ERROR LOGGING</td>
<td>196</td>
</tr>
<tr>
<td>ESCAPE</td>
<td>80</td>
</tr>
<tr>
<td>EVALUATE TRANSFORMATION MATRIX</td>
<td>193</td>
</tr>
<tr>
<td>FILL AREA</td>
<td>83</td>
</tr>
<tr>
<td>FLUSH DEVICE EVENTS</td>
<td>139</td>
</tr>
<tr>
<td>GENERALIZED DRAWING PRIMITIVE (GDP)</td>
<td>86</td>
</tr>
<tr>
<td>GET CHOICE</td>
<td>140</td>
</tr>
<tr>
<td>GET ITEM TYPE FROM GKSM</td>
<td>142</td>
</tr>
<tr>
<td>GET LOCATOR</td>
<td>139</td>
</tr>
<tr>
<td>GET PICK</td>
<td>140</td>
</tr>
<tr>
<td>GET STRING</td>
<td>141</td>
</tr>
<tr>
<td>GET STROKE</td>
<td>139</td>
</tr>
<tr>
<td>GET VALUATOR</td>
<td>140</td>
</tr>
<tr>
<td>INITIALISE CHOICE</td>
<td>124</td>
</tr>
<tr>
<td>INITIALISE LOCATOR</td>
<td>119</td>
</tr>
<tr>
<td>INITIALISE PICK</td>
<td>126</td>
</tr>
<tr>
<td>INITIALISE STRING</td>
<td>127</td>
</tr>
<tr>
<td>INITIALISE STROKE</td>
<td>121</td>
</tr>
<tr>
<td>INITIALISE VALUATOR</td>
<td>123</td>
</tr>
<tr>
<td>INQUIRE CHOICE DEVICE STATE</td>
<td>168</td>
</tr>
<tr>
<td>INQUIRE CLIPPING</td>
<td>151</td>
</tr>
<tr>
<td>INQUIRE COLOUR FACILITIES</td>
<td>180</td>
</tr>
<tr>
<td>INQUIRE COLOUR REPRESENTATION</td>
<td>164</td>
</tr>
<tr>
<td>INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES</td>
<td>149</td>
</tr>
<tr>
<td>INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER</td>
<td>149</td>
</tr>
<tr>
<td>INQUIRE CURRENT PICK IDENTIFIER VALUE</td>
<td>148</td>
</tr>
<tr>
<td>INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES</td>
<td>148</td>
</tr>
<tr>
<td>INQUIRE DEFAULT CHOICE DEVICE DATA</td>
<td>187</td>
</tr>
<tr>
<td>INQUIRE DEFAULT DEFERRAL STATE VALUES</td>
<td>173</td>
</tr>
<tr>
<td>INQUIRE DEFAULT LOCATOR DEVICE DATA</td>
<td>185</td>
</tr>
<tr>
<td>INQUIRE DEFAULT PICK DEVICE DATA</td>
<td>188</td>
</tr>
</tbody>
</table>
INQUIRE DEFAULT STRING DEVICE DATA 188
INQUIRE DEFAULT STROKE DEVICE DATA 186
INQUIRE DEFAULT VALUATOR DEVICE DATA 186
INQUIRE DISPLAY SPACE SIZE 171
INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES 184
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES 172
INQUIRE FILL AREA FACILITIES 177
INQUIRE FILL AREA REPRESENTATION 162
INQUIRE GENERALIZED DRAWING PRIMITIVE 182
INQUIRE INPUT QUEUE OVERFLOW 192
INQUIRE LEVEL OF GKS 145
INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES 181
INQUIRE LIST OF AVAILABLE WORKSTATION TYPES 145
INQUIRE LIST OF COLOUR INDICES 164
INQUIRE LIST OF FILL AREA INDICES 161
INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS 150
INQUIRE LIST OF PATTERN INDICES 162
INQUIRE LIST OF POLYLINE INDICES 154
INQUIRE LIST OF POLYMARKER INDICES 156
INQUIRE LIST OF TEXT INDICES 157
INQUIRE LOCATOR DEVICE STATE 166
INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES 182
INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER 146
INQUIRE MORE SIMULTANEOUS EVENTS 152
INQUIRE NAME OF OPEN SEGMENT 151
INQUIRE NORMALIZATION TRANSFORMATION 150
INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES 184
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED 183
INQUIRE OPERATING STATE VALUE 144
INQUIRE PATTERN FACILITIES 179
INQUIRE PATTERN REPRESENTATION 163
INQUIRE PICK DEVICE STATE 169
INQUIRE PIXEL 191
INQUIRE PIXEL ARRAY 191
INQUIRE PIXEL ARRAY DIMENSIONS 190
INQUIRE POLYLINE FACILITIES 173
INQUIRE POLYLINE REPRESENTATION 155
INQUIRE POLYMARKER FACILITIES 175
INQUIRE POLYMARKER REPRESENTATION 156
INQUIRE PREDEFINED COLOUR REPRESENTATION 180
INQUIRE PREDEFINED FILL AREA REPRESENTATION 178
INQUIRE PREDEFINED PATTERN REPRESENTATION 179
INQUIRE PREDEFINED POLYLINE REPRESENTATION 174
INQUIRE PREDEFINED POLYMARKER REPRESENTATION 175
INQUIRE PREDEFINED TEXT REPRESENTATION 177
INQUIRE SEGMENT ATTRIBUTES 189
INQUIRE SET OF ACTIVE WORKSTATIONS 147
INQUIRE SET OF ASSOCIATED WORKSTATIONS 189
INQUIRE SET OF OPEN WORKSTATIONS 147
INQUIRE SET OF SEGMENT NAMES IN USE 152
INQUIRE SET OF SEGMENT NAMES ON WORKSTATION 166
INQUIRE STRING DEVICE STATE 170
INQUIRE STROKE DEVICE STATE 167
INQUIRE TEXT EXTENT 158
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INQUIRE TEXT FACILITIES</td>
<td>176</td>
</tr>
<tr>
<td>INQUIRE TEXT REPRESENTATION</td>
<td>157</td>
</tr>
<tr>
<td>INQUIRE VALUATOR DEVICE STATE</td>
<td>168</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION CATEGORY</td>
<td>170</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION CLASSIFICATION</td>
<td>171</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION CONNECTION AND TYPE</td>
<td>153</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES</td>
<td>154</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION MAXIMUM NUMBERS</td>
<td>146</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION STATE</td>
<td>153</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION TRANSFORMATION</td>
<td>165</td>
</tr>
<tr>
<td>INSERT SEGMENT</td>
<td>114</td>
</tr>
<tr>
<td>INTERPRET ITEM</td>
<td>143</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>80</td>
</tr>
<tr>
<td>OPEN GKS</td>
<td>74</td>
</tr>
<tr>
<td>OPEN WORKSTATION</td>
<td>74</td>
</tr>
<tr>
<td>POLYLINE</td>
<td>82</td>
</tr>
<tr>
<td>POLYMARKER</td>
<td>82</td>
</tr>
<tr>
<td>READ ITEM FROM GKSM</td>
<td>143</td>
</tr>
<tr>
<td>REDRAW ALL SEGMENTS ON WORKSTATION</td>
<td>77</td>
</tr>
<tr>
<td>RENAME SEGMENT</td>
<td>111</td>
</tr>
<tr>
<td>REQUEST CHOICE</td>
<td>133</td>
</tr>
<tr>
<td>REQUEST LOCATOR</td>
<td>131</td>
</tr>
<tr>
<td>REQUEST PICK</td>
<td>134</td>
</tr>
<tr>
<td>REQUEST STRING</td>
<td>134</td>
</tr>
<tr>
<td>REQUEST STROKE</td>
<td>132</td>
</tr>
<tr>
<td>REQUEST VALUATOR</td>
<td>133</td>
</tr>
<tr>
<td>SAMPLE CHOICE</td>
<td>136</td>
</tr>
<tr>
<td>SAMPLE LOCATOR</td>
<td>135</td>
</tr>
<tr>
<td>SAMPLE PICK</td>
<td>137</td>
</tr>
<tr>
<td>SAMPLE STRING</td>
<td>137</td>
</tr>
<tr>
<td>SAMPLE STROKE</td>
<td>135</td>
</tr>
<tr>
<td>SAMPLE VALUATOR</td>
<td>136</td>
</tr>
<tr>
<td>SELECT NORMALIZATION TRANSFORMATION</td>
<td>108</td>
</tr>
<tr>
<td>SET ASPECT SOURCE FLAGS</td>
<td>99</td>
</tr>
<tr>
<td>SET CHARACTER EXPANSION FACTOR</td>
<td>93</td>
</tr>
<tr>
<td>SET CHARACTER HEIGHT</td>
<td>95</td>
</tr>
<tr>
<td>SET CHARACTER SPACING</td>
<td>94</td>
</tr>
<tr>
<td>SET CHARACTER UP VECTOR</td>
<td>95</td>
</tr>
<tr>
<td>SET CHOICE MODE</td>
<td>130</td>
</tr>
<tr>
<td>SET CLIPPING INDICATOR</td>
<td>108</td>
</tr>
<tr>
<td>SET COLOUR REPRESENTATION</td>
<td>105</td>
</tr>
<tr>
<td>SET DEFERRAL STATE</td>
<td>79</td>
</tr>
<tr>
<td>SET DETECTABILITY</td>
<td>117</td>
</tr>
<tr>
<td>SET FILL AREA COLOUR INDEX</td>
<td>98</td>
</tr>
<tr>
<td>SET FILL AREA INDEX</td>
<td>96</td>
</tr>
<tr>
<td>SET FILL AREA INTERIOR STYLE</td>
<td>96</td>
</tr>
<tr>
<td>SET FILL AREA REPRESENTATION</td>
<td>103</td>
</tr>
<tr>
<td>SET FILL AREA STYLE INDEX</td>
<td>97</td>
</tr>
<tr>
<td>SET HIGHLIGHTING</td>
<td>116</td>
</tr>
<tr>
<td>SET LINETYPE</td>
<td>89</td>
</tr>
<tr>
<td>SET LINETYPE SCALE FACTOR</td>
<td>90</td>
</tr>
<tr>
<td>SET LOCATOR MODE</td>
<td>128</td>
</tr>
<tr>
<td>SET MARKER SIZE SCALE FACTOR</td>
<td>92</td>
</tr>
<tr>
<td>SET MARKER TYPE</td>
<td>91</td>
</tr>
</tbody>
</table>
A.2.1 Control functions 5.2
OPEN GKS 74
CLOSE GKS 74
OPEN WORKSTATION 74
CLOSE WORKSTATION 75
ACTIVATE WORKSTATION 76
DEACTIVATE WORKSTATION 76
CLEAR WORKSTATION 76
REDRAW ALL SEGMENTS ON WORKSTATION 77
UPDATE WORKSTATION 78
SET DEFERRAL STATE 79
MESSAGE 80
ESCAPE 80

A.2.2 Output functions 5.3
POLYLINE 82
POLYMARKER 82
TEXT 83
FILL AREA 83
CELL ARRAY 85
GENERALIZED DRAWING PRIMITIVE (GDP)

A.2.8 Output attributes 5.4

A.2.8.1 Workstation independent primitive attributes 5.4.1
SET POLYLINE INDEX 89
SET LINETYPE 89
SET LINEWIDTH SCALE FACTOR 90
SET POLYLINE COLOUR INDEX 90
SET POLYMARKER INDEX 91
SET MARKER TYPE 91
SET MARKER SIZE SCALE FACTOR 92
SET POLYMARKER COLOUR INDEX 92
SET TEXT INDEX 92
SET TEXT FONT AND PRECISION 93
SET CHARACTER EXPANSION FACTOR 93
SET CHARACTER SPACING 94
SET TEXT COLOUR INDEX 94
SET CHARACTER HEIGHT 95
SET CHARACTER UP VECTOR 95
SET TEXT PATH 95
SET TEXT Alignment 96
SET FILL AREA INDEX 96
SET FILL AREA INTERIOR STYLE 96
SET FILL AREA STYLE INDEX 97
SET FILL AREA COLOUR INDEX 98
SET PATTERN SIZE 98
SET PATTERN REFERENCE POINT 98
SET ASPECT SOURCE FLAGS 99
SET PICK IDENTIFIER 99

A.2.8.2 Workstation attributes (representations) 5.4.2
SET POLYLINE REPRESENTATION 100
SET POLYMARKER REPRESENTATION 101
SET TEXT REPRESENTATION 102
SET FILL AREA REPRESENTATION 103
SET PATTERN REPRESENTATION 104
SET COLOUR REPRESENTATION 105

A.2.4 Transformation functions 5.5

A.2.4.1 Normalization transformation 5.5.1
SET WINDOW 107
SET VIEWPORT 107
SET VIEWPORT INPUT PRIORITY 107
SELECT NORMALIZATION TRANSFORMATION 108
SET CLIPPING INDICATOR 108

A.2.4.2 Workstation transformation 5.5.2
SET WORKSTATION WINDOW 109
SET WORKSTATION VIEWPORT 109
### A.2.5 Segment functions 5.6

#### A.2.5.1 Segment manipulation functions 5.6.1
- `CREATE SEGMENT` 111
- `CLOSE SEGMENT` 111
- `RENAME SEGMENT` 111
- `DELETE SEGMENT` 112
- `DELETE SEGMENT FROM WORKSTATION` 112
- `ASSOCIATE SEGMENT WITH WORKSTATION` 113
- `COPY SEGMENT TO WORKSTATION` 113
- `INSERT SEGMENT` 114

#### A.2.5.2 Segment attributes 5.6.2
- `SET SEGMENT TRANSFORMATION` 115
- `SET VISIBILITY` 116
- `SET HIGHLIGHTING` 116
- `SET SEGMENT PRIORITY` 117
- `SET DETECTABILITY` 117

### A.2.6 Input functions 5.7

#### A.2.6.1 Initialisation of input devices 5.7.1
- `INITIALISE LOCATOR` 119
- `INITIALISE STROKE` 121
- `INITIALISE VALUATOR` 123
- `INITIALISE CHOICE` 124
- `INITIALISE PICK` 126
- `INITIALISE STRING` 127

#### A.2.6.2 Setting the mode of input devices 5.7.2
- `SET LOCATOR MODE` 128
- `SET STROKE MODE` 129
- `SET VALUATOR MODE` 129
- `SET CHOICE MODE` 130
- `SET PICK MODE` 130
- `SET STRING MODE` 131

#### A.2.6.3 Request input functions 5.7.3
- `REQUEST LOCATOR` 131
- `REQUEST STROKE` 132
- `REQUEST VALUATOR` 133
- `REQUEST CHOICE` 133
- `REQUEST PICK` 134
- `REQUEST STRING` 134

#### A.2.6.4 Sample input functions 5.7.4
- `SAMPLE LOCATOR` 135
- `SAMPLE STROKE` 135
- `SAMPLE VALUATOR` 136
- `SAMPLE CHOICE` 136
- `SAMPLE PICK` 137
- `SAMPLE STRING` 137
### A.2.6.5 Event input functions 5.7.5

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWAIT EVENT</td>
<td>138</td>
</tr>
<tr>
<td>FLUSH DEVICE EVENTS</td>
<td>139</td>
</tr>
<tr>
<td>GET LOCATOR</td>
<td>139</td>
</tr>
<tr>
<td>GET STROKE</td>
<td>139</td>
</tr>
<tr>
<td>GET VALUATOR</td>
<td>140</td>
</tr>
<tr>
<td>GET CHOICE</td>
<td>140</td>
</tr>
<tr>
<td>GET PICK</td>
<td>140</td>
</tr>
<tr>
<td>GET STRING</td>
<td>141</td>
</tr>
</tbody>
</table>

### A.2.7 Metafile functions 5.8

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRITE ITEM TO GKSM</td>
<td>142</td>
</tr>
<tr>
<td>GET ITEM TYPE FROM GKSM</td>
<td>142</td>
</tr>
<tr>
<td>READ ITEM FROM GKSM</td>
<td>143</td>
</tr>
<tr>
<td>INTERPRET ITEM</td>
<td>143</td>
</tr>
</tbody>
</table>

### A.2.8 Inquiry functions 5.9

#### A.2.8.1 Inquiry function for operating state value 5.9.2

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INQUIRE OPERATING STATE VALUE</td>
<td>144</td>
</tr>
</tbody>
</table>

#### A.2.8.2 Inquiry functions for GKS description table 5.9.3

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INQUIRE LEVEL OF GKS</td>
<td>145</td>
</tr>
<tr>
<td>INQUIRE LIST OF AVAILABLE WORKSTATION TYPES</td>
<td>145</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION MAXIMUM NUMBERS</td>
<td>146</td>
</tr>
<tr>
<td>INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER</td>
<td>146</td>
</tr>
</tbody>
</table>

#### A.2.8.3 Inquiry functions for GKS state list 5.9.4

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INQUIRE SET OF OPEN WORKSTATIONS</td>
<td>147</td>
</tr>
<tr>
<td>INQUIRE SET OF ACTIVE WORKSTATIONS</td>
<td>147</td>
</tr>
<tr>
<td>INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES</td>
<td>148</td>
</tr>
<tr>
<td>INQUIRE CURRENT PICK IDENTIFIER VALUE</td>
<td>148</td>
</tr>
<tr>
<td>INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES</td>
<td>149</td>
</tr>
<tr>
<td>INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER</td>
<td>149</td>
</tr>
<tr>
<td>INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS</td>
<td>150</td>
</tr>
<tr>
<td>INQUIRE NORMALIZATION TRANSFORMATION</td>
<td>150</td>
</tr>
<tr>
<td>INQUIRE CLIPPING</td>
<td>151</td>
</tr>
<tr>
<td>INQUIRE NAME OF OPEN SEGMENT</td>
<td>151</td>
</tr>
<tr>
<td>INQUIRE SET OF SEGMENT NAMES IN USE</td>
<td>152</td>
</tr>
<tr>
<td>INQUIRE MORE SIMULTANEOUS EVENTS</td>
<td>152</td>
</tr>
</tbody>
</table>

#### A.2.8.4 Inquiry functions for workstation state list 5.9.5

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INQUIRE WORKSTATION CONNECTION AND TYPE</td>
<td>153</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION STATE</td>
<td>153</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES</td>
<td>154</td>
</tr>
<tr>
<td>INQUIRE LIST OF POLYLINE INDICES</td>
<td>154</td>
</tr>
<tr>
<td>INQUIRE POLYLINE REPRESENTATION</td>
<td>155</td>
</tr>
<tr>
<td>INQUIRE LIST OF POLYMARKER INDICES</td>
<td>156</td>
</tr>
<tr>
<td>INQUIRE POLYMARKER REPRESENTATION</td>
<td>156</td>
</tr>
<tr>
<td>INQUIRE LIST OF TEXT INDICES</td>
<td>157</td>
</tr>
<tr>
<td>INQUIRE TEXT REPRESENTATION</td>
<td>157</td>
</tr>
<tr>
<td>INQUIRE TEXT EXTENT</td>
<td>158</td>
</tr>
<tr>
<td>INQUIRE LIST OF FILL AREA INDICES</td>
<td>161</td>
</tr>
<tr>
<td>INQUIRE FILL AREA REPRESENTATION</td>
<td>162</td>
</tr>
</tbody>
</table>
INQUIRE LIST OF PATTERN INDICES 162
INQUIRE PATTERN REPRESENTATION 163
INQUIRE LIST OF COLOUR INDICES 164
INQUIRE COLOUR REPRESENTATION 164
INQUIRE WORKSTATION TRANSFORMATION 165
INQUIRE SET OF SEGMENT NAMES ON WORKSTATION 166
INQUIRE LOCATOR DEVICE STATE 166
INQUIRE STROKE DEVICE STATE 167
INQUIRE VALUATOR DEVICE STATE 168
INQUIRE CHOICE DEVICE STATE 168
INQUIRE PICK DEVICE STATE 169
INQUIRE STRING DEVICE STATE 170

A.2.8.5 Inquiry functions for workstation description table 5.9.6
INQUIRE WORKSTATION CATEGORY 170
INQUIRE WORKSTATION CLASSIFICATION 171
INQUIRE DISPLAY SPACE SIZE 171
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES 172
INQUIRE DEFAULT DEFERRAL STATE VALUES 173
INQUIRE POLYLINE FACILITIES 173
INQUIRE PREDEFINED POLYLINE REPRESENTATION 174
INQUIRE POLYMARKER FACILITIES 175
INQUIRE PREDEFINED POLYMARKER REPRESENTATION 175
INQUIRE TEXT FACILITIES 176
INQUIRE PREDEFINED TEXT REPRESENTATION 177
INQUIRE FILL AREA FACILITIES 177
INQUIRE PREDEFINED FILL AREA REPRESENTATION 178
INQUIRE PATTERN FACILITIES 179
INQUIRE PREDEFINED PATTERN REPRESENTATION 179
INQUIRE COLOUR FACILITIES 180
INQUIRE PREDEFINED COLOUR REPRESENTATION 180
INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES 181
INQUIRE GENERALIZED DRAWING PRIMITIVE 182
INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES 182
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED 183
INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES 184
INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES 184
INQUIRE DEFAULT LOCATOR DEVICE DATA 185
INQUIRE DEFAULT STROKE DEVICE DATA 186
INQUIRE DEFAULT VALUATOR DEVICE DATA 186
INQUIRE DEFAULT CHOICE DEVICE DATA 187
INQUIRE DEFAULT PICK DEVICE DATA 188
INQUIRE DEFAULT STRING DEVICE DATA 188

A.2.8.6 Inquiry functions for segment state list 5.9.7
INQUIRE SET OF ASSOCIATED WORKSTATIONS 189
INQUIRE SEGMENT ATTRIBUTES 189

A.2.8.7 Pixel inquiries 5.9.8
INQUIRE PIXEL ARRAY DIMENSIONS 190
INQUIRE PIXEL ARRAY 191
INQUIRE PIXEL 191
Appendix A

A.2.8.8 Inquiry function for GKS error state list 5.9.9
INQUIRE INPUT QUEUE OVERFLOW 192

A.2.9 Utility functions 5.10
EVALUATE TRANSFORMATION MATRIX 193
ACCUMULATE TRANSFORMATION MATRIX 193

A.2.10 Error handling 5.11
EMERGENCY CLOSE GKS 195
ERROR HANDLING 195
ERROR LOGGING 196

A.9 Ordered by level

A.8.1 Level ma
ACTIVATE WORKSTATION 76
CLEAR WORKSTATION 76
CLOSE GKS 74
CLOSE WORKSTATION 75
DEACTIVATE WORKSTATION 76
ESCAPE 80
FILL AREA 83
INQUIRE CLIPPING 151
INQUIRE COLOUR FACILITIES 180
INQUIRE COLOUR REPRESENTATION 164
INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES 149
INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER 149
INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES 148
INQUIRE DISPLAY SPACE SIZE 171
INQUIRE FILL AREA FACILITIES 177
INQUIRE LEVEL OF GKS 145
INQUIRE LIST OF COLOUR INDICES 164
INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES 182
INQUIRE NORMALIZATION TRANSFORMATION 150
INQUIRE POLYLINE FACILITIES 173
INQUIRE POLYMARKER FACILITIES 175
INQUIRE TEXT EXTENT 158
INQUIRE TEXT FACILITIES 176
INQUIRE WORKSTATION CONNECTION AND TYPE 153
INQUIRE WORKSTATION TRANSFORMATION 165
OPEN GKS 74
OPEN WORKSTATION 74
POLYLINE 82
POLYMARKER 82
SELECT NORMALIZATION TRANSFORMATION 108
SET CHARACTER HEIGHT 95
SET CHARACTER UP VECTOR 95
SET CLIPPING INDICATOR 108
SET COLOUR REPRESENTATION 105
SET FILL AREA COLOUR INDEX 98
SET FILL AREA INTERIOR STYLE 96
SET LINETYPE 89
SET MARKER TYPE 91
SET POLYLINE COLOUR INDEX 90
SET POLYMARKER COLOUR INDEX 92
SET TEXT ALIGNMENT 96
SET TEXT COLOUR INDEX 94
SET VIEWPORT 107
SET WINDOW 107
SET WORKSTATION VIEWPORT 109
SET WORKSTATION WINDOW 109
TEXT 83
UPDATE WORKSTATION 78

A.9.2 Level mb

INITIALISE CHOICE 124
INITIALISE LOCATOR 119
INITIALISE STRING 127
INITIALISE STROKE 121
INITIALISE VALUATOR 123
INQUIRE CHOICE DEVICE STATE 168
INQUIRE DEFAULT CHOICE DEVICE DATA 187
INQUIRE DEFAULT LOCATOR DEVICE DATA 185
INQUIRE DEFAULT STRING DEVICE DATA 188
INQUIRE DEFAULT STROKE DEVICE DATA 186
INQUIRE DEFAULT VALUATOR DEVICE DATA 186
INQUIRE LOCATOR DEVICE STATE 166
INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES 184
INQUIRE STRING DEVICE STATE 170
INQUIRE STROKE DEVICE STATE 167
INQUIRE VALUATOR DEVICE STATE 168
REQUEST CHOICE 133
REQUEST LOCATOR 131
REQUEST STRING 134
REQUEST STROKE 132
REQUEST VALUATOR 133
SET CHOICE MODE 130
SET LOCATOR MODE 128
SET STRING MODE 131
SET STROKE MODE 129
SET VALUATOR MODE 129
SET VIEWPORT INPUT PRIORITY 107

A.9.3 Level mc

AWAIT EVENT 138
FLUSH DEVICE EVENTS 139
GET CHOICE 140
GET LOCATOR 139
GET STRING 141
GET STROKE 139
GET VALUATOR 140
INQUIRE INPUT QUEUE OVERFLOW 192
INQUIRE MORE SIMULTANEOUS EVENTS 152
SAMPLE CHOICE 136
SAMPLE LOCATOR 135
SAMPLE STRING 137
SAMPLE STROKE 135
Appendix A

SAMPLE VALUATOR

A.8.4 Level 0a

CELL ARRAY
EMERGENCY CLOSE GKS
ERROR HANDLING
ERROR LOGGING
GENERALIZED DRAWING PRIMITIVE (GDP)
GET ITEM TYPE FROM GKS
INQUIRE GENERALIZED DRAWING PRIMITIVE
INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES
INQUIRE LIST OF AVAILABLE WORKSTATION TYPES
INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS
INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER
INQUIRE OPERATING STATE VALUE
INQUIRE PATTERN FACILITIES
INQUIRE PIXEL
INQUIRE PIXEL ARRAY
INQUIRE PIXEL ARRAY DIMENSIONS
INQUIRE PREDEFINED COLOUR REPRESENTATION
INQUIRE PREDEFINED FILL AREA REPRESENTATION
INQUIRE PREDEFINED PATTERN REPRESENTATION
INQUIRE PREDEFINED POLYLINE REPRESENTATION
INQUIRE PREDEFINED POLYMARKER REPRESENTATION
INQUIRE PREDEFINED TEXT REPRESENTATION
INQUIRE SET OF OPEN WORKSTATIONS
INQUIRE WORKSTATION CATEGORY
INQUIRE WORKSTATION CLASSIFICATION
INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES
INQUIRE WORKSTATION STATE
INTERPRET ITEM
READ ITEM FROM GKS
SET ASPECT SOURCE FLAGS
SET CHARACTER EXPANSION FACTOR
SET CHARACTER SPACING
SET FILL AREA INDEX
SET FILL AREA STYLE INDEX
SET LINEWIDTH SCALE FACTOR
SET MARKER SIZE SCALE FACTOR
SET PATTERN REFERENCE POINT
SET PATTERN SIZE
SET POLYLINE INDEX
SET POLYMARKER INDEX
SET TEXT FONT AND PRECISION
SET TEXT INDEX
SET TEXT PATH
WRITE ITEM TO GKS

A.8.5 Level 0b

none
### A.8.6 Level 0c

none

### A.8.7 Level 1a

<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCUMULATE TRANSFORMATION MATRIX</td>
<td>193</td>
</tr>
<tr>
<td>CLOSE SEGMENT</td>
<td>111</td>
</tr>
<tr>
<td>CREATE SEGMENT</td>
<td>111</td>
</tr>
<tr>
<td>DELETE SEGMENT</td>
<td>112</td>
</tr>
<tr>
<td>DELETE SEGMENT FROM WORKSTATION</td>
<td>112</td>
</tr>
<tr>
<td>EVALUATE TRANSFORMATION MATRIX</td>
<td>193</td>
</tr>
<tr>
<td>INQUIRE DEFAULT DEFERRAL STATE VALUES</td>
<td>173</td>
</tr>
<tr>
<td>INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES</td>
<td>184</td>
</tr>
<tr>
<td>INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES</td>
<td>172</td>
</tr>
<tr>
<td>INQUIRE FILL AREA REPRESENTATION</td>
<td>162</td>
</tr>
<tr>
<td>INQUIRE LIST OF FILL AREA INDICES</td>
<td>161</td>
</tr>
<tr>
<td>INQUIRE LIST OF PATTERN INDICES</td>
<td>162</td>
</tr>
<tr>
<td>INQUIRE LIST OF POLYLINE INDICES</td>
<td>154</td>
</tr>
<tr>
<td>INQUIRE LIST OF POLYMARKER INDICES</td>
<td>156</td>
</tr>
<tr>
<td>INQUIRE LIST OF TEXT INDICES</td>
<td>157</td>
</tr>
<tr>
<td>INQUIRE NAME OF OPEN SEGMENT</td>
<td>151</td>
</tr>
<tr>
<td>INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED</td>
<td>183</td>
</tr>
<tr>
<td>INQUIRE PATTERN REPRESENTATION</td>
<td>163</td>
</tr>
<tr>
<td>INQUIRE POLYLINE REPRESENTATION</td>
<td>155</td>
</tr>
<tr>
<td>INQUIRE POLYMARKER REPRESENTATION</td>
<td>156</td>
</tr>
<tr>
<td>INQUIRE SEGMENT ATTRIBUTES</td>
<td>189</td>
</tr>
<tr>
<td>INQUIRE SET OF ACTIVE WORKSTATIONS</td>
<td>147</td>
</tr>
<tr>
<td>INQUIRE SET OF ASSOCIATED WORKSTATIONS</td>
<td>189</td>
</tr>
<tr>
<td>INQUIRE SET OF SEGMENT NAMES IN USE</td>
<td>152</td>
</tr>
<tr>
<td>INQUIRE SET OF SEGMENT NAMES ON WORKSTATION</td>
<td>166</td>
</tr>
<tr>
<td>INQUIRE TEXT REPRESENTATION</td>
<td>157</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION MAXIMUM NUMBERS</td>
<td>146</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>80</td>
</tr>
<tr>
<td>REDRAW ALL SEGMENTS ON WORKSTATION</td>
<td>77</td>
</tr>
<tr>
<td>RENAME SEGMENT</td>
<td>111</td>
</tr>
<tr>
<td>SET DEFERRAL STATE</td>
<td>79</td>
</tr>
<tr>
<td>SET FILL AREA REPRESENTATION</td>
<td>103</td>
</tr>
<tr>
<td>SET HIGHLIGHTING</td>
<td>116</td>
</tr>
<tr>
<td>SET PATTERN REPRESENTATION</td>
<td>104</td>
</tr>
<tr>
<td>SET POLYLINE REPRESENTATION</td>
<td>100</td>
</tr>
<tr>
<td>SET POLYMARKER REPRESENTATION</td>
<td>101</td>
</tr>
<tr>
<td>SET SEGMENT PRIORITY</td>
<td>117</td>
</tr>
<tr>
<td>SET SEGMENT TRANSFORMATION</td>
<td>115</td>
</tr>
<tr>
<td>SET TEXT REPRESENTATION</td>
<td>102</td>
</tr>
<tr>
<td>SET VISIBILITY</td>
<td>116</td>
</tr>
</tbody>
</table>

### A.8.8 Level 1b

<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>INITIALISE PICK</td>
<td>126</td>
</tr>
<tr>
<td>INQUIRE CURRENT PICK IDENTIFIER VALUE</td>
<td>148</td>
</tr>
<tr>
<td>INQUIRE DEFAULT PICK DEVICE DATA</td>
<td>188</td>
</tr>
<tr>
<td>INQUIRE PICK DEVICE STATE</td>
<td>169</td>
</tr>
<tr>
<td>REQUEST PICK</td>
<td>134</td>
</tr>
<tr>
<td>SET DETECTABILITY</td>
<td>117</td>
</tr>
<tr>
<td>SET PICK IDENTIFIER</td>
<td>99</td>
</tr>
<tr>
<td>SET PICK MODE</td>
<td>130</td>
</tr>
</tbody>
</table>
Appendix A

A.9.9 Level 1c
GET PICK 140
SAMPLE PICK 137

A.9.10 Level 2a
ASSOCIATE SEGMENT WITH WORKSTATION 113
COPY SEGMENT TO WORKSTATION 113
INSERT SEGMENT 114

A.9.11 Level 2b
none

A.9.12 Level 2c
none

A.4 Ordered by state

A.4.1 Functions allowed in state GKCL
EMERGENCY CLOSE GKS 195
ERROR HANDLING 195
ERROR LOGGING 196
Inquiry functions
OPEN GKS 74

A.4.2 Functions allowed in state GKOP
ACCUMULATE TRANSFORMATION MATRIX 193
CLOSE GKS 74
EMERGENCY CLOSE GKS 195
ERROR HANDLING 195
ERROR LOGGING 196
ESCAPE 80
EVALUATE TRANSFORMATION MATRIX 193
Inquiry functions
INTERPRET ITEM 143
OPEN WORKSTATION 74
SELECT NORMALIZATION TRANSFORMATION 108
SET ASPECT SOURCE FLAGS 99
SET CHARACTER EXPANSION FACTOR 93
SET CHARACTER HEIGHT 95
SET CHARACTER SPACING 94
SET CHARACTER UP VECTOR 95
SET CLIPPING INDICATOR 108
SET FILL AREA COLOUR INDEX 98
SET FILL AREA INDEX 96
SET FILL AREA INTERIOR STYLE 96
SET FILL AREA STYLE INDEX 97
SET LINETYPE 89
SET LINEWIDTH SCALE FACTOR 90
SET MARKER SIZE SCALE FACTOR 92
SET MARKER TYPE 91
SET PATTERN REFERENCE POINT 98
SET PATTERN SIZE 93
A.4.8 Functions not allowed in state WSOP

CELL ARRAY 85
CLOSE GKS 74
CREATE SEGMENT 111
DEACTIVATE WORKSTATION 76
FILL AREA 83
GENERALIZED DRAWING PRIMITIVE (GDP) 86
INSERT SEGMENT 114
OPEN GKS 74
POLYLINE 82
POLYMARKER 82
TEXT 83
WRITE ITEM TO GKSM 142

A.4.4 Functions not allowed in state WSAC

CLOSE GKS 74
CLOSE SEGMENT 111
OPEN GKS 74

A.4.5 Functions not allowed in state SGOP

ACTIVATE WORKSTATION 76
ASSOCIATE SEGMENT WITH WORKSTATION 113
CLEAR WORKSTATION 75
CLOSE GKS 74
COPY SEGMENT TO WORKSTATION 113
CREATE SEGMENT 111
DEACTIVATE WORKSTATION 76
OPEN GKS 74

A.5 Applicability to workstation groups

Table 3 lists all GKS functions and the workstation categories to which they apply, directly or indirectly.
### Table 3. GKS functions and workstation categories to which they apply

<table>
<thead>
<tr>
<th>GKS Function</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Control functions</strong></td>
<td></td>
</tr>
<tr>
<td>OPEN GKS</td>
<td>not applicable</td>
</tr>
<tr>
<td>CLOSE GKS</td>
<td>not applicable</td>
</tr>
<tr>
<td>OPEN WORKSTATION</td>
<td>SS MO O OI I MI</td>
</tr>
<tr>
<td>CLOSE WORKSTATION</td>
<td>SS MO O OI I MI</td>
</tr>
<tr>
<td>ACTIVATE WORKSTATION</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>DEACTIVATE WORKSTATION</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>CLEAR WORKSTATION</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>REDRAW ALL SEGMENTS ON WORKSTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>UPDATE WORKSTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>SET DEFERRAL STATE</td>
<td>MO O OI</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>MO O OI I</td>
</tr>
<tr>
<td>ESCAPE</td>
<td>SS MO O OI I MI</td>
</tr>
<tr>
<td><strong>Output functions</strong></td>
<td></td>
</tr>
<tr>
<td>POLYLINE</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>POLYMARKER</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>TEXT</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>FILL AREA</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>CELL ARRAY</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>GENERALIZED DRAWING PRIMITIVE (GDP)</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td><strong>Output attributes</strong></td>
<td></td>
</tr>
<tr>
<td>SET POLYLINE INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET LINETYPE</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET LINEWIDTH SCALE FACTOR</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET POLYLINE COLOUR INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET POLYMARKER INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET MARKER TYPE</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET MARKER SIZE SCALE FACTOR</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET POLYMARKER COLOUR INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET TEXT INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET TEXT FONT AND PRECISION</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET CHARACTER EXPANSION FACTOR</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET CHARACTER SPACING</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET TEXT COLOUR INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET CHARACTER HEIGHT</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET CHARACTER UP VECTOR</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET TEXT PATH</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET TEXT ALIGNMENT</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET FILL AREA INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET FILL AREA INTERIOR STYLE</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET FILL AREA STYLE INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET FILL AREA COLOUR INDEX</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET PATTERN SIZE</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET PATTERN REFERENCE POINT</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET ASPECT SOURCE FLAGS</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET PICK IDENTIFIER</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>SET POLYLINE REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>SET POLYMARKER REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>SET TEXT REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>SET FILL AREA REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>SET PATTERN REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>SET COLOUR REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td><strong>Transformation functions</strong></td>
<td></td>
</tr>
<tr>
<td>SET WINDOW</td>
<td>SS MO O OI I</td>
</tr>
<tr>
<td>SET VIEWPORT</td>
<td>SS MO O OI I</td>
</tr>
<tr>
<td>SET VIEWPORT INPUT PRIORITY</td>
<td>SS MO O OI I</td>
</tr>
<tr>
<td>SELECT NORMALIZATION TRANSFORMATION</td>
<td>SS MO O OI I</td>
</tr>
<tr>
<td>SET CLIPPING INDICATOR</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>GKS Function</td>
<td>Applies to</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>SET WORKSTATION WINDOW</td>
<td>MO O O I</td>
</tr>
<tr>
<td>SET WORKSTATION VIEWPORT</td>
<td>MO O O I</td>
</tr>
<tr>
<td><strong>Segment functions</strong></td>
<td></td>
</tr>
<tr>
<td>CREATE SEGMENT</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>CLOSE SEGMENT</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>RENAME SEGMENT</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>DELETE SEGMENT</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>DELETE SEGMENT FROM WORKSTATION</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>ASSOCIATE SEGMENT WITH WORKSTATION</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>COPY SEGMENT TO WORKSTATION</td>
<td>(SS) MO O I</td>
</tr>
<tr>
<td>INSERT SEGMENT</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>SET SEGMENT TRANSFORMATION</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>SET VISIBILITY</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>SET HIGHLIGHTING</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>SET SEGMENT PRIORITY</td>
<td>SS MO O I</td>
</tr>
<tr>
<td>SET DETECTABILITY</td>
<td>SS MO O I</td>
</tr>
<tr>
<td><strong>Input functions</strong></td>
<td></td>
</tr>
<tr>
<td>INITIALISE LOCATOR</td>
<td>OI I</td>
</tr>
<tr>
<td>INITIALISE STROKE</td>
<td>OI I</td>
</tr>
<tr>
<td>INITIALISE VALUATOR</td>
<td>OI I</td>
</tr>
<tr>
<td>INITIALISE CHOICE</td>
<td>OI I</td>
</tr>
<tr>
<td>INITIALISE PICK</td>
<td>OI I</td>
</tr>
<tr>
<td>INITIALISE STRING</td>
<td>OI I</td>
</tr>
<tr>
<td>SET LOCATOR MODE</td>
<td>OI I</td>
</tr>
<tr>
<td>SET STROKE MODE</td>
<td>OI I</td>
</tr>
<tr>
<td>SET VALUATOR MODE</td>
<td>OI I</td>
</tr>
<tr>
<td>SET CHOICE MODE</td>
<td>OI I</td>
</tr>
<tr>
<td>SET PICK MODE</td>
<td>OI I</td>
</tr>
<tr>
<td>SET STRING MODE</td>
<td>OI I</td>
</tr>
<tr>
<td>REQUEST LOCATOR</td>
<td>OI I</td>
</tr>
<tr>
<td>REQUEST STROKE</td>
<td>OI I</td>
</tr>
<tr>
<td>REQUEST VALUATOR</td>
<td>OI I</td>
</tr>
<tr>
<td>REQUEST CHOICE</td>
<td>OI I</td>
</tr>
<tr>
<td>REQUEST PICK</td>
<td>OI I</td>
</tr>
<tr>
<td>REQUEST STRING</td>
<td>OI I</td>
</tr>
<tr>
<td>SAMPLE LOCATOR</td>
<td>OI I</td>
</tr>
<tr>
<td>SAMPLE STROKE</td>
<td>OI I</td>
</tr>
<tr>
<td>SAMPLE VALUATOR</td>
<td>OI I</td>
</tr>
<tr>
<td>SAMPLE CHOICE</td>
<td>OI I</td>
</tr>
<tr>
<td>SAMPLE PICK</td>
<td>OI I</td>
</tr>
<tr>
<td>SAMPLE STRING</td>
<td>OI I</td>
</tr>
<tr>
<td>AWAIT EVENT</td>
<td>OI I</td>
</tr>
<tr>
<td>FLUSH DEVICE EVENTS</td>
<td>OI I</td>
</tr>
<tr>
<td>GET LOCATOR</td>
<td>OI I</td>
</tr>
<tr>
<td>GET STROKE</td>
<td>OI I</td>
</tr>
<tr>
<td>GET VALUATOR</td>
<td>OI I</td>
</tr>
<tr>
<td>GET CHOICE</td>
<td>OI I</td>
</tr>
<tr>
<td>GET PICK</td>
<td>OI I</td>
</tr>
<tr>
<td>GET STRING</td>
<td>OI I</td>
</tr>
<tr>
<td><strong>Metafile functions</strong></td>
<td></td>
</tr>
<tr>
<td>WRITE ITEM TO GKSM</td>
<td>MO</td>
</tr>
<tr>
<td>GET ITEM TYPE FROM GKSM</td>
<td>MI</td>
</tr>
<tr>
<td>READ ITEM FROM GKSM</td>
<td>MI</td>
</tr>
<tr>
<td>INTERPRET ITEM</td>
<td>SS MO O I</td>
</tr>
<tr>
<td><strong>Inquiry functions</strong></td>
<td></td>
</tr>
<tr>
<td>INQUIRE OPERATING STATE VALUE</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE LEVEL OF GKS</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE LIST OF AVAILABLE WORKSTATION TYPES</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION MAXIMUM NUMBERS</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER</td>
<td>not applicable</td>
</tr>
</tbody>
</table>

Page 228
### Applicability to workstation groups

<table>
<thead>
<tr>
<th>GKS Function</th>
<th>Applies to</th>
</tr>
</thead>
<tbody>
<tr>
<td>INQUIRE SET OF OPEN WORKSTATIONS</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE SET OF ACTIVE WORKSTATIONS</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE CURRENT PICK IDENTIFIER VALUE</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE NORMALIZATION TRANSFORMATION</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE CLIPPING</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE NAME OF OPEN SEGMENT</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE SET OF SEGMENT NAMES IN USE</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE MORE SIMULTANEOUS EVENTS</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION CONNECTION AND TYPE</td>
<td>not applicable</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION STATE</td>
<td>SS MO O OI I MI</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>INQUIRE LIST OF POLYLINE INDICES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE POLYLINE REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE LIST OF POLYMARKER INDICES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE POLYMARKER REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE LIST OF TEXT INDICES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE TEXT REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE TEXT EXTENT</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE LIST OF FILL AREA INDICES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE FILL AREA REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE LIST OF PATTERN INDICES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE PATTERN REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE LIST OF COLOUR INDICES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE COLOUR REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION TRANSFORMATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE SET OF SEGMENT NAMES ON WORKSTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE LOCATOR DEVICE STATE</td>
<td>OI I</td>
</tr>
<tr>
<td>INQUIRE STROKE DEVICE STATE</td>
<td>OI I</td>
</tr>
<tr>
<td>INQUIRE VALUATOR DEVICE STATE</td>
<td>OI I</td>
</tr>
<tr>
<td>INQUIRE CHOICE DEVICE STATE</td>
<td>OI I</td>
</tr>
<tr>
<td>INQUIRE PICK DEVICE STATE</td>
<td>OI</td>
</tr>
<tr>
<td>INQUIRE STRING DEVICE STATE</td>
<td>OI</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION CATEGORY</td>
<td>SS MO O OI I MI</td>
</tr>
<tr>
<td>INQUIRE WORKSTATION CLASSIFICATION</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>INQUIRE DISPLAY SPACE SIZE</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE DEFAULT DEFERRAL STATE VALUES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE POLYLINE FACILITIES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE PREDEFINED POLYLINE REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE POLYMARKER FACILITIES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE PREDEFINED POLYMARKER REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE TEXT FACILITIES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE PREDEFINED TEXT REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE FILL AREA FACILITIES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE PREDEFINED FILL AREA REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE PATTERN FACILITIES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE PREDEFINED PATTERN REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE COLOUR FACILITIES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE PREDEFINED COLOUR REPRESENTATION</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE GENERALIZED DRAWING PRIMITIVE</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE DEFAULT LOCATOR DEVICE DATA</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE DEFAULT STROKE DEVICE DATA</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE DEFAULT VALUATOR DEVICE DATA</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE DEFAULT CHOICE DEVICE DATA</td>
<td>MO O OI</td>
</tr>
<tr>
<td>INQUIRE DEFAULT PICK DEVICE DATA</td>
<td>MO O OI</td>
</tr>
<tr>
<td>GKS Function</td>
<td>Applies to</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>INQUIRE DEFAULT STRING DEVICE DATA</td>
<td>OI I</td>
</tr>
<tr>
<td>INQUIRE SET OF ASSOCIATED WORKSTATIONS</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>INQUIRE SEGMENT ATTRIBUTES</td>
<td>SS MO O OI</td>
</tr>
<tr>
<td>INQUIRE PIXEL ARRAY DIMENSIONS</td>
<td>O OI</td>
</tr>
<tr>
<td>INQUIRE PIXEL ARRAY</td>
<td>O OI</td>
</tr>
<tr>
<td>INQUIRE PIXEL</td>
<td>O OI</td>
</tr>
<tr>
<td>INQUIRE INPUT QUEUE OVERFLOW</td>
<td>not applicable</td>
</tr>
<tr>
<td>Utility functions</td>
<td></td>
</tr>
<tr>
<td>EVALUATE TRANSFORMATION MATRIX</td>
<td>not applicable</td>
</tr>
<tr>
<td>ACCUMULATE TRANSFORMATION MATRIX</td>
<td>not applicable</td>
</tr>
<tr>
<td>Error handling</td>
<td></td>
</tr>
<tr>
<td>EMERGENCY CLOSE GKS</td>
<td>not applicable</td>
</tr>
<tr>
<td>ERROR HANDLING</td>
<td>not applicable</td>
</tr>
<tr>
<td>ERROR LOGGING</td>
<td>not applicable</td>
</tr>
</tbody>
</table>

Key:

- SS  Workstation Independent Segment Storage
- MO  workstation of category MO
- O   workstation of category OUTPUT
- OI  workstation of category OUTIN
- I   workstation of category INPUT
- MI  workstation of category MI
- (SS) Workstation Independent Segment Storage is fundamental to the operation of this GKS function, but the workstation identifier parameter cannot be Workstation Independent Segment Storage
Appendix B  Error list

NOTE This appendix is not part of the Standard, but provides additional information.

B.1 Implementation dependent

<0 Implementation dependent errors

B.2 States

1 GKS not in proper state: GKS shall be in the state GKCL
2 GKS not in proper state: GKS shall be in the state GKOP
3 GKS not in proper state: GKS shall be in the state WSAC
4 GKS not in proper state: GKS shall be in the state SGOP
5 GKS not in proper state: GKS shall be either in the state WSAC or in the state SGOP
6 GKS not in proper state: GKS shall be either in the state WSOP or in the state WSAC
7 GKS not in proper state: GKS shall be in one of the states WSOP, WSAC or SGOP
8 GKS not in proper state: GKS shall be in one of the states GKOP, WSOP, WSAC or SGOP

B.3 Workstations

20 Specified workstation identifier is invalid
21 Specified connection identifier is invalid
22 Specified workstation type is invalid
28 Specified workstation type does not exist
24 Specified workstation is open
25 Specified workstation is not open
26 Specified workstation cannot be opened
27 Workstation Independent Segment Storage is not open
28 Workstation Independent Segment Storage is already open
29 Specified workstation is active
30 Specified workstation is not active
31 Specified workstation is of category MO
32 Specified workstation is not of category MO
38 Specified workstation is of category MI
34 Specified workstation is not of category MI
35 Specified workstation is of category INPUT
36 Specified workstation is Workstation Independent Segment Storage
37 Specified workstation is not of category OUTIN
38 Specified workstation is neither of category INPUT nor of category OUTIN
39 Specified workstation is neither of category OUTPUT nor of category OUTIN
40 Specified workstation has no pixel store readback capability
41 Specified workstation type is not able to generate the specified generalized drawing primitive
42 Maximum number of simultaneously open workstations would be exceeded
48 Maximum number of simultaneously active workstations would be exceeded

B.4 Transformations

50 Transformation number is invalid
51 Rectangle definition is invalid
52 Viewport is not within the Normalized Device Coordinate unit square
58 Workstation window is not within the Normalized Device Coordinate unit square
54 Workstation viewport is not within the display space
B.5 Output attributes

60 Polyline index is invalid
61 A representation for the specified polyline index has not been defined on this workstation
62 A representation for the specified polyline index has not been predefined on this workstation
63 Linetype is equal to zero
64 Specified linetype is not supported on this workstation
65 Linewidth scale factor is less than zero
66 Polymarker index is invalid
67 A representation for the specified polymarker index has not been defined on this workstation
68 A representation for the specified polymarker index has not been predefined on this workstation
69 Marker type is equal to zero
70 Specified marker type is not supported on this workstation
71 Marker size scale factor is less than zero
72 Text index is invalid
73 A representation for the specified text index has not been defined on this workstation
74 A representation for the specified text index has not been predefined on this workstation
75 Text font is equal to zero
76 Requested text font is not supported for the specified precision on this workstation
77 Character expansion factor is less than or equal to zero
78 Character height is less than or equal to zero
79 Length of character up vector is zero
80 Fill area index is invalid
81 A representation for the specified fill area index has not been defined on this workstation
82 A representation for the specified fill area index has not been predefined on this workstation
83 Specified fill area interior style is not supported on this workstation
84 Style (pattern or hatch) index is equal to zero
85 Specified pattern index is invalid
86 Specified hatch style is not supported on this workstation
87 Pattern size value is not positive
88 A representation for the specified pattern index has not been defined on this workstation
89 A representation for the specified pattern index has not been predefined on this workstation
90 Interior style PATTERN is not supported on this workstation
91 Dimensions of colour array are invalid
92 Colour index is less than zero
93 Colour index is invalid
94 A representation for the specified colour index has not been defined on this workstation
95 A representation for the specified colour index has not been predefined on this workstation
96 Colour is outside range \([0,1]\)
97 Pick identifier is invalid

B.6 Output primitives

100 Number of points is invalid
101 Invalid code in string
102 Generalized drawing primitive identifier is invalid
103 Content of generalized drawing primitive data record is invalid
104 At least one active workstation is not able to generate the specified generalized drawing primitive
105 At least one active workstation is not able to generate the specified generalized drawing primitive under the current transformations and clipping rectangle
Appendix B

B.7 Segments

120 Specified segment name is invalid
121 Specified segment name is already in use
122 Specified segment does not exist
123 Specified segment does not exist on specified workstation
124 Specified segment does not exist on Workstation Independent Segment Storage
125 Specified segment is open
126 Segment priority is outside the range [0,1]

B.8 Input

140 Specified input device is not present on workstation
141 Input device is not in REQUEST mode
142 Input device is not in SAMPLE mode
143 EVENT and SAMPLE input mode are not available at this level of GKS
144 Specified prompt and echo type is not supported on this workstation
145 Echo area is outside display space
146 Contents of input data record are invalid
147 Input queue has overflowed
148 Input queue has not overflowed since GKS was opened or the last invocation of INQUIRE
149 Input queue has overflowed, but associated workstation has been closed
150 No input value of the correct class is in the current event report
151 Timeout is invalid
152 Initial value is invalid
153 Number of points in the initial stroke is greater than the buffer size
154 Length of the initial string is greater than the buffer size

B.9 Metafiles

160 Item type is not allowed for user items
161 Item length is invalid
162 No item is left in GKS Metafile input
163 Metafile item is invalid
164 Item type is not a valid GKS item
165 Content of item data record is invalid for the specified item type
166 Maximum item data record length is invalid
167 User item cannot be interpreted
168 Specified function is not supported in this level of GKS

B.10 Escape

180 Specified escape function is not supported
181 Specified escape function identification is invalid
182 Contents of escape data record are invalid

B.11 Miscellaneous

200 Specified error file is invalid

B.12 System

800 Storage overflow has occurred in GKS
801 Storage overflow has occurred in segment storage
802 Input/Output error has occurred while reading
803 Input/Output error has occurred while writing
804 Input/Output error has occurred while sending data to a workstation
805 Input/Output error has occurred while receiving data from a workstation
306 Input/Output error has occurred during program library management
307 Input/Output error has occurred while reading workstation description table
308 Arithmetic error has occurred

B.18 Reserved errors
Unused error numbers less than 2000 are reserved for future standardization.

Error numbers 2000-3999 are reserved for language bindings.

Error numbers greater than or equal to 4000 are reserved for registration or future standardization.

NOTE. Error numbers are registered in the ISO International Register of Graphical Items, which is maintained by the Registration Authority. When an error has been approved by the ISO Working Group on Computer Graphics \(^1\), the error number will be assigned by the Registration Authority.

\(^1\) Currently, the Working Group on Computer Graphics is known as ISO/TC97/SC5/WG2.
This appendix is not part of the standard, but provides additional information.

C.1 Introduction GKS is described in abstract terms, in order that it may be useful to applications in a wide range of environments such as programming languages and communication protocols. Before it can be used by a particular application program written in a particular programming language (host language), two further stages of specification are required:

a) language binding: the abstract functions and data types of GKS must be instantiated in terms of the constructs available in the host language;

b) implementation: this set of language specific facilities must then be provided using the facilities of a particular machine and operating system.

C.2 Language Binding

A GKS language binding is a document describing how GKS functions are accessed by programs written in a specific language. The following guidelines should be observed when binding GKS to a host language. The object of a binding is to provide the functions and data types of GKS in a natural and efficient manner using the facilities of the host language, without violating the style or design philosophy of the language.

Rule L1: GKS functionalities may be partitioned in a binding so long as this partitioning does not violate the GKS requirement that the GKS operating state or any value in any state list must not be in an ambiguous state between function invocations.

This guideline allows the binding to map single GKS abstract functions into sequences of language functions called by the application program, as long as the state of GKS is well-defined between each step in the sequence. This does not imply that such factoring should be done for its own sake, but only that it is allowed when other considerations make it desirable.

Rule L2: GKS functions should be bound in such a manner that the functionality of GKS could be, wherever possible, extended without requiring alterations to existing application programs.

The intent of this guideline is to allow possible future extensions of GKS without making obsolete applications based on GKS. An example is the addition of new attributes and the impact of this extension on setting and inquiring bundle contents.

Rule L3: The binding of GKS functions should observe the principles of good human factors engineering.

Function names should be easy to remember and associate with their functionality. In a subroutine binding, the maximum number of arguments per routine should be a reasonably small number, perhaps ten.

Rule L4: The language binding should specify a set of function identifiers acceptable to the language.

The names used for GKS functions in the standard are merely tools for describing the semantics of the standard; they should be replaced by actual identifiers conforming to the restrictions of the host language. A one-to-one mapping from abstract functions to language functions is preferred when no other considerations apply.

Rule L5: The language binding should specify, for each of the GKS data types, a corresponding data type acceptable to the language. Where convenient for the host language, additional data types may be specified in terms of the GKS data types.

The data types used in the standard are merely tools for describing the semantics of standard; they should be replaced by actual data types conforming to the restrictions of the host language.
Rule L6: The language binding should specify, for each GKS abstract function, how the
   corresponding language function or functions are to be invoked, and the means
   whereby each of the abstract input parameters is transmitted to the language func-
   tion and the means whereby each of the abstract output parameters is received
   from the language function.

   Where the host language allows, the abstract functions will be mapped onto language functions
   or procedures. The parameters will typically be transmitted via a parameter list. The items in
   such a list may either be, or be references to, items of the data types corresponding to the GKS
   data types, or aggregates of these types.

Rule L7: If the scope rules of the host language are not sufficient to restrict the visibility of
   identifiers outside a GKS implementation, then the language binding should specify
   a set of identifiers, acceptable to the language, which may be used by an implementa-
   tion for internal communication.

   An implementation may be unable to restrict its use of externally visible identifiers to those
   specified as a consequence of guidelines L4-L6. Applications should, therefore, avoid using
   identifiers from the set specified by guideline L7. The set may consist, for example, of all
   identifiers beginning with 'GKS.'

Rule L8: The identifiers and data types specified in a language binding should be similar to
   those in other language bindings if the languages are similar.

   It would be detrimental to inter-language and programmer portability if each language binding
   were designated from scratch with no consideration of existing language bindings. While it is
   valuable to take advantage of the capabilities of the host language, bindings to similar languages
   can share many common aspects. For example, two languages that both restrict function
   identifiers to six alphanumeric characters should use the same set of function identifiers for
   GKS functions.

Rule L9: ANS GKS functions should be bound in such a manner that the language binding
   is compatible with the ISO 7942 specification of GKS.

   The language binding should not needlessly incorporate features which can create incompatibili-
   ties with the ISO 7942 specification. For example, the enumerated type corresponding to the
   levels of ISO GKS should match those for ANSI GKS.

C.8 Implementation

One form of GKS implementation is a module or library of modules written for a specific pro-
gramming language and conforming to a GKS language binding. The following guidelines
should be observed when implementing GKS. The objective is to provide all the functions of a
particular level of GKS in an efficient manner using the facilities available from the host
machine and operating system.

Rule II: The documentation of a GKS implementation should include a list of all identifiers
   for procedures, functions, global data aggregates, and files that are visible either to
   an application program or to the underlying operating system.

   Because this set of identifiers is, in general, a superset of the names specified by the language
   binding, programs transported to an implementation from other implementations of the same
   binding might have used names that clash. Documentation is required to enable potential
   clashes to be detected (see also rule L7).

Rule II: Implementations should minimize restrictions of an application program’s use of
   non-graphical I/O facilities provided by the host language or operating system. However, a GKS implementation can assume (but need not guarantee) that it has
exclusive control over the non-graphical and graphical resources it is managing.
   The documentation should include a list of all resources assumed to be reserved for
exclusive use by the implementation.
Non-graphical resources reserved by a GKS implementation might include logical unit numbers used to identify graphics devices, and a file for error messages. If other processes outside GKS use a graphics device, such as for output of messages from the computer operator, the results are undefined. Since restricting such use is often impossible, it is not required. This guideline allows an implementation to leave graphics devices in graphics mode between functions, for example.

Rule 13: The documentation of an implementation should specify, for each of the implementation and workstation dependencies, how the dependencies have been resolved.

Several details of the standard have been deliberately unspecified so as to provide implementors with sufficient freedom to adapt to particular computers and operating systems. These are indicated in the text by the words "implementation dependent". Others have been left unspecified to allow for adaptation to particular graphics devices. These are indicated in the text by the words "workstation dependent". A list of all such details is given as Appendix D. The resolution of each of these details should be documented so that the behaviour of application programs may be predicted.

Rule 14: The documentation of each workstation of an implementation/installation should specify the correspondence between physical input devices and operator actions, and the logical input devices on that workstation (if any).

The correspondence between physical input devices and operator actions, and the logical input devices on a workstation may be static, and not under the control of the application program. These correspondences need to be documented. It is desirable that workstation implementors provide means whereby these correspondences may be changed, perhaps during a GKS configuration phase. However, any such means lie outside the scope of the standard.

Rule 15: The documentation of a standard implementation should specify the highest level of GKS supported and list any extensions beyond that level. These extensions should be further identified as extensions defined at a higher level of GKS or extensions not defined at any level of GKS.

Extensions must be clearly identified so they can be avoided by application programs that require portability to different implementations. It is preferable that the escape and GDP functions be used to implement non-standard extensions where possible. Extensions not defined in GKS shall not interfere with the correct execution of functions within GKS.
Appendix D  Allowable differences in GKS implementations

NOTE. This appendix is not part of the Standard, but provides additional information.

D.1 Introduction
A number of details of GKS are deliberately not specified, so as to provide the freedom to
adapt implementations to different environments and different requirements. In particular,
GKS is described in abstract terms, so that it can be useful to application programs written in a
wide range of programming languages. In a language binding, the abstract GKS functions are
embedded in a language dependent layer, according to a number of rules. These rules are set
out in Appendix C and are not considered further here.

Other allowable differences fall into two categories:
   a) global differences;
   b) workstation dependent differences.

The purpose of this Appendix is to itemise these allowable differences. The documentation
accompanying a particular implementation needs to list, for each allowable difference, the
specific choices made in that implementation.

D.2 Global differences
A number of differences are global in the sense of applying to an implementation as a whole
rather than to a particular workstation. These global differences are itemized below.

   a) Functional scope:
      1) GKS level.

   b) Capacity:
      1) number of available workstation types;
      2) list of available workstation types;
      3) maximum number of simultaneously open workstations;
      4) maximum number of simultaneously active workstations;
      5) maximum number of workstations associated with a segment;
      6) maximum normalization transformation number;
      7) number of simultaneously definable segments (per workstation);
      8) maximum size of input queue;
      9) number of fonts available;
      10) number of GDP's;
      11) number of ESCAPE functions.

   c) Miscellaneous:
      1) EMERGENCY CLOSE GKS behaviour;
      2) actions performed on parameters of inquiry functions if information is unavailable;
      3) metafile format used by each workstation type of category MO;
      4) font definitions (with the restriction that font numbers greater than 1 may be
         registered);
      5) Internal format of data records

NOTE 1 Items in a) and b) 1) to b) 6) are held in the GKS description table, and can be inquired by an application
program

NOTE 2 As well as specifying 'maximum size of input queue', the documentation needs to specify its interpretation
Globa! differences

(Note 3. At different GKS levels, certain minimum capabilities are defined in 4.10.

D.8 Workstation dependent differences

This group of allowable differences provides for a range of workstations to be used in a GKS implementation. The major group of differences are listed as the workstation description table, specified in 6.6, which forms part of the GKS data structures. Entries in this table may be inquired by an application program.

There are restrictions, however, on the values of some entries; at different GKS levels, certain minimal capabilities of a workstation are defined (see 4.10).

In addition, a number of further workstation dependent differences are listed here:

a) Control functions:
   1) Realization of GKS functions: MESSAGE, ESCAPE;
   2) Buffering of deferred actions in deferral modes BNIL, BNIG and ASTI.

b) Output functions and attributes:
   1) POLYLINE
      i) whether linetype is continuous or restarted, at the start of a polyline, at the start of a clipped piece of a polyline and at each vertex of a polyline;
      ii) graphical representation of available linetypes (with the restrictions that linetypes 1 to 4 need to be recognizable as solid, dashed, dotted and dashed-dotted, linetypes greater than 4 may be registered and linetypes less than 0 need to have similar appearance on all workstations on which they are available);
      iii) the shape of the ends of lines for certain values of the linewidth scale factor aspect;
   2) POLYMARKER
      i) graphical representation of available marker types (with the restrictions that marker types 1 to 5 need to be recognizable as dot, plus sign, asterisk, circle and diagonal cross, marker types greater than 5 may be registered and marker types less than 0 need to have similar appearance on all workstations on which they are available);
      ii) clipping of markers, whose position is just inside a clipping boundary;
   3) TEXT
      i) clipping of STRING and CHAR precision text;
      ii) for STRING precision, how current settings of the text aspects are taken into account;
      iii) for CHAR precision, evaluation of the aspects character expansion factor, CHARACTER HEIGHT, and CHARACTER UP VECTOR;
      iv) the effect of control characters in the character string;
   4) FILL AREA
      i) graphical representation of available hatch styles (with the restriction that hatch styles greater than 0 may be registered and hatch styles less than 0 need to have similar appearance on all workstations on which they are available);
      ii) whether patterns and hatching are affected by transformations;
      iii) linetype and linewidth for interior style HOLLOW;
5) CELL ARRAY
   i) whether CELL ARRAY is fully supported or sometimes simulated and, if the
      latter, the simulation (minimal action required is to draw the transformed boundaries
      of the cell rectangle, using implementation dependent colour, linetype and linewidth);

6) GDP
   i) realization of each GDP;

7) all primitives
   i) colour index used if an output primitive is displayed with a colour index that is not
      present in the colour table;
   ii) on monochrome workstations, algorithm for mapping (red, green, blue) colour
       values to intensity. A recommended algorithm is:

       \[
       \text{intensity} = 0.30 \times \text{red} + 0.59 \times \text{green} + 0.11 \times \text{blue}
       \]

       as in U.S. colour television systems (NTSC encoding). The resulting intensity is
       mapped into the nearest available.
   iii) what is drawn when points are collinear or points or lines coincide (see 4.5.3).
       (All primitives except POLYMARKER.)

c) Segments:
   1) picking segments of equal priority;
   2) display of overlapping segments of equal priority;
   3) realization of highlighting.

d) Input functions:
   1) realization of logical input devices (for each logical input device, its measure and
      trigger need to be described in terms of the physical devices available on a workstation);
   2) default prompt and echo type realization;
   3) use of input data record for optional parameters.

e) Inquiry
   1) values returned by INQUIRE TEXT EXTENT;
   2) values returned by PIXEL inquiry functions;
   3) answers returned by inquiry when the REALIZED flag is set.
Appendix E  Metafile structure

NOTE  This annex does not form an integral part of the standard, but provides additional information

E.1  Metafiles

E.1.1  Introduction

In section 1, it states that GKS 'includes functions for storage on and retrieval from an external graphics file'. This external file is called a graphics metafile or metafile. GKS metafiles can be used for a variety of purposes (as stated in 4.9);

a) transporting graphical information between systems;

b) transporting graphical information from one place to another;

c) transporting graphical information from one GKS application to another;

d) storing accompanying non-graphical information.

These purposes cover different aims including picture capture, structured picture capture and session capture, the latter also being called audit trail. The graphical information needed for these aims corresponds to different types of metafile. For picture capture, some output-related functions (such as those that manipulate segments) may be recorded as the resulting set of primitives and their attributes. For structured picture capture or session capture, all functions that GKS sends to the workstation need to be recorded.

The encoding mechanism used for writing the metafile may depend on the application or environment, for example:

  e) encoding within the rules of ISO 2022 to enable network transfer;

  f) binary encoding to ease storage on a machine for later use on that machine or to minimize the processing requirement;

  g) clear text encoding to enable transfer between highly different computer architectures and easy editing.

The specification of the format and content of a metafile is not part of GKS. GKS only specifies the interface to the metafile. An implementation of GKS may support any number of workstation types of category MI or MO. The user may select the most appropriate of these depending on the application and environment. Two metafiles are outlined in E.1.2 and E.1.3.

E.1.2  ISO 8632

This metafile (ISO 8632 Information Processing Systems - Computer Graphics - Metafile for Transfer and Storage of Picture Description Information) may be categorized as one which aims to provide a means of recording pictures using metafile elements compatible with Level 0a of GKS. It is suitable for picture capture but less suitable for session capture or structured picture capture, the latter not being possible.

The metafile permits a variety of encoding schemes to be used from a single abstract metafile specification. The encodings that are included are:

  a) a character encoding based on ISO 2022 code extension procedures;

  b) a binary encoding based on IEEE (Draft 810 Task P754) floating point formats;

  c) a clear text encoding.

Other standardized encodings may be added in time; private encodings based on the abstract structure and the rules of conformance given in the standard are also allowed.

1) In course of preparation
E.1.8 Metafile designed for GKS

This metafile may be categorized as one which aims to provide a means of recording the exact sequence of function calls made to a GKS workstation. Its functional capability covers the entire range of GKS output functions, from level 0 to level 2. It is suitable for picture capture, structured picture capture or session capture. It is particularly suitable for transporting graphical information from one GKS application to another and for applications where the individual graphics actions need to be replayed, with optional editing.

Two encodings are specified for this metafile. They are:

a) a clear text encoding;

b) an unspecified binary format.

This metafile is described in the following subsections.

E.2 File format and data format

The GKS metafile is built up as a sequence of logical data items. The file starts with a file header in fixed format which describes the origin of the metafile (author, installation), the format of the following items, and the number representation. The file ends with an end item indicating the logical end of the file. In between these two items the following information is recorded in the sense of an audit trail:

a) workstation control items and message items;

b) output primitive items, describing elementary graphics objects;

c) attribute information, including output primitive attributes, segment attributes, and workstation attributes;

d) segment items, describing the segment structure and dynamic segment manipulations;

e) user items.

The overall structure of the GKS metafile is as follows:

FILE:  

| file header | item | ... | item data record | ... | item | N | end item |

ITEM:  

| item header | item data record |

ITEM HEADER:  

| 'GKSM' (optional) | item type identification number | length of item data record in bytes |

All data items except the file header have an item header containing:

f) the character string ‘GKSM’ (optional) which is present to improve legibility of the file and to provide an error control facility;

g) the item type identification number which indicates the kind of information that is contained in the item;

h) the length of the item data record.

The lengths of these fields of the item header are implementation dependent and are specified in the file header. The content of the item data record is fully described for each item type later in this annex.

The metafile contains characters, integer numbers, and real numbers marked (c), (i), (r) in the item description. Characters in the metafile are represented according to ISO 646 and ISO
Numbers are represented according to ISO 6093 using format F1 for integers and format F2 for reals.

**NOTE** Formats F1 and F2 can be written and read via FORTRAN formats I and F respectively.

Real numbers describing coordinates and length units are stored as normalized device coordinates. The workstation transformation, if specified in the application program for a workstation writing a metafile of this format, is not performed but WORKSTATION WINDOW and WORKSTATION VIEWPORT are stored in data items for later use. Real numbers may be stored as integers. In this case transformation parameters are specified in the file header to allow proper transformation of integers into normalized device coordinates.

For reasons of economy, numbers can be stored using an internal binary format. As no standard exists for binary number representation, this format limits the portability of the metafile. The specification of such a binary number representation is outside the scope of this standard.

When exchanging metafiles between different installations, the physical structure of data sets on specific storage media should be standardised. Such a specification is outside the scope of this standard.

### E.9 Generation of metafiles

Table 4 contains a list, by class, of all GKS functions which apply to workstations of category MO, and their effects on this GKSM. In the table, GKSM-OUT is a workstation identifier indicating a workstation writing a metafile of this format.

The concepts of clipping rectangle and clipping indicator are encapsulated in one metafile item which specifies a clipping rectangle. This item is written to the metafile on activate workstation with the values (0,1,0,1), if the 'clipping indicator' entry in the GKS state list is NOCLIP, or the clipping rectangle in the GKS state list if the 'clipping indicator' entry in the GKS state list is CLIP. If the clipping rectangle in the GKS state list is redefined when the 'clipping indicator' entry in the GKS state list is CLIP, a further clipping rectangle item is written. If the 'clipping indicator' entry in the GKS state list is changed to NOCLIP, a clipping rectangle item (0,1,0,1) is written. If the 'clipping indicator' entry in the GKS state list is changed to CLIP, an item containing the clipping rectangle in the GKS state list is written. This is analogous to the handling of clipping in segments (see 4.7.6).
### Table 4. GKS functions and their effect on GKSM output workstations

<table>
<thead>
<tr>
<th>GKS functions which apply to workstations of category MO</th>
<th>GKSM item created or effect</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Control functions</strong></td>
<td></td>
</tr>
<tr>
<td>OPEN WORKSTATION (GKSM-OUT,...)</td>
<td>- (file header)</td>
</tr>
<tr>
<td>CLOSE WORKSTATION (GKSM-OUT)</td>
<td>1 (CONDITIONAL)</td>
</tr>
<tr>
<td>ACTIVATE WORKSTATION (GKSM-OUT)</td>
<td>0 (end item)</td>
</tr>
<tr>
<td>DEACTIVATE WORKSTATION (GKSM-OUT)</td>
<td>61, 21-44</td>
</tr>
<tr>
<td>CLEAR WORKSTATION (GKSM-OUT,...)</td>
<td>ensure attributes current; enable output</td>
</tr>
<tr>
<td>REDRAW ALL SEGMENTS ON WORKSTATION (GKSM-OUT)</td>
<td>disable output</td>
</tr>
<tr>
<td>UPDATE WORKSTATION (GKSM-OUT,...)</td>
<td>1</td>
</tr>
<tr>
<td>SET DEFERRAL STATE (GKSM-OUT,...)</td>
<td>2</td>
</tr>
<tr>
<td>MESSAGE (GKSM-OUT,...)</td>
<td>3</td>
</tr>
<tr>
<td>ESCAPE</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>5 (message)</td>
</tr>
<tr>
<td></td>
<td>6</td>
</tr>
<tr>
<td><strong>Output primitives</strong></td>
<td></td>
</tr>
<tr>
<td>POLYLINE</td>
<td>11</td>
</tr>
<tr>
<td>POLYMARKER</td>
<td>12</td>
</tr>
<tr>
<td>TEXT</td>
<td>13</td>
</tr>
<tr>
<td>FILL AREA</td>
<td>14</td>
</tr>
<tr>
<td>CELL ARRAY</td>
<td>15</td>
</tr>
<tr>
<td>GENERALIZED DRAWING PRIMITIVE</td>
<td>16</td>
</tr>
<tr>
<td><strong>Output attributes</strong></td>
<td></td>
</tr>
<tr>
<td>SET POLYLINE INDEX</td>
<td>21</td>
</tr>
<tr>
<td>SET LINETYPE</td>
<td>22</td>
</tr>
<tr>
<td>SET LINEWIDTH SCALE FACTOR</td>
<td>23</td>
</tr>
<tr>
<td>SET POLYLINE COLOUR INDEX</td>
<td>24</td>
</tr>
<tr>
<td>SET POLYMARKER INDEX</td>
<td>25</td>
</tr>
<tr>
<td>SET MARKER TYPE</td>
<td>26</td>
</tr>
<tr>
<td>SET MARKER SIZE SCALE FACTOR</td>
<td>27</td>
</tr>
<tr>
<td>SET POLYMARKER COLOUR INDEX</td>
<td>28</td>
</tr>
<tr>
<td>SET TEXT INDEX</td>
<td>29</td>
</tr>
<tr>
<td>SET TEXT FONT AND PRECISION</td>
<td>30</td>
</tr>
<tr>
<td>SET CHARACTER EXPANSION FACTOR</td>
<td>31</td>
</tr>
<tr>
<td>SET CHARACTER SPACING</td>
<td>32</td>
</tr>
<tr>
<td>SET TEXT COLOUR INDEX</td>
<td>33</td>
</tr>
<tr>
<td>SET CHARACTER HEIGHT</td>
<td>34</td>
</tr>
<tr>
<td>SET CHARACTER UP VECTOR</td>
<td>34</td>
</tr>
<tr>
<td>SET TEXT PATH</td>
<td>35</td>
</tr>
<tr>
<td>SET TEXT ALIGNMENT</td>
<td>36</td>
</tr>
<tr>
<td>SET FILL AREA INDEX</td>
<td>37</td>
</tr>
<tr>
<td>SET FILL AREA INTERIOR STYLE</td>
<td>38</td>
</tr>
<tr>
<td>SET FILL AREA STYLE INDEX</td>
<td>39</td>
</tr>
<tr>
<td>SET FILL AREA COLOUR INDEX</td>
<td>40</td>
</tr>
<tr>
<td>SET PATTERN SIZE</td>
<td>41</td>
</tr>
<tr>
<td>SET PATTERN REFERENCE POINT</td>
<td>42</td>
</tr>
<tr>
<td>SET ASPECT SOURCE FLAGS</td>
<td>43</td>
</tr>
</tbody>
</table>
### GKS functions which apply to workstations of category MO

<table>
<thead>
<tr>
<th>GKS function</th>
<th>GKSM Item created or effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET PICK IDENTIFIER</td>
<td>44</td>
</tr>
</tbody>
</table>

#### Workstation attributes

<table>
<thead>
<tr>
<th>Function</th>
<th>Item created or effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET POLYLINE REPRESENTATION (GKSM-OUT,...)</td>
<td>51</td>
</tr>
<tr>
<td>SET POLYMARKER REPRESENTATION (GKSM-OUT,...)</td>
<td>52</td>
</tr>
<tr>
<td>SET TEXT REPRESENTATION (GKSM-OUT,...)</td>
<td>53</td>
</tr>
<tr>
<td>SET FILL AREA REPRESENTATION (GKSM-OUT,...)</td>
<td>54</td>
</tr>
<tr>
<td>SET PATTERN REPRESENTATION (GKSM-OUT,...)</td>
<td>55</td>
</tr>
<tr>
<td>SET COLOUR REPRESENTATION (GKSM-OUT,...)</td>
<td>56</td>
</tr>
</tbody>
</table>

#### Transformation functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Item created or effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET WINDOW of current normalization transformation (see note 2)</td>
<td>34, 41, 42</td>
</tr>
<tr>
<td>SET VIEWPORT of current normalization transformation (see notes 1 and 2)</td>
<td>61, 34, 41, 42</td>
</tr>
<tr>
<td>SELECT NORMALIZATION TRANSFORMATION (see notes 1 and 2)</td>
<td>61, 34, 41, 42</td>
</tr>
<tr>
<td>SET CLIP INDICATOR (see note 1)</td>
<td>61</td>
</tr>
<tr>
<td>SET WORKSTATION WINDOW (GKSM-OUT,...)</td>
<td>71</td>
</tr>
<tr>
<td>SET WORKSTATION VIEWPORT (GKSM-OUT,...)</td>
<td>72</td>
</tr>
</tbody>
</table>

#### Segment functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Item created or effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>CREATE SEGMENT</td>
<td>81</td>
</tr>
<tr>
<td>CLOSE SEGMENT</td>
<td>82</td>
</tr>
<tr>
<td>RENAME SEGMENT</td>
<td>83</td>
</tr>
<tr>
<td>DELETE SEGMENT</td>
<td>84</td>
</tr>
<tr>
<td>DELETE SEGMENT FROM WORKSTATION (GKSM-OUT,...)</td>
<td>84</td>
</tr>
<tr>
<td>ASSOCIATE SEGMENT WITH WORKSTATION (GKSM-OUT,...)</td>
<td>81, (91-95), (21-44), (11-16), (61), 82</td>
</tr>
<tr>
<td>COPY SEGMENT TO WORKSTATION (GKSM-OUT,...)</td>
<td>(21-44), (11-16), (61)</td>
</tr>
<tr>
<td>INSERT SEGMENT</td>
<td>(21-44), (11-16), (61)</td>
</tr>
</tbody>
</table>

#### Segment attributes

<table>
<thead>
<tr>
<th>Function</th>
<th>Item created or effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET SEGMENT TRANSFORMATION</td>
<td>91</td>
</tr>
<tr>
<td>SET VISIBILITY</td>
<td>92</td>
</tr>
<tr>
<td>SET HIGHLIGHTING</td>
<td>93</td>
</tr>
<tr>
<td>SET SEGMENT PRIORITY</td>
<td>94</td>
</tr>
<tr>
<td>SET DETECTABILITY</td>
<td>95</td>
</tr>
</tbody>
</table>

#### Metafile functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Item created or effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRITE ITEM TO GKSM</td>
<td>&gt;100</td>
</tr>
</tbody>
</table>

**NOTE 1.** Item 61 (CLIPPING RECTANGLE) is described more fully in E.2.

**NOTE 2.** When the current normalization transformation is altered, items corresponding to attributes containing coordinate information are sent (items 34, 41 and 42).
E.4 Interpretation of metafiles

E.4.1 Introduction
The interpretation of metafiles in GKS is described in 4.9. The effects of INTERPRET ITEM for all types of metafile item are described in E.4.2 to E.4.8. Items are grouped by class as in table 4.

E.4.2 Control items
Interpretation of items in this class is described under the definitions of each item in E.5.

E.4.3 Output primitives
Interpretation of items in this class generates output corresponding to the primitive functions, except that coordinates of points are expressed in NDC. Output primitives have bound to them the appropriate primitive attributes from the GKS state list.

E.4.4 Output primitive attributes
Interpretation of items in this class sets entries in the GKS state list. The geometric attribute information, which is expressed in NDC, is transformed by the inverse of the currently selected normalization transformation before being used to set these entries. Interpretation of the character vectors item causes the two vectors to be thus transformed and the entries in the GKS state list to be set as follows. The 'current character height' entry is set to the length of the transformed character height vector and the 'current character up vector' entry is set to a vector of arbitrary length, parallel to the transformed character height vector. The 'current character width' and 'current character base vector' are similarly set using the transformed character width vector.

E.4.5 Workstation attributes
Interpretation of items in this class has the same effect as invocation of the corresponding GKS functions shown in table 4. The GKS functions are performed on all active workstations.

E.4.6 Transformations
Interpretation of a clipping rectangle item sets the 'clipping rectangle' entry in the GKS state list and also sets the 'clipping indicator' entry in the GKS state list to CLIP. Interpretation of other items in this class (WORKSTATION WINDOW and WORKSTATION VIEWPORT) causes the invocation of the corresponding GKS functions on all active workstations.

E.4.7 Segment manipulation
Interpretation of items in this class has the same effect as invocation of the corresponding GKS functions shown in table 4. (Item 84 causes an invocation of DELETE SEGMENT.)

E.4.8 Segment attributes
Interpretation of items in this class has the same effect as invocation of the corresponding GKS functions shown in table 4.

E.5 Control items
FILE HEADER

| GKSM | N | D | V | H | T | L | I | R | F | RI | ZERO | ONE |

All fields in the file header item have fixed length. Numbers are formatted according to Format F1 of ISO 6093.
Appendix E

General information:

GKSM 4 bytes containing string 'GKSM'
N 40 bytes containing name of author/installation
D 8 bytes date (year/month/day, for example 79/12/31)
V 2 bytes version number:
the metafile described here has version number 1

Specification of field length

H 2 bytes integer specifying how many bytes of the string 'GKSM' are repeated
  at the beginning of each record. Possible values: 0, 1, 2, 3, 4.
T 2 bytes length of item type indicator field
L 2 bytes length of item data record length indicator field
I 2 bytes length of field for each integer in the item data record
  (applies to all data marked (i) in the
  item description)
R 2 bytes length of field for each real in the item data record
  (applies to all data marked (r) in the
  item description)

Specification of number representation

F 2 bytes Possible values: 1, 2. This applies to all data in the items marked (i)
  or (r) and to item type and item data record length:
  1: all numbers are formatted according to ISO 6093
  2: all numbers (except in the file header) are stored in an internal
  binary format
RI 2 bytes Possible values: 1, 2. This is the number representation for data
  marked (r):
  1 = real, 2 = integer
ZERO 11 bytes integer equivalent to 0.0, if RI=2
ONE 11 bytes integer equivalent to 1.0, if RI=2

After the file header, which is in fixed format, all values in the following items are in the for¬
mat defined by the file header. For the following description, the setting:

H=4; T=3; F=1

is assumed. In addition to formats (c), (i) and (r), which are already described, (p) denotes a
point represented by a pair of real numbers (2r). The notation allows the single letter to be
preceded by an expression, indicating the number of values of that type.

END ITEM

['GKSM 0' L]

Last item of every GKS Metafile. Sets condition for the error 'No item is left in GKS Metafile
input'
CLEAR WORKSTATION

'GKSM 1' L C

Requests CLEAR WORKSTATION on all active workstations

C(i): clearing control flag
(0=CONDITIONAL, 1=ALWAYS)

REDRAW ALL SEGMENTS ON WORKSTATION

'GKSM 2' L

Requests REDRAW ALL SEGMENTS ON WORKSTATION on all active workstations

UPDATE WORKSTATION

'GKSM 3' L R

Requests UPDATE WORKSTATION on all active workstations

R(i): update regeneration flag
(0=PERFORM, 1=POSTPONE)

DEFERRAL STATE

'GKSM 4' L D R

Requests SET DEFERRAL STATE on all active workstations

D(i): deferral mode
(0=ASAP, 1=BNIG, 2=BNIL, 3=ASTI)

R(i): implicit regeneration mode
(0=ALLOWED, 1=SUPPRESSED)

MESSAGE

'GKSM 5' L N T

Requests MESSAGE on all active workstations

N(i): number of characters in string

T(Nc): string with N characters

ESCAPE

'GKSM 6' L FI L M I R

Requests ESCAPE

FI(i): function identifier

L(i): length of integer data in data record

M(i): length of real data in data record

I(Li): integer data

R(Mr): real data
E.6 Items for output primitives

POLYLINE

\[ 'GKSM\ 11' \ \ L \ \ N \ \ P \]

N(i): number of points of the polyline
P(Np): list of points

POLYMARKER

\[ 'GKSM\ 12' \ \ L \ \ N \ \ P \]

N(i): number of points
P(Np): list of points

TEXT

\[ 'GKSM\ 13' \ \ L \ \ P \ \ N \ \ T \]

P(p): starting point of character string
N(i): number of characters in string T
T(Nc): string with N characters from the set of ISO 646

FILL AREA

\[ 'GKSM\ 14' \ \ L \ \ N \ \ P \]

N(i): number of points
P(Np): list of points

CELL ARRAY

\[ 'GKSM\ 15' \ \ L \ \ P \ \ Q \ \ R \ \ N \ \ M \ \ CT \]

P(p), Q(p), R(p): coordinates of corner points of pixel array (P and Q are the images of the points P and Q specified in the function CELL ARRAY and R is the point associated with the (DX,1) cell)
N(i): number of columns in array
M(i): number of rows in array
CT(MNi): array of colour indices stored row by row
GENERALIZED DRAWING PRIMITIVE

'GKSM 16' L GI N LI LR P I R

GI(i): GDP identifier
N(i): number of points
LI(i): length of integer data in data record
LR(i): length of real data in data record
P(Np): list of points
I(LIi): integer data
R(LRr): real data

E.7 Items for output primitive attributes

POLYLINE INDEX

'GKSM 21' L I

I(i): polyline index

LINETYPE

'GKSM 22' L LT

LT(i): linetype

LINEWIDTH SCALE FACTOR

'GKSM 23' L LW

LW(r): linewidth scale factor

POLYLINE COLOUR INDEX

'GKSM 24' L CI

CI(i): polyline colour index

POLYMARKER INDEX

'GKSM 25' L I

I(i): polymarker index

MARKER TYPE

'GKSM 26' L MT

MT(i): marker type
Appendix E

Items for output primitive attributes

MARKER SIZE SCALE FACTOR

'GKSM 27' L MS

MS(r): marker size scale factor

POLYMARKER COLOUR INDEX

'GKSM 28' L CI

CI(i): polymarker colour index

TEXT INDEX

'GKSM 29' L I

I(i): text index

TEXT FONT AND PRECISION

'GKSM 30' L F P

F(i): text font
P(i): text precision

(0=STRING, 1=CHAR, 2=STROKE)

CHARACTER EXPANSION FACTOR

'GKSM 31' L CEF

CEF(r): character expansion factor

CHARACTER SPACING

'GKSM 32' L CS

CS(r): character spacing

TEXT COLOUR INDEX

'GKSM 33' L CI

CI(i): text colour index

CHARACTER VECTORS

'GKSM 34' L CH CW

CH(2r): character height vector
CW(2r): character width vector

NOTE: These vectors are the height and width vectors described in 4.4.5
Items for output primitive attributes

Appendix E

TEXT PATH

\[ 'GKSM 35' \quad L \quad P \]

P(i): text path
(0=RIGHT, 1=LEFT, 2=UP, 3=DOWN)

TEXT ALIGNMENT

\[ 'GKSM 36' \quad L \quad H \quad V \]

H(i): horizontal character alignment
(0=NORMAL, 1=LEFT, 2=CENTRE, 3=RIGHT)

V(i): vertical character alignment
(0=NORMAL, 1=TOP, 2=CAP, 3=HALF, 4=BASE, 5=BOTTOM)

FILL AREA INDEX

\[ 'GKSM 37' \quad L \quad I \]

I(i): fill area index

FILL AREA INTERIOR STYLE

\[ 'GKSM 38' \quad L \quad S \]

S(i): fill area interior style
(0=HOLLOW, 1=SOLID, 2=PATTERN, 3=HATCH)

FILL AREA STYLE INDEX

\[ 'GKSM 39' \quad L \quad SI \]

SI(i): fill area style index

FILL AREA COLOUR INDEX

\[ 'GKSM 40' \quad L \quad CI \]

CI(i): fill area colour index

PATTERN VECTORS

\[ 'GKSM 41' \quad L \quad PW \quad PH \]

PW(2r): pattern width vector
PH(2r): pattern height vector

PATTERN REFERENCE POINT

\[ 'GKSM 42' \quad L \quad P \]

P(p): reference point
Appendix E

Items for output primitive attributes

ASPECT SOURCE FLAGS

`'GKSM 43' L F`

F(13i): aspect source flags
(0=BUNDLED, 1=INDIVIDUAL)

PICK IDENTIFIER

`'GKSM 44' L P`

P(i): pick identifier

E.8 Items for workstation attributes

POLYLINE REPRESENTATION

`'GKSM 51' L I LT LW CI`

I(i): polyline index
LT(i): linetype number
LW(r): linewidth scale factor
CI(i): polyline colour index

POLYMARKER REPRESENTATION

`'GKSM 52' L I MT MS CI`

I(i): polymarker index
MT(i): marker type
MS(r): marker size scale factor
CI(i): polymarker colour index

TEXT REPRESENTATION

`'GKSM 53' L I F P CEF CS CI`

I(i): text index
F(i): text font
P(i): text precision
(0=STRING, 1=CHAR, 2=STROKE)
CEF(r): character expansion factor
CS(r): character spacing
CI(i): text colour index
FILL AREA REPRESENTATION

I(i): fill area index
S(i): fill area interior style
   (0=HOLLOW, 1=SOLID, 2=PATTERN, 3=HATCH)
SI(i): fill area style index
CI(i): fill area colour index

PATTERN REPRESENTATION

I(i): pattern index
N(i): number of columns in array
M(i): number of rows in array
CT(MNi): table of colour indices stored row by row

COLOUR REPRESENTATION

CI(i): colour index
RGB(3r): red, green, blue intensities

E.9 Items for transformations

CLIPPING RECTANGLE

C(4r): limits of clipping rectangle (XMIN, XMAX, YMIN, YMAX)

WORKSTATION WINDOW

W(4r): limits of workstation window (XMIN, XMAX, YMIN, YMAX)

WORKSTATION VIEWPORT

V(4r): limits of workstation viewport (XMIN, XMAX, YMIN, YMAX)

E.10 Items for segment manipulation

CREATE SEGMENT

S(i): segment name
Appendix E

Items for segment manipulation

CLOSE SEGMENT

'GKSM 82' L

Indicates end of segment

RENAME SEGMENT

'GKSM 83' L SO SN

SO(i): old segment name
SN(i): new segment name

DELETE SEGMENT

'GKSM 84' L S

S(i): segment name

E.11 Items for segment attributes

SET SEGMENT TRANSFORMATION

'GKSM 91' L S M

S(i): segment name
M(6r): transformation matrix

\[ M_{11}, M_{12}, M_{13}, M_{21}, M_{22}, M_{23} \]

SET VISIBILITY

'GKSM 92' L S V

S(i): segment name
V(i): visibility

(0 = VISIBLE, 1 = INVISIBLE)

SET HIGHLIGHTING

'GKSM 93' L S H

S(i): segment name
H(i): highlighting

(0 = NORMAL, 1 = HIGHLIGHTED)

SET SEGMENT PRIORITY

'GKSM 94' L S P

S(i): segment name
P(r): segment priority
SET DETECTABILITY

GKSM 95

S(i): segment name
D(i): detectability
0 = UNDETECTABLE, 1 = DETECTABLE

E.12 User items

USER ITEM

GKSMXXX

XXX > 100
D: user data (L bytes)
Appendix F  GKS functions summary

NOTE. This appendix is not part of the Standard, but provides additional information.

F.1 Control functions

OPEN GKS
   Start working with GKS.
CLOSE GKS
   Stop working with GKS.
OPEN WORKSTATION
   Create a connection between the specified workstation and GKS.
CLOSE WORKSTATION
   Release the connection between the specified workstation and GKS.
ACTIVATE WORKSTATION
   Output is routed to the specified workstation.
DEACTIVATE WORKSTATION
   Output is no longer routed to the specified workstation.
CLEAR WORKSTATION
   Perform all deferred actions and clear display space on the specified workstation. All segments stored on the workstation are deleted.
REDRAW ALL SEGMENTS ON WORKSTATION
   Redraw all visible segments stored on the specified workstation.
UPDATE WORKSTATION
   Perform all deferred actions and, if necessary, redraw all visible segments stored on the specified workstation.
SET DEFERRAL STATE
   Set deferral state for the specified workstation.
MESSAGE
   Send a message to the specified workstation.
ESCAPE
   A standard way of invoking non-standard features.

F.2 Output functions

POLYLINE
   Generate a polyline defined by points in world coordinates.
POLYMARKER
   Generate markers of a given type at specified points in world coordinates.
TEXT
   Generate a text string at the given position in world coordinates.
FILL AREA
   Generate a polygon which may be filled with a colour, a hatch or a pattern, or may be hollow.
CELL ARRAY
   Map the given array of colour indices onto the display surface.
GENERALIZED DRAWING PRIMITIVE
Generate a generalized drawing primitive defined by a sequence of points in world coordinates and a data record.

F.3 Output attributes

F.3.1 Workstation independent primitive attributes

SET POLYLINE INDEX
Select a bundle index for polylines.

SET LINETYPE
Set the linetype for use when the corresponding ASF is INDIVIDUAL.

SET LINEWIDTH SCALE FACTOR
Set the linewidth scale factor for use when the corresponding ASF is INDIVIDUAL.

SET POLYLINe COLOUR INDEX
Set the polyline colour index for use when the corresponding ASF is INDIVIDUAL.

SET POLYMARKER INDEX
Select a bundle index for polymarkers.

SET MARKER TYPE
Set the marker type for use when the corresponding ASF is INDIVIDUAL.

SET MARKER SIZE SCALE FACTOR
Set the marker size scale factor for use when the corresponding ASF is INDIVIDUAL.

SET POLYMARKER COLOUR INDEX
Set the polymarker colour index for use when the corresponding ASF is INDIVIDUAL.

SET TEXT INDEX
Select a bundle index for text.

SET TEXT FONT AND PRECISION
Set the text font and precision for use when the corresponding ASF is INDIVIDUAL.

SET CHARACTER EXPANSION FACTOR
Set the character expansion factor as a fraction of the character height for use when the corresponding ASF is INDIVIDUAL.

SET CHARACTER SPACING
Set the character spacing as a fraction of the character height for use when the corresponding ASF is INDIVIDUAL.

SET TEXT COLOUR INDEX
Set the text colour index for use when the corresponding ASF is INDIVIDUAL.

SET CHARACTER HEIGHT
Set the character height in world coordinates.

SET CHARACTER UP VECTOR
Set the character up vector in world coordinates.

SET TEXT PATH
Set the text path.

SET TEXT ALIGNMENT
Set the horizontal and vertical alignment of text strings.

SET FILL AREA INDEX
Select a bundle index for fill area.

SET FILL AREA INTERIOR STYLE
Set the fill area interior style for use when the corresponding ASF is INDIVIDUAL.
Appendix F

Output attributes

SET FILL AREA STYLE INDEX
Set the fill area style index for use when the corresponding ASF is INDIVIDUAL.

SET FILL AREA COLOUR INDEX
Set the fill area colour index for use when the corresponding ASF is INDIVIDUAL.

SET PATTERN SIZE
Set the pattern size in world coordinates for use in the display of fill area primitives with interior style PATTERN.

SET PATTERN REFERENCE POINT
Set the pattern reference point in world coordinates for use in the display of fill area primitives with interior style PATTERN.

SET ASPECT SOURCE FLAGS
Define whether the value of each non-geometric aspect is obtained from the corresponding individual attribute or from the appropriate bundle on the workstation.

SET PICK IDENTIFIER
Set pick identifier.

F.8.2 Workstation attributes (representations)

SET POLYLINE REPRESENTATION
Define the representation of polylines on the specified workstation.

SET POLYMARKER REPRESENTATION
Define the representation of polymarkers on the specified workstation.

SET TEXT REPRESENTATION
Define the representation of text on the specified workstation.

SET FILL AREA REPRESENTATION
Define the representation of fill area primitives on the specified workstation.

SET PATTERN REPRESENTATION
Define the pattern to be associated with a pattern index (i.e. a fill area style index) on the specified workstation.

SET COLOUR REPRESENTATION
Define the colour to be associated with a colour index on the specified workstation.

F.4 Transformation functions

F.4.1 Normalization transformation

SET WINDOW
Set the window in world coordinates of the specified normalization transformation.

SET VIEWPORT
Set the viewport in normalized device coordinates of the specified normalization transformation.

SET VIEWPORT INPUT PRIORITY
Set the input priority of the specified viewport for locator and stroke input.

SELECT NORMALIZATION TRANSFORMATION
Select a normalization transformation for output.

SET CLIPPING INDICATOR
Set the clipping indicator for the clipping rectangle.
F.4.2 Workstation transformation

SET WORKSTATION WINDOW
Set the workstation window in normalized device coordinates.

SET WORKSTATION VIEWPORT
Set the workstation viewport in device coordinates.

F.5 Segment functions

F.5.1 Segment manipulation functions

CREATE SEGMENT
The specified segment is created and becomes the open segment.

CLOSE SEGMENT
Close the open segment.

RENAME SEGMENT
Change the name of the specified segment.

DELETE SEGMENT
Delete the specified segment.

DELETE SEGMENT FROM WORKSTATION
Delete the specified segment from the specified workstation.

ASSOCIATE SEGMENT WITH WORKSTATION
Associate the specified segment, present in workstation independent segment storage, with the specified open workstation.

COPY SEGMENT TO WORKSTATION
Copy the primitives of the specified segment, present in workstation independent segment storage, to the specified workstation.

INSERT SEGMENT
Insert the specified segment, present in workstation independent segment storage, (after the segment transformation and the insert transformation have been applied) into the open segment or the stream of primitives outside segments.

F.5.2 Segment attributes

SET SEGMENT TRANSFORMATION
Set the segment transformation attribute for the specified segment.

SET VISIBILITY
Set the visibility attribute for the specified segment.

SET HIGHLIGHTING
Set the highlighting attribute for the specified segment.

SET SEGMENT PRIORITY
Set the segment priority attribute for the specified segment.

SET DETECTABILITY
Set the segment detectability attribute for the specified segment.

F.6 Input functions

F.6.1 Initialisation of input devices

INITIALISE LOCATOR
Initialise the specified locator device.
INITIALISE STROKE
  Initialise the specified stroke device.

INITIALISE VALUATOR
  Initialise the specified valuator device.

INITIALISE CHOICE
  Initialise the specified choice device.

INITIALISE PICK
  Initialise the specified pick device.

INITIALISE STRING
  Initialise the specified string device.

F.6.2 Setting mode of input devices

SET LOCATOR MODE
  Set operating mode of the specified locator device.

SET STROKE MODE
  Set operating mode of the specified stroke device.

SET VALUATOR MODE
  Set operating mode of the specified valuator device.

SET CHOICE MODE
  Set operating mode of the specified choice device.

SET PICK MODE
  Set operating mode of the specified pick device.

SET STRING MODE
  Set operating mode of the specified string device.

F.6.3 Request input functions

REQUEST LOCATOR
  Request position in world coordinates and normalization transformation number from the
  specified locator device.

REQUEST STROKE
  Request sequence of points in world coordinates and normalization transformation number
  from the specified stroke device.

REQUEST VALUATOR
  Request real value from the specified valuator device.

REQUEST CHOICE
  Request non-negative integer, representing a selection from a number of choices, and
  choice status from the specified choice device.

REQUEST PICK
  Request segment name, pick identifier and pick status from the specified pick device.

REQUEST STRING
  Request character string from the specified string device.

F.6.4 Sample input functions

The current setting of a logical input device is tested and the value is sent back without waiting
for any operator action.

SAMPLE LOCATOR
  Sample the specified locator device, delivering a point in world coordinates and a normaliza-
  tion transformation number.
SAMPLE STROKE
Sample the specified stroke device, delivering a sequence of points in world coordinates and a normalization transformation number.

SAMPLE VALUATOR
Sample the specified valuator device, delivering a real value.

SAMPLE CHOICE
Sample the specified choice device, delivering a non-negative integer, which represents a selection from a number of choices and choice status.

SAMPLE PICK
Sample the specified pick device, delivering a segment name, pick identifier and pick status.

SAMPLE STRING
Sample the specified string device, delivering a character string.

F.6.5 Event input functions
Input items are collected in an input queue managed by GKS and can be obtained by the application program from this queue.

AWAIT EVENT
If the input queue is empty, wait for an input item until the specified time has elapsed. Read the workstation identifier, input class, and logical input device number of the oldest entry in the input queue and pass the values to the current event report for subsequent interrogation by the GET functions.

FLUSH DEVICE EVENTS
Delete all the events from the specified logical input device in the input queue.

GET LOCATOR
Transfer position in world coordinates and normalization transformation number from the current event report to the application program.

GET STROKE
Transfer sequence of points in world coordinates and normalization transformation number from the current event report to the application program.

GET VALUATOR
Transfer real value from the current event report to the application program.

GET CHOICE
Transfer non-negative integer, representing a selection from a number of choices, and choice status from the current event report to the application program.

GET PICK
Transfer segment name, pick identifier and pick status from the current event report to the application program.

GET STRING
Transfer character string from the current event report to the application program.

F.7 Metafile functions

WRITE ITEM TO GKSM
Pass non-graphical data from the application program to the GKS metafile.

GET ITEM TYPE FROM GKSM
Pass the item type and item data record length of the current item back to the application program.

READ ITEM FROM GKSM
Pass the current item to the application program (graphical or non-graphical item).
INTPRET ITEM
Interpret the item read in by READ ITEM FROM GKSM. The interpretation causes appropriate changes in the set of GKS state variables and generates appropriate graphical output as determined by the metafile specification.

F.8 Inquiry functions
There are some 75 different inquiry functions in GKS. All variables contained in any existing state list and in the description tables may be inquired at any time when GKS is open.

An inquiry function is provided for text extent to allow concatenation of character strings. On raster workstations, the size and colour of pixels may be inquired.

The operating state of GKS may be inquired, even when GKS is closed.

F.9 Utility functions
EVALUATE TRANSFORMATION MATRIX
Evaluate the transformation specified by fixed point, shift vector, rotation angle and scale factors and return the result in the output transformation matrix.

ACCUMULATE TRANSFORMATION MATRIX
Evaluate the transformation specified by fixed point, shift vector, rotation angle and scale factors, combine it with the input transformation matrix and return the result in the output transformation matrix.

F.10 Error handling
EMERGENCY CLOSE GKS
Tries to close GKS in case of an error, saving as much information as possible.

ERROR HANDLING
A procedure called by GKS when an error is detected. It may be user supplied.

ERROR LOGGING
A procedure called by the standard GKS error handling procedure. It prints an error message and function identification on the error file.
Appendix G  Differences Between ANS GKS and ISO GKS

This appendix is not part of the Standard, but provides additional information.

The following is a description of the changes made to ISO GKS to produce ANS GKS. The changes are listed in the order in which they appear in the document.

1. A new title page with new abstract and a new Foreword with a history section were created to conform to the ANSI Style Manual.
2. Changes were made in the Table of Contents to reflect changes in the document.
3. In the Introduction a Conformance section was added. This included the a different definition of conforming to a level of GKS, reflected here and in other places in the document.
4. ANSI references were added to section 2.
5. Parts of the document discussing levels were changed to reflect the addition of level m. Areas affected include the Definitions, the description of the levels, the level designations of individual functions, and Appendix A.
6. Starting in Section 4 and continuing throughout the document, references to 'annexes' were changed to 'appendices'. This was done to avoid confusion, as the ANSI Style Manual has explicit rules with respect to appendices; retention of the term 'annexes' might have caused ambiguity. The non-U.S. spelling of various terms (e.g., 'colour') was not changed because there was little danger of ambiguity.
7. In 4.4.2, Output Primitive Attributes, the default for ASF's is INDIVIDUAL, rather than implementation dependent.
8. Input data records are part of the standard.
9. Appendix A and Appendix B were marked as not part of the standard. It is not part of the standard because all the information contained in it is also in the body of the standard; redundant information in the standard increases the potential for ambiguities.
10. Annex C was replaced with a new Appendix C reflecting a number of changes.
11. Annex F, Sample Programs, was deleted. Its function is replaced with examples in the derivative standards reflecting the bindings written for each language.
12. Annex G was changed to Appendix F.
13. This Appendix G was added consisting of this list of differences between ISO GKS and ANS GKS.
American National Standard

for information systems -

computer graphics -

graphical kernel system (GKS)

FORTRAN binding
Approval of an American National Standard requires verification by ANSI that the requirements for due process, consensus, and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that a concerted effort be made toward their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give an interpretation of any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken to reaffirm, revise, or withdraw this standard no later than five years from the date of approval. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.
Abstract

This standard provides the FORTRAN language syntax for American National Standard for Information Systems — Computer Graphics — Graphical Kernel System (GKS) Functional Description, ANSI X3.124-1985. The FORTRAN language binding of GKS is a syntactic specification, presented as a set of subroutines that, taken as a whole, provide the semantics of GKS for use by a FORTRAN application program.

For each GKS function, the FORTRAN subroutine name, argument list, and argument data types are given. In addition, any special errors associated only with the FORTRAN language binding of GKS are specified and assigned unique error numbers. Finally, for the GKS enumeration data types, the spellings of integer variables and their numeric data value assignments are suggested to further aid in the production of portable and maintainable GKS FORTRAN programs.
This American National Standard provides access to a set of basic functions for computer
graphics programming in American National Standard Programming Language FORTRAN,
ANSI X3.9-1978, colloquially known as FORTRAN '77. These graphics functions taken
as a whole are called the FORTRAN language binding of the graphical kernel system
(GKS).

The graphical kernel system is a set of basic functions for computer graphics programming,
useable by many graphics-producing applications. This standard (1) allows FORTRAN
graphics application programs to be easily transported between installations, (2) aids
FORTRAN graphics applications programmers in understanding and using graphics meth¬
ods, and (3) guides device manufacturers on useful graphics capabilities.

This standard defines a FORTRAN application level programming interface to a graphics
system. Hence, it contains functions for (1) outputting graphical primitives, (2) controlling
the appearance of graphical primitives with attributes, (3) controlling graphical work¬
stations, (4) controlling transformations and coordinate systems, (5) generating and
controlling groups of primitives called segments, (6) obtaining graphical input, (7) manipu¬
lating groups of device-independent instructions called metafiles, (8) inquiring the
capabilities and states of the graphics system, and (9) handling errors.

For each GKS function, the FORTRAN subroutine name, argument list, and argument
data types are given. In addition, any special errors associated only with the FORTRAN
language binding of GKS are specified and assigned unique error numbers. Finally, for the
GKS enumeration data types, the spellings of integer variables and their numeric data
value assignments are suggested to further aid in the production of portable and maintain¬
able GKS FORTRAN programs.

Twelve upwardly compatible levels of conformance are defined, addressing the most com¬
mon classes of equipment and applications.

Kernel System (GKS) Functional Description, ANSI X3.124-1985, is supplemented by
this derivative standard. ANSI X3.124-1985 corresponds to ISO 7942-1985 in that it
represents the functional aspects of GKS. ANSI X3.124.1-1985 contains specifications
not present in ANSI X3.124-1985, namely, the syntax for using GKS functions and data
types from FORTRAN '77.

The design of this standard is based on the work of many groups. Much of the early de¬
design methodology of graphics standards was developed at the Workshop on Graphics
Standards Methodology held in May, 1976, in Seillac, France, under IFIP WG5.2 spon¬
sorship. GKS itself was originally developed by Deutsches Institut fur Normung (DIN),
the West German standardization institute, in 1978 and was subsequently refined exten¬
sively between 1980 and 1982 by Working Group 2 of the Subcommittee on Program¬
mming Languages of the Technical Committee on Information Processing of the Interna¬
tional Organization for Standardization (ISO TC 97/SC5/WG2). The resulting Interna¬
tional Standard (Information Processing – Computer Graphics – Graphical Kernel Sys¬
tem (GKS) Functional Description, ISO 7942-1985) was the basis for ANSI X3.124-1985.
The development of the GKS was heavily influenced by the work of the Graphic Stan¬
dards Planning Committee of the Special Interest Group on Computer Graphics of the
Association for Computing Machinery (ACM SIGGRAPH GSPC). This work, known as
Core System Proposal, was published and widely distributed in 1977 and again (in a re¬
vised version) in 1979.

The FORTRAN binding of GKS was started by American and British participants of ISO
TC 97/SC5/WG2, which, in 1985, was renamed ISO TC 97/SC21/WG2. After refinement
by both Technical Committee X3H3 (Computer Graphics) of Accredited Standards Com¬
ittee X3 (Information Processing) and by ISO TC 97/SC21/WG2, the document was
ANSI X3.124.1-1985 is identical to ISO DP 8651/1 in almost all areas of the standard. All functional capabilities of ISO GKS are found in the ANSI GKS and are bound to the FORTRAN programming language identically. The ANSI GKS does, however, differ in the following ways:

1. A new minimal output level (denoted m) is defined in ANSI X3.124-1985.
2. A new section defining a conforming program and a conforming implementation replaces a more restrictive conformance statement found in the body of the ISO GKS standard document.
3. Several of the Annexes in the ISO GKS document have been modified. Also, the word “Annex” has been changed to “Appendix.”
4. The default for ASF’s is INDIVIDUAL.
5. The data records for INPUT have been defined.

All these differences are also reflected in this standard.

Three additional language bindings of GKS are under development by Technical Committee X3H3: Pascal, Ada, and C. These standards, when approved by X3 and ANSI, will be published as ANSI X3.124.2, X3.124.3, and X3.124.4, respectively. Internationally, these language bindings of GKS will be published as parts of a multipart ISO standard, currently known as ISO/DP 8651.

This standard was developed by Technical Committee X3H3 of Accredited Standards Committee X3 under two projects authorized by X3; namely, project 268D and project 362D. More specifically, GKS, as a whole, meets the goals of project 268D, while the minimal output level m found in this American National Standard, but not present in ISO 7942-1985, meets the goals of project 362D. Both projects authorized the specification of syntax (as embodied in a programming language binding) as well as semantics.

This standard was approved as an American National Standard by the American National Standards Institute on June 24, 1985.

Suggestions for improvement of this standard will be welcome. They should be sent to the Computer and Business Equipment Manufacturers Association, 311 First Street, NW, Suite 500, Washington, DC 20001.

This standard was processed and approved for submittal to ANSI by the Accredited Standards Committee on Information Processing Systems, X3. Committee approval of the standard does not necessarily imply that all committee members voted for its approval. At the time it approved this standard, the X3 Committee had the following members:

Edward Lohse, Chair
Catherine A. Kachurik, Administrative Secretary

Organization Represented
American Library Association .................................................. Paul Peters
American Nuclear Society .......................................................... Geraldine C. Main
AMP Incorporated ................................................................. D. R. Vondy (Alt)
Association of American Railroads .............................................. Patrick E. Lannan
Association for Computing Machinery ....................................... Edward Kelly (Alt)
Association of the Institute for Certification of Computer Professionals ............................................. R. A. Petrasch
AT&T Corporation ........................................................................ Kenneth Mangel
AT&T Information Systems ........................................................ Jon A. Meads (Alt)

Name of Representative
Thomas M. Kurihara
Ardyn E. Dubnow (Alt)
Henry L. Marchese
Stuart H. Garland (Alt)
Richard Gibson (Alt)
Herbert V. Bertine
Paul D. Bartoli (Alt)
<table>
<thead>
<tr>
<th>Organization Represented</th>
<th>Name of Representative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Burroughs Corporation</td>
<td>Ira R. Purchis</td>
</tr>
<tr>
<td>Control Data Corporation</td>
<td>Stanley Fenner (Alt)</td>
</tr>
<tr>
<td>Cooperating Users of Burroughs Equipment</td>
<td>Charles E. Cooper</td>
</tr>
<tr>
<td>Data General Corporation</td>
<td>Keith Lucke (Alt)</td>
</tr>
<tr>
<td>Data Processing Management Association</td>
<td>Thomas Easterday</td>
</tr>
<tr>
<td>Digital Equipment Computer Users Society</td>
<td>Donald Miller (Alt)</td>
</tr>
<tr>
<td>Digital Equipment Corporation</td>
<td>John Pilat</td>
</tr>
<tr>
<td>General Electric Company</td>
<td>Lyman Chapin (Alt)</td>
</tr>
<tr>
<td>General Services Administration</td>
<td>Christian G. Meyer</td>
</tr>
<tr>
<td>GUIDE International</td>
<td>Paula Morin</td>
</tr>
<tr>
<td>Harris Corporation</td>
<td>Lois C. Frampton</td>
</tr>
<tr>
<td>Hewlett-Packard</td>
<td>Gary S. Robinson (Alt)</td>
</tr>
<tr>
<td>Honeywell Information Systems</td>
<td>Delbert L. Shoemaker (Alt)</td>
</tr>
<tr>
<td>IBM Corporation</td>
<td>William R. Knesi</td>
</tr>
<tr>
<td>IEEE Computer Society</td>
<td>William C. Rinehuls</td>
</tr>
<tr>
<td>Lawrence Berkeley Laboratory</td>
<td>Larry L. Jackson (Alt)</td>
</tr>
<tr>
<td>Lawrence Berkeley Laboratory</td>
<td>Thomas F. O'Leary, Jr (Alt)</td>
</tr>
<tr>
<td>Life Office Management Association</td>
<td>Sam Mathan</td>
</tr>
<tr>
<td>Life Office Management Association</td>
<td>Rajiv Sinha (Alt)</td>
</tr>
<tr>
<td>Moore Business Forms</td>
<td>Donald C. Loughry</td>
</tr>
<tr>
<td>National Bureau of Standards</td>
<td>Thomas J. McNamara</td>
</tr>
<tr>
<td>National Communications System</td>
<td>David M. Taylor (Alt)</td>
</tr>
<tr>
<td>NCR Corporation</td>
<td>Mary Anne Gray</td>
</tr>
<tr>
<td>NCR Corporation</td>
<td>Robert H. Follett (Alt)</td>
</tr>
<tr>
<td>Perkin-Elmer Corporation</td>
<td>Sava I. Sherr</td>
</tr>
<tr>
<td>Prime Computer, Inc</td>
<td>David Gelperin (Alt)</td>
</tr>
<tr>
<td>Recognition Technology Users Association</td>
<td>Thomas A. Varetoni (Alt)</td>
</tr>
<tr>
<td>SHARE, Inc</td>
<td>David F. Stevens</td>
</tr>
<tr>
<td>Sperry Corporation</td>
<td>John S. Colonias (Alt)</td>
</tr>
<tr>
<td>Texas Instruments, Inc</td>
<td>James J. Merrick</td>
</tr>
<tr>
<td>Texas Instruments, Inc</td>
<td>James P. Foley, Jr (Alt)</td>
</tr>
<tr>
<td>3M Company</td>
<td>Delmer H. Oddy</td>
</tr>
<tr>
<td>Travelers Insurance Companies, Inc</td>
<td>Robert E. Rountree</td>
</tr>
<tr>
<td>U.S. Department of Defense</td>
<td>James H. Burrows (Alt)</td>
</tr>
<tr>
<td>VIM</td>
<td>Marshall L. Cain</td>
</tr>
<tr>
<td>VIM</td>
<td>George W. White (Alt)</td>
</tr>
<tr>
<td>Wang Laboratories, Inc</td>
<td>Thomas W. Kern</td>
</tr>
<tr>
<td>Wang Laboratories, Inc</td>
<td>A. R. Daniels (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Christoph Beling</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Russ Lombardo (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Andrew F. Burlingame</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Jacqueline Barbour (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Herbert F. Schantz</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>G. W. Wetzel (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Thomas B. Steel</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Daniel Schuster (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Marvin W. Bass</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Presley Smith</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Richard F. Trow, Jr (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>J. Wade Van Valkenburg</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Joseph T. Brophy</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Belkis Leong-Hong (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Fred Virtue (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Chris Tanner</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Madeleine Sparks (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Marsha Hayek</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Joseph St. Amand (Alt)</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>John L. Wheeler</td>
</tr>
<tr>
<td>Xerox Corporation</td>
<td>Arthur R. Machell (Alt)</td>
</tr>
</tbody>
</table>
Technical Committee X3H3 on Computer Graphics, which developed the draft proposals, which held the U.S. Technical Advisory Group responsibilities for ISO TC 97/SC5/WG2, and through which this standard was completed, had the following members:

Peter R. Bono, Chair  
(Athena Systems)
Barry Shepherd, Vice-Chair  
(IBM)
Randall L. Simons, Secretary  
(Sandia, NM)
Janet Chin, International Representative  
(Tymshare, Inc)

David C. Bailey  
(Sanders Associates)
Jerry Bedrick  
(Wang Laboratories)
John Blair  
(Mindset Corporation)
Robert Bruns  
(Meatek)
Albert Bunshaft  
(RPI)
Debbie Cahn  
(LBL)
Fred Canfield  
(Systonetics, Inc)
George S. Carson  
(GSC Associates)
Tom Clarkson  
(GSS)
Bruce Cohen  
(Intel)
Geraldine Cuthbert  
(Harris)
Warren Dale  
(Calma)
Richard Ehlers  
(E & S)
Sam Gill  
(Benson)
Jim Hargrove  
(Data General)
Terry Harney  
(Hughes Aircraft)
Loften Henderson  
(NCAR)
Margaret Journey  
(Precision Visuals)
Jim Kearney  
(US Army)
Fred Langhorst  
(Digital Research)
Olga Lopczak  
(Nrippa)
Allen Leinwand  
(Olivetti)

K. C. Leung  
(Computer Sciences)
Chris Mannhardt  
(BNR)
Tom Morrissey  
(Hewlett-Packard)
Paul Norman  
(Imlac Corporation)
Brian Plunkett  
(Houston Instrument)
Tom Powers  
(DEC)
L. Preheim  
(JPL)
Richard F. Puk  
(Puk Consulting Services)
Tom Reed  
(LANL)
Jeff Rowe  
(Lawrence Livermore National Laboratory)
Kurt Schmucker  
(NSA)
Chuck Seum  
(CDC)
David Shuey  
(McAuto)
Mark Skall  
(NBS)
Elaine Sondergeger  
(SIGGRAPH)
Norman Soong  
(Sperry)
Madeleine Sparks  
(SDC)
Richard Stout  
(SAI)
David Straayer  
(Tektronix)
Meredith L. Whyles  
(Amoco Production)
Thomas Wright  
(ISSCO)

The document editor for this standard was Thomas Wright. The camera-ready masters for the body of this standard were provided by Integrated Software Systems Corporation, San Diego, CA.
## Contents

0. Introduction 1
1. Scope and field of application 2
2. References 3
3. Principles
   3.1 Mapping of GKS Function Names to FORTRAN Subroutine Names 4
   3.2 Parameters 4
   3.3 The FORTRAN Subset 4
   3.4 Error Handling 5
4. Generating FORTRAN Subroutine Names 6
5. Data Types 8
6. Enumeration Types 12
7. List of the GKS Function Names and Levels
   7.1 List Ordered by Bound Names 16
   7.2 List Ordered by GKS Function Names 20
   7.3 List Ordered by Levels 23
8. GKS Errors Specific to the FORTRAN Binding 28
9. The GKS Function Interface
   9.1 General Principles 29
   9.2 Control Functions 30
   9.3 Output Functions 33
   9.4 Output Attributes
      9.4.1 Workstation Independent Primitive Attributes 35
      9.4.2 Workstation Attributes (Representations) 39
   9.5 Transformation Functions
      9.5.1 Normalization Transformation 41
      9.5.2 Workstation Transformation 42
   9.6 Segment Functions
      9.6.1 Segment Manipulation Functions 43
      9.6.2 Segment Attributes 44
   9.7 Input Functions
      9.7.1 Initialisation of Input Devices 46
      9.7.2 Setting Mode of Input Devices 51
      9.7.3 Request Input Functions 52
      9.7.4 Sample Input Functions 54
      9.7.5 Event Input Functions 56
   9.8 Metafile Functions 59
   9.9 Inquiry Functions
      9.9.1 Inquiry Function for Operating State Value 61
      9.9.2 Inquiry Functions for GKS Description Table 61
      9.9.3 Inquiry Functions for GKS State List 62
      9.9.4 Inquiry Functions for Workstation State List 70
      9.9.5 Inquiry Functions for Workstation Description Table 80
      9.9.6 Inquiry Functions for Segment State List 91
      9.9.7 Pixel Inquiries 91
      9.9.8 Inquiry Function for GKS Error State List 92
   9.10 Utility Functions 93
   9.11 Error Handling 94
10. Utility Functions not defined in GKS 95
    A. FORTRAN Examples 97
0. Introduction

The Graphical Kernel System (GKS) is registered as ANS X3.124-1985. As explained in the Scope and Field of Applications of X3.124-1985, that American National 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 FORTRAN computer programming language.
1. Scope and field of application

The Graphical Kernel System (GKS) ANS X3.124-1985 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 document specifies such a language dependent layer for the FORTRAN language.
2. References

ANSI X3.4-1977  7-bit coded character set for information processing interchange

ANSI X3.9-1978  FORTRAN Language Specification
Principles

3. Principles
This document defines the GKS language binding interface for ANS FORTRAN (ANSI X3.9-1978), commonly known as FORTRAN 77. With some minor modifications, application programs can be transported between full FORTRAN 77 and FORTRAN 77 Subset GKS installations.

8.1 Mapping of GKS Function Names to FORTRAN Subroutine Names
The function names of GKS are all mapped to FORTRAN subroutine names which start with the letter 'G'. The mapping is generally done in a one-to-one correspondence to the GKS document. However, some inquiry functions are split into more than one subroutine in this binding, due to the number of parameters required. The remaining letters after the first one are obtained by deriving a unique acronym from the words of the function name; e.g., ACTIVATE becomes AC, WORKSTATION becomes WK. Hence, the FORTRAN subroutine name of GKS function ACTIVATE WORKSTATION is GACWK. For a list of all abbreviations, see 'Generating FORTRAN Subroutine Names' below. Names used internally which may be known outside GKS, e.g., during linking, start with some easily recognized and documented form such as 'GK' (subroutine, function, and common block names). Therefore, no external names starting with this construct should be chosen when using GKS, in order to avoid name conflicts. Globally used GKS names may be renamed if necessary.

8.2 Parameters
In general, the order of GKS function parameters is preserved. For some subroutines, however, there are additional parameters which have been inserted in the normal parameter sequence (e.g., array length for arrays which are output parameters).

Values of input parameters are unaltered by any GKS function as well as PACK DATA RECORD and UNPACK DATA RECORD.

In order that any element of a list (member of a set), such as the set of segment names, can be inquired, in this binding the inquiry functions return only a single element of a list (member of a set). In addition, the total number of elements of the list (members of the set) is always returned. The elements (members) are numbered starting from 1; each invocation of the inquiry function requires the desired element (member) number as an input parameter and returns the corresponding element (member). When the list (set) is empty, a zero is returned as the number of elements (members) and the parameter representing the single element in the list is undefined.

8.9 The FORTRAN Subset
The binding for FORTRAN 77 Subset is different from that for full FORTRAN 77 in order to accommodate the FORTRAN 77 Subset restrictions.

Those GKS subroutines in the full FORTRAN 77 binding that have arguments of type CHARACTER(*) have alternative subroutine definitions that include fixed length character strings, CHARACTER*80, for the Subset.

In some cases, an additional INTEGER parameter (the number of characters) appears in the parameter list and the Subset version is distinguished by the addition of a final 'S', so that they can coexist in the same implementation. In other cases the INTEGER is already present and the FORTRAN 77 Subset version has the same name as the full FORTRAN 77 version.

A full FORTRAN 77 implementation shall include both subroutines in the case when the names are distinct and only the full FORTRAN 77 version when the names are the same.
The FORTRAN Subset

The enumeration values in this binding may be redefined by replacing the PARAMETER statements with corresponding DATA statements.

A 'FORTRAN 66' binding can be derived from the FORTRAN 77 Subset binding by replacing CHARACTER declarations with INTEGER arrays.

8.4 Error Handling

There are two error routines in every GKS system, named GERLOG and GERHND. The user may replace the latter with his own subroutine using the same name, GERHND. Furthermore, this user-defined error routine may call the system-defined error logging procedure GERLOG.
4. Generating FORTRAN Subroutine Names

For the binding of the GKS functions which inquire lists (sets), the word 'element' ('member') is added to the GKS function name before the subroutine name is generated from the resulting terms.

The derivation of the abbreviation for the subroutine names is performed in several steps. First, plurals are reduced to their singular form, and grammatical derivations are unified. Next, some compound terms are reduced. Finally, each remaining word is replaced by the null string or by an abbreviation.

### Plurals

- ATTRIBUTES → ATTRIBUTE
- DEVICES → DEVICE
- EVENTS → EVENT
- FACILITIES → FACILITY
- FLAGS → FLAG
- INDICES → INDEX
- NAMES → NAME
- NUMBERS → NUMBER
- PRIMITIVES → PRIMITIVE
- PRIORITIES → PRIORITY
- SEGMENTS → SEGMENT
- TYPES → TYPE
- VALUES → VALUE
- WORKSTATIONS → WORKSTATION

### Keeping Uniqueness

- ACTIVE → ACTIVATE
- DRAWING → DRAW
- IDENTIFIER → IDENTIFICATION
- SPACING → SPACE

### Reduce Compound Terms:

- STATE TABLES → TABLES
- TRANSFORMATION NUMBER → TRANSFORMATION N
- SET member → member
- CURRENT NORMALISATION → CN
- MAXIMUM LENGTH → LENGTH

### Deletions

- ALL FACTOR LIST OF TABLES TO
- AND FROM member ON TYPE
- AVAILABLE GKSM MODIFICATION POINT VALUE
- CURRENT IN MORE SIZE VECTOR
- DATA INDICATOR NAME STATES
- DEVICE LENGTH NUMBER SUPPORTED WITH
- EVENT

Page 6
### Generating FORTRAN Subroutine Names

<table>
<thead>
<tr>
<th>Abbreviations</th>
<th>→</th>
<th>Abbreviations</th>
<th>→</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCUMULATE</td>
<td>AC</td>
<td>LINETYPE</td>
<td>LN</td>
</tr>
<tr>
<td>ACTIVATE</td>
<td>AC</td>
<td>LINEWIDTH</td>
<td>LN</td>
</tr>
<tr>
<td>ALIGNMENT</td>
<td>AL</td>
<td>LOCATOR</td>
<td>LC</td>
</tr>
<tr>
<td>AREA</td>
<td>A</td>
<td>LOGGING</td>
<td>LOG</td>
</tr>
<tr>
<td>ARRAY</td>
<td>A</td>
<td>LOGICAL</td>
<td>L</td>
</tr>
<tr>
<td>ASPECT</td>
<td>A</td>
<td>MARKER</td>
<td>MK</td>
</tr>
<tr>
<td>ASSOCIATE</td>
<td>A</td>
<td>MATRIX</td>
<td>M</td>
</tr>
<tr>
<td>ASSOCIATED</td>
<td>AS</td>
<td>MAXIMUM</td>
<td>M</td>
</tr>
<tr>
<td>ATTRIBUTE</td>
<td>A</td>
<td>MESSAGE</td>
<td>MSG</td>
</tr>
<tr>
<td>AWAIT</td>
<td>WAIT</td>
<td>MODE</td>
<td>M</td>
</tr>
<tr>
<td>BASE</td>
<td>B</td>
<td>NORMALIZATION</td>
<td>N</td>
</tr>
<tr>
<td>CATEGORY</td>
<td>CA</td>
<td>OPERATING</td>
<td>OP</td>
</tr>
<tr>
<td>CELL</td>
<td>C</td>
<td>OVERFLOW</td>
<td>OV</td>
</tr>
<tr>
<td>CHARACTER</td>
<td>CH</td>
<td>PACK</td>
<td>P</td>
</tr>
<tr>
<td>CLASSIFICATION</td>
<td>CL</td>
<td>PATTERN</td>
<td>PA</td>
</tr>
<tr>
<td>CLEAR</td>
<td>CLR</td>
<td>PICK</td>
<td>PK</td>
</tr>
<tr>
<td>CLIPPING</td>
<td>CLIP</td>
<td>PIXEL</td>
<td>PX</td>
</tr>
<tr>
<td>CLOSE</td>
<td>C</td>
<td>POLYLINE</td>
<td>PL</td>
</tr>
<tr>
<td>COLOUR</td>
<td>C</td>
<td>POLYMARKER</td>
<td>PM</td>
</tr>
<tr>
<td>CONNECTION</td>
<td>C</td>
<td>PRECISION</td>
<td>P</td>
</tr>
<tr>
<td>COPY</td>
<td>C</td>
<td>READ</td>
<td>RD</td>
</tr>
<tr>
<td>CREATE</td>
<td>CR</td>
<td>RECORD</td>
<td>REC</td>
</tr>
<tr>
<td>DEACTIVATE</td>
<td>DA</td>
<td>REDRAW</td>
<td>R</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>D</td>
<td>REFERENCE</td>
<td>RF</td>
</tr>
<tr>
<td>DEFERRAL</td>
<td>D</td>
<td>RENAME</td>
<td>REN</td>
</tr>
<tr>
<td>DELETE</td>
<td>D</td>
<td>QUEUE</td>
<td>Q</td>
</tr>
<tr>
<td>DETECTABILITY</td>
<td>DTEC</td>
<td>READ</td>
<td>RD</td>
</tr>
<tr>
<td>DIMENSIONS</td>
<td>D</td>
<td>REDRAW</td>
<td>R</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>D</td>
<td>REFERENCE</td>
<td>RF</td>
</tr>
<tr>
<td>DRAW</td>
<td>D</td>
<td>RENAME</td>
<td>REN</td>
</tr>
<tr>
<td>DYNAMIC</td>
<td>D</td>
<td>REPRESENTATION</td>
<td>R</td>
</tr>
<tr>
<td>element</td>
<td>E</td>
<td>REQUEST</td>
<td>RQ</td>
</tr>
<tr>
<td>EMERGENCY</td>
<td>E</td>
<td>SAMPLE</td>
<td>SM</td>
</tr>
<tr>
<td>ERROR</td>
<td>ER</td>
<td>SCALE</td>
<td>SC</td>
</tr>
<tr>
<td>ESCAPE</td>
<td>ESC</td>
<td>SEGMENT</td>
<td>SG</td>
</tr>
<tr>
<td>EVALUATE</td>
<td>EV</td>
<td>SELECT</td>
<td>SEL</td>
</tr>
<tr>
<td>EXPANSION</td>
<td>XP</td>
<td>SET</td>
<td>S</td>
</tr>
<tr>
<td>EXTENT</td>
<td>X</td>
<td>SIMULTANEOUS</td>
<td>SIM</td>
</tr>
<tr>
<td>FACILITY</td>
<td>F</td>
<td>SOURCE</td>
<td>S</td>
</tr>
<tr>
<td>FILL</td>
<td>F</td>
<td>SPACE</td>
<td>SP</td>
</tr>
<tr>
<td>FLAG</td>
<td>F</td>
<td>STATE</td>
<td>S</td>
</tr>
<tr>
<td>FONT</td>
<td>F</td>
<td>STRING</td>
<td>ST</td>
</tr>
<tr>
<td>GENERALISED</td>
<td>G</td>
<td>STROKE</td>
<td>SK</td>
</tr>
<tr>
<td>GET</td>
<td>GT</td>
<td>STYLE</td>
<td>S</td>
</tr>
<tr>
<td>GKS</td>
<td>KS</td>
<td>SURFACE</td>
<td>S</td>
</tr>
<tr>
<td>HANDLING</td>
<td>HND</td>
<td>TEXT</td>
<td>TX</td>
</tr>
<tr>
<td>HEIGHT</td>
<td>H</td>
<td>TRANSFORMATION</td>
<td>T</td>
</tr>
<tr>
<td>HIGHLIGHTING</td>
<td>HLT</td>
<td>UNPACK</td>
<td>U</td>
</tr>
<tr>
<td>IDENTIFICATION</td>
<td>ID</td>
<td>UPDATE</td>
<td>U</td>
</tr>
<tr>
<td>INDEX</td>
<td>I</td>
<td>USE</td>
<td>US</td>
</tr>
<tr>
<td>INITIALISE</td>
<td>IN</td>
<td>VALUATOR</td>
<td>VL</td>
</tr>
<tr>
<td>INPUT</td>
<td>I</td>
<td>VIEWPORT</td>
<td>VP</td>
</tr>
<tr>
<td>INQUIRE</td>
<td>Q</td>
<td>VISIBILITY</td>
<td>VIS</td>
</tr>
<tr>
<td>INSERT</td>
<td>IN</td>
<td>WIDTH</td>
<td>W</td>
</tr>
<tr>
<td>INTERIOR</td>
<td>I</td>
<td>WINDOW</td>
<td>WN</td>
</tr>
<tr>
<td>INTERPRET</td>
<td>I</td>
<td>WORKSTATION</td>
<td>WK</td>
</tr>
<tr>
<td>ITEM</td>
<td>ITM</td>
<td>WRITE</td>
<td>W</td>
</tr>
<tr>
<td>LINE</td>
<td>LN</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5. Data Types

In the GKS standard, parameters of several types are used. The following shows the correspondence between the types used in the GKS document and their realisation in a FORTRAN implementation.

<table>
<thead>
<tr>
<th>GKS Data Type</th>
<th>FORTRAN Data Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>I integer</td>
<td>INTEGER</td>
</tr>
<tr>
<td>R real</td>
<td>REAL</td>
</tr>
<tr>
<td>S string</td>
<td></td>
</tr>
</tbody>
</table>

1) In a full FORTRAN 77 subroutine:
   a) INTEGER containing the number of characters returned (for output string argument only)
   b) CHARACTER*(*) containing the string. In addition, if a character string which is an input parameter may reasonably contain no characters, then an INTEGER (≥0) is used to give the number of characters to be passed to the subroutine.

2) In a FORTRAN 77 Subset subroutine:
   a) INTEGER containing the number of characters passed to the subroutine (for input string only, i.e. only one INTEGER needed for output).
   b) INTEGER containing the number of characters returned (for output string argument only).
   c) CHARACTER*80 containing the string.

P point

const x simple_type where simple_type is I or R (vector of values, for example 2xR)
   In non-inquiry functions, separate simple_type parameters are used (note: in GKS, const ≤4)

   In inquiry functions, if const ≤3, separate simple_type parameters are used; if const ≥4, a simple_type array of dimension const is used.

const x P (only occurs in non-inquiry functions)
   Separate REAL parameters, with the X- and Y- coordinates of one point being followed by the X- and Y- coordinates of the next.

const x E (only occurrence in GKS is const = 13)
   An array of INTEGER elements of dimension const is used, each element being an enumeration alternative.

const 1 x const 2 x R (matrix of values, for example 2x3xR)
   REAL array (const 1, const 2)

list of n values of one simple_type (for example nx1)
   1) For input parameter:
      a) INTEGER (input parameter) containing length n of the list (unless the length is already present as a separate GKS parameter, in which case it is not duplicated)
      b) array of dimension n, whose elements are of the appropriate simple_type.

When the length was legally be defined as zero within GKS, the
binding indicates the array dimension by *. The implementation checks that the given length is ≥1.

2) For output parameter in non-inquiry functions:
   a) INTEGER (input parameter) containing the dimension of the array
   b) INTEGER (output parameter) containing the number of elements of the array actually used.
   c) an array whose elements are of the appropriate simple_type. The input dimension being too small is a language binding error condition (error 2001).

   In both cases (input or output), where the simple_type is a point, there is a REAL array for the X-coordinates and another for the Y-coordinates.

3) For inquiry functions, a single call only returns a single element of the list. For a complete list of length n,
   a) INTEGER (input parameter) containing the sequence number of required list element (in the range 0...n).
   b) INTEGER (output parameter) containing the number of items in the list n.
   c) a parameter of the appropriate simple_type containing the requested element.

   If the sequence number given is 0, the requested element returned is undefined, but an error is not indicated thereby; the number of items in the list n is returned. If the sequence number given is <0 or >n, then error 2002 is indicated, the number of items in the list is returned, but the requested element is undefined; the exception to this is when the list size is 0, and in that case an error is not indicated thereby.

4) A complete inquired list is returned from a single call when the maximum size of the list is a small constant m:
   a) INTEGER (output parameter) containing the number of elements of the array actually used.
   b) an array of dimension m, whose elements are of the appropriate simple_type.

list of n values of a compound type (for example, nx4xR):
   This only occurs in an inquiry function. A single call only returns a single element of the list exactly as for the list of values of one simple_type, except that here the requested element is several FORTRAN parameters.

array of integers (in GKS, nxnx1, where the two occurrences of n may have different values).
   This is described more fully below, where the representations of CELL ARRAY, PIXEL ARRAY and PATTERN are described.

an ordered pair of different types (for example (I;E))
   The different types are represented in turn in the FORTRAN parameter list.
Name: INTEGER

1) Workstation Identifier, Segment Name, Pick Identifier:
   An implementation may restrict the range but must at least provide
   all non-negative integers which are available at that implementation.
   Note: the default value for pick identifier is zero.

2) Workstation Type, Connection Identifier, Error File:
   The set of valid values is implementation dependent. The Connection
   Identifier and Error File may be logical unit numbers.

3) GDP Identifier, Escape Identifier:
   The set of legal values is described in the GKS standard.

4) Identification of GKS procedure:
   The range is shown under 'Enumeration Types'.

Enumeration: INTEGER

Note: All values are mapped to the range zero to N-1, where N is
the number of enumeration alternatives. The single exception is
GKS LEVEL, which starts with -3 to ensure compatibility with an
ISO GKS FORTRAN Binding. Except for null values, the order of
the enumeration alternatives is the same as in the GKS document:
null values always appear in the first position. If the integer value
given by the application program is not in the range 0 to N-1, there
is a language binding error condition (error 2000).

Data Record: Represented as a set of scalar values and an array of type CHAR¬
ACTER*80 containing the data. In addition, an INTEGER input
parameter is used to dimension the array. Where the data record is
an output parameter, an additional argument 'number of array ele¬
ments of data record occupied' is needed. There are no scalar
values except where the data record contains values which are com¬
pulsory in GKS.

Note: Data can be read from and written into the data record with
the FORTRAN READ and WRITE statements. Special utility
functions are defined to pack INTEGER, REAL, and CHARAC¬
TER data into the data record and to unpack the data record to the
individual data items (GPREC, GUREC). The content of the
packed data records is implementation dependent, but GPREC must
perform the inverse function to GUREC and vice versa.

The representation of CELL ARRAY, PIXEL ARRAY, and PATTERN allows the user of the
routines requiring a cell array parameter to pass any portion of the array as an argument. Two
examples should make this clear.

Certainly the user can pass an entire two-dimensional array. In this case the number of columns
of the cell array is the same as the first dimension of the FORTRAN array:

INTEGER DIMX, DIMY, CELLS (DIMX,DIMY)
CALL GCA (XI, Y1, X2, Y2, DIMX, DIMY, 1, 1, DIMX, DIMY, CELLS)
To use an arbitrary portion of an array the user passes the upper left corner of the portion as starting address and the first dimension of the entire array for the proper treatment of addresses. The area inside the small box is the cell array being passed:

```
INTEGER STARTX, STARTY, DX, DY, DIMX, DIMY, CELLS (DIMX,DIMY)
DATA STARTX/3/, STARTY/6/, DX/2/, DY/3/
CALL GCA (X1,Y1,X2,Y2,DIMX,DIMY,STARTX,STARTY,DY,CELLS)
```

```
(1,1)  (2,1)  (3,1)  (4,1)  ...  (DIMX,1)
(1,2)  (2,2)  (3,2)  (4,2)  ...  (DIMX,2)
  :    :    :    :    :    :    :
(1,DIMY)  (2,DIMY)  (3,DIMY)  (4,DIMY)  ...  (DIMX,DIMY)
```
### 6. Enumeration Types

All the enumeration types of GKS are mapped to FORTRAN INTEGERS. The correspondence between GKS scalars and FORTRAN INTEGERS is shown below in a list of symbolic FORTRAN constants which may be included in any application program. The following section contains a method of mapping GKS enumeration types to FORTRAN variable names. In a FORTRAN 77 Subset implementation, this mapping could be accomplished by the DATA statement. Also, a numbering of all GKS functions is given for use in the error handling procedures.

#### Mnemonic FORTRAN names and their values for GKS ENUMERATION type values:

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Source</th>
<th>Bundled</th>
<th>Individual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GBUNDL, GINDIV</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GBUNDL=0, GINDIV=1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Clear control flag</th>
<th>Conditionally</th>
<th>Always</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GCONDI, GALWAY</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GCONDI=0, GALWAY=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Clipping indicator</th>
<th>No clip</th>
<th>Clip</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GNCLIP, GCLIP</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GNCLIP=0, GCLIP=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Colour available</th>
<th>Monochrome</th>
<th>Colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GMONOC, GCOLOR</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GMONOC=0, GCOLOR=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Coordinate switch</th>
<th>WC</th>
<th>NDC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GWC, GNDC</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GWC=0, GNDC=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Deferral mode</th>
<th>ASAP, BNIG, BNIL, ASTI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GASAP, GBNIG, GBNIL, GASTI</td>
</tr>
<tr>
<td>Parameter</td>
<td>GASAP=0, GBNIG=1, GBNIL=2, GASTI=3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Detectability</th>
<th>Undetectable</th>
<th>Detectable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GUNDET, GDETEC</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GUNDET=0, GDETEC=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Device coordinate units</th>
<th>Metres</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GMETRE, GOTHU</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GMETRE=0, GOTHU=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Display surface empty</th>
<th>Not empty</th>
<th>Empty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GNEMPT, GEMPTY</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GNEMPT=0, GEMPTY=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dynamic modification</th>
<th>IRG</th>
<th>IMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GIRG, GIMM</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GIRG=0, GIMM=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Echo switch</th>
<th>No echo</th>
<th>Echo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GNECHO, GECHO</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GNECHO=0, GECHO=1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Fill area interior style</th>
<th>Hollow</th>
<th>Solid</th>
<th>Pattern</th>
<th>Hatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>GHOLLO, GSOLID, GPATTR, GHATCH</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>GHOLLO=0, GSOLID=1, GPATTR=2, GHATCH=3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Enumeration Types

highlighting normal, highlighted
INTEGER GNORML, GHILIT
PARAMETER ( GNORML=0, GHILIT=1 )

input device status none, ok, nopick, nochoice
INTEGER GNONE, GOK, GNPICK, GNCHOI
PARAMETER ( GNONE=0, GOK=1, GNPICK=2, GNCHOI=2 )

input class none, locator, stroke, valuator, choice, pick
INTEGER GNCLAS, GLOCAT, GSTROK, GVALUA, GCHOIC, GPICK,
PARAMETER ( GNCLAS=0, GLOCAT=1, GSTROK=2, GVALUA=3, GCHOIC=4, GPICK=5,
GSTRIN=6 )

implicit regeneration mode suppressed, allowed
INTEGER GSUPPD, GALLOW
PARAMETER ( GSUPPD=0, GALLOW=1 )

level of GKS Lma, Lmb, Lmc, L0a, L0b, L0c,
INTEGER GLMA, GLMB, GLMC, GL0A, GL0B, GL0C,
* GLIA, GLIB, GL1C, GL2A, GL2B, GL2C
PARAMETER ( GLMA=-3, GLMB=-2, GLMC=-1, GL0A=0, GL0B=1, GL0C=2,
GLIA=3, GLIB=4, GL1C=5, GL2A=6, GL2B=7, GL2C=8 )

new frame action necessary no, yes
INTEGER GNO, GYES
PARAMETER ( GNO=0, GYES=1 )

operating mode request, sample, event
INTEGER GREQU, GSAMPL, GEVENT
PARAMETER ( GREQU=0, GSAMPL=1, GEVENT=2 )

operating state value (see GKS 4 111)
GKCL, GKOP, WSOP, WSAC, SGOP
INTEGER GGKCL, GGKOP, GWSOP, GWSAC, GSGOP
PARAMETER ( GGKCL=0, GGKOP=1, GWSOP=2, GWSAC=3, GSGOP=4 )

presence of invalid values absent, present
INTEGER GABSNT, GPRSNT
PARAMETER ( GABSNT=0, GPRSNT=1 )

regeneration flag postpone, perform
INTEGER GPOSTP, GPERFO
PARAMETER ( GPOSTP=0, GPERFO=1 )

relative input priority higher, lower
INTEGER GHIGHR, GLOWER
PARAMETER ( GHIGHR=0, GLOWER=1 )

simultaneous events flag nomore, more
INTEGER GNMORE, GMORE
PARAMETER ( GNMORE=0, GMORE=1 )

text alignment horizontal normal, left, center, right
INTEGER GAHNOR, GALEFT, GACENT, GARITE
PARAMETER ( GAHNOR=0, GALEFT=1, GACENT=2, GARITE=3 )

text alignment vertical normal, top, cap, half, base, bottom
INTEGER GAVNOR, GATOP, GACAP, GAHALF, GABASE, GABOTT
PARAMETER ( GAVNOR=0, GATOP=1, GACAP=2, GAHALF=3, GABASE=4, GABOTT=5 )
## Enumeration Types

<table>
<thead>
<tr>
<th>Text Path</th>
<th>Right, left, up, down</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GRIGHT, GLEFT, GUP, GDOWN</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GRIGHT=0, GLEFT=1, GUP=2, GDOWN=3)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Text Precision</th>
<th>String, character, stroke</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GSTRP, GCHARP, GSTRKP</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GSTRP=0, GCHARP=1, GSTRKP=2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type of Returned Values</th>
<th>Set, realized</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GSET, GREALI</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GSET=0, GREALI=1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Update State</th>
<th>Notpending, pending</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GNPEND, GPEND</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GNPEND=0, GPEND=1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Vector/Raster/Other Type</th>
<th>Vector, raster, other</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GVECTR, GRASTR, GOTHWK</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GVECTR=0, GRASTR=1, GOTHWK=2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Visibility</th>
<th>Invisible, visible</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GINVIS, GVISI</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GINVIS=0, GVISI=1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Workstation Category</th>
<th>(see GKS 4.5.1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GOUTPT, GINPUT, GOUTIN, GWISS, GMO, GMI</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GOUTPT=0, GINPUT=1, GOUTIN=2, GWISS=3, GMO=4, GMI=5)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Workstation State</th>
<th>Inactive, active</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GINACT, GACTIV</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GINACT=0, GACTIV=1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>List of GDP Attributes</th>
<th>Polyline attribute, polymarker attribute, text attribute, fill area attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GPLATT, GPMATT, GTXATT, GFAATT</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GPLATT=0, GPMATT=1, GTXATT=2, GFAATT=3)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line Type</th>
<th>Solid, dash, dot, dash-dot</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GLSOLI, GLDASH, GLDOT, GLDASD</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GLSOLI=1, GLDASH=2, GLDOT=3, GLDASD=4)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Marker Type</th>
<th>' ', '+ ', 'o ', 'x '</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GPOINT, GPLUS, GAST, GOMARK, GXMARK</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GPOINT=1, GPLUS=2, GAST=3, GOMARK=4, GXMARK=5)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attribute Control Flag</th>
<th>Current, specified</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GCURNT, GSPEC</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GCURNT=0, GSPEC=1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Polyline/Fill Area Control Flag</th>
<th>Polyline, fill area</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GPLINE, GFILLA</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GPLINE=0, GFILLA=1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Initial Choice Prompt Flag</th>
<th>Off, on</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTEGER</td>
<td>GPROFF, GPRON</td>
</tr>
<tr>
<td>PARAMETER</td>
<td>(GPROFF=0, GPRON=1)</td>
</tr>
</tbody>
</table>
Enumetion Types

GKS functions -
These names are used for error handling. The names are the same as the GKS function names except that the sentinel character 'G' is replaced by 'E'. The same function identification is used for both full FORTRAN 77 and FORTRAN 77 Subset.

<table>
<thead>
<tr>
<th>INTEGER</th>
<th>EOPKS,</th>
<th>ECLKS,</th>
<th>EOPWK,</th>
<th>ECLWK,</th>
<th>EACWK</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARAMETER ( EOPKS = 0,</td>
<td>ECLKS = 1,</td>
<td>EOPWK = 2,</td>
<td>ECLWK = 3,</td>
<td>EACWK = 4 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>EDAWK,</td>
<td>ECLRWK,</td>
<td>ERSG WK,</td>
<td>EUWK,</td>
<td>ESDS</td>
</tr>
<tr>
<td>PARAMETER ( EDAWK = 5,</td>
<td>ECLRWK = 6,</td>
<td>ERSG WK = 7,</td>
<td>EUWK = 8,</td>
<td>ESDS = 9 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>MSG,</td>
<td>ESC,</td>
<td>EPL,</td>
<td>EFM,</td>
<td>ETX</td>
</tr>
<tr>
<td>PARAMETER ( MSG = 10,</td>
<td>ESC = 11,</td>
<td>EPL = 12,</td>
<td>EFM = 13,</td>
<td>ETX = 14 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>EFA,</td>
<td>ECA,</td>
<td>EGDF,</td>
<td>ESPL,</td>
<td>ESLN</td>
</tr>
<tr>
<td>PARAMETER ( EFA = 15,</td>
<td>ECA = 16,</td>
<td>EGDF = 17,</td>
<td>ESPL = 18,</td>
<td>ESLN = 19 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESLWSC,</td>
<td>ESPLCI,</td>
<td>ESPI,</td>
<td>ESMK,</td>
<td>ESMKSC</td>
</tr>
<tr>
<td>PARAMETER ( ESLWSC = 20,</td>
<td>ESPLCI = 21,</td>
<td>ESPI = 22,</td>
<td>ESMK = 23,</td>
<td>ESMKSC = 24 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESLMCI,</td>
<td>ESTXI,</td>
<td>ESTXFP,</td>
<td>ESCXHP,</td>
<td>ESCISH</td>
</tr>
<tr>
<td>PARAMETER ( ESLMCI = 25,</td>
<td>ESTXI = 26,</td>
<td>ESTXFP = 27,</td>
<td>ESCXHP = 28,</td>
<td>ESCISH = 29 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESTXCI,</td>
<td>ESCHH,</td>
<td>ESCHUP,</td>
<td>ESTXP,</td>
<td>ESTXAL</td>
</tr>
<tr>
<td>PARAMETER ( ESTXCI = 30,</td>
<td>ESCHH = 31,</td>
<td>ESCHUP = 32,</td>
<td>ESTXP = 33,</td>
<td>ESTXAL = 34 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESFAI,</td>
<td>ESFAI,</td>
<td>ESFAI,</td>
<td>ESFAI,</td>
<td>ESPF</td>
</tr>
<tr>
<td>PARAMETER ( ESFAI = 35,</td>
<td>ESFAI = 36,</td>
<td>ESFAI = 37,</td>
<td>ESFAI = 38,</td>
<td>ESPF = 39 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESFARF,</td>
<td>ESAF,</td>
<td>ESPKID,</td>
<td>ESPLR,</td>
<td>ESPMR</td>
</tr>
<tr>
<td>PARAMETER ( ESFARF = 40,</td>
<td>ESAF = 41,</td>
<td>ESPKID = 42,</td>
<td>ESPLR = 43,</td>
<td>ESPMR = 44 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESTXR,</td>
<td>ESFAR,</td>
<td>ESPAR,</td>
<td>ESCHR,</td>
<td>ESWN</td>
</tr>
<tr>
<td>PARAMETER ( ESTXR = 45,</td>
<td>ESFAR = 46,</td>
<td>ESPAR = 47,</td>
<td>ESCHR = 48,</td>
<td>ESWN = 49 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESVP,</td>
<td>ESVPF,</td>
<td>ESCLNT,</td>
<td>ESCLIP,</td>
<td>ESWKN</td>
</tr>
<tr>
<td>PARAMETER ( ESVF = 50,</td>
<td>ESVPF = 51,</td>
<td>ESCLNT = 52,</td>
<td>ESCLIP = 53,</td>
<td>ESWKN = 54 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESWKVP,</td>
<td>ECRSG,</td>
<td>ECLSG,</td>
<td>ERENSG,</td>
<td>EDSG</td>
</tr>
<tr>
<td>PARAMETER ( ESWKVP = 55,</td>
<td>ECRSG = 56,</td>
<td>ECLSG = 57,</td>
<td>ERENSG = 58,</td>
<td>EDSG = 59 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>EDSGWK,</td>
<td>EASGWK,</td>
<td>ECSGWK,</td>
<td>EINSG,</td>
<td>ESSGT</td>
</tr>
<tr>
<td>PARAMETER ( EDSGWK = 60,</td>
<td>EASGWK = 61,</td>
<td>ECSGWK = 62,</td>
<td>EINSG = 63,</td>
<td>ESSGT = 64 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESVIS,</td>
<td>ESHLIT,</td>
<td>ESSGP,</td>
<td>ESDTEC,</td>
<td>EINLC</td>
</tr>
<tr>
<td>PARAMETER ( ESVIS = 65,</td>
<td>ESHLIT = 66,</td>
<td>ESSGP = 67,</td>
<td>ESDTEC = 68,</td>
<td>EINLC = 69 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>EINSK,</td>
<td>EINVL,</td>
<td>EINCH,</td>
<td>EINPK,</td>
<td>EINST</td>
</tr>
<tr>
<td>PARAMETER ( EINSK = 70,</td>
<td>EINVL = 71,</td>
<td>EINCH = 72,</td>
<td>EINPK = 73,</td>
<td>EINST = 74 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESLCM,</td>
<td>ESSKM,</td>
<td>ESVLM,</td>
<td>ESCHM,</td>
<td>ESKU</td>
</tr>
<tr>
<td>PARAMETER ( ESLCM = 75,</td>
<td>ESSKM = 76,</td>
<td>ESVLM = 77,</td>
<td>ESCHM = 78,</td>
<td>ESKU = 79 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESSTM,</td>
<td>ERQC,</td>
<td>ERQLC,</td>
<td>ERQVL,</td>
<td>ERQCH</td>
</tr>
<tr>
<td>PARAMETER ( ESSTM = 80,</td>
<td>ERQC = 81,</td>
<td>ERQLC = 82,</td>
<td>ERQVL = 83,</td>
<td>ERQCH = 84 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ERQPK,</td>
<td>ERQST,</td>
<td>ESMLC,</td>
<td>ESMK,</td>
<td>ESMVL</td>
</tr>
<tr>
<td>PARAMETER ( ERQPK = 85,</td>
<td>ERQST = 86,</td>
<td>ESMLC = 87,</td>
<td>ESMK = 88,</td>
<td>ESMVL = 89 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>ESMCH,</td>
<td>ESMFK,</td>
<td>EMSST,</td>
<td>EWAIT,</td>
<td>EFLUSH</td>
</tr>
<tr>
<td>PARAMETER ( ESMCH = 90,</td>
<td>ESMFK = 91,</td>
<td>EMSST = 92,</td>
<td>EWAIT = 93,</td>
<td>EFLUSH = 94 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>EGTL,</td>
<td>EGTSK,</td>
<td>EGTVL,</td>
<td>EGTC,</td>
<td>EGTPK</td>
</tr>
<tr>
<td>PARAMETER ( EGTL = 95,</td>
<td>EGTSK = 96,</td>
<td>EGTVL = 97,</td>
<td>EGTC = 98,</td>
<td>EGTPK = 99 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>EGDT,</td>
<td>EWITM,</td>
<td>EGTTIM,</td>
<td>ERDITM,</td>
<td>EHTM</td>
</tr>
<tr>
<td>PARAMETER ( EGDT = 100,</td>
<td>EWITM = 101,</td>
<td>EGTTIM = 102,</td>
<td>ERDITM = 103,</td>
<td>EHTM = 104 )</td>
<td></td>
</tr>
<tr>
<td>INTEGER</td>
<td>EMT,</td>
<td>EACTM,</td>
<td>EFREC,</td>
<td>EUREC</td>
<td></td>
</tr>
<tr>
<td>PARAMETER ( EMT = 105,</td>
<td>EACTM = 106,</td>
<td>EFREC = 107,</td>
<td>EUREC = 108 )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# List of the GKS Function Names and Levels

The complete list of GKS function names and their levels follows. Note that in nearly all cases the level matches that of the GKS document, but in some cases where one function in the GKS document has been mapped into many in the FORTRAN Binding, the levels may have been adjusted as appropriate.

## 7.1 List Ordered by Bound Names

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Bound Names</th>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GACTM</td>
<td>AC-T-M</td>
<td>1a</td>
<td>Accumulate Transformation Matrix</td>
</tr>
<tr>
<td>GACWK</td>
<td>AC-WK</td>
<td>ma</td>
<td>Activate Workstation</td>
</tr>
<tr>
<td>GASGWK</td>
<td>A-SG-WK</td>
<td>2a</td>
<td>Associate Segment with Workstation</td>
</tr>
<tr>
<td>GCA</td>
<td>C-A</td>
<td>0a</td>
<td>Cell Array</td>
</tr>
<tr>
<td>GCLKS</td>
<td>CL-KS</td>
<td>ma</td>
<td>Close GKS</td>
</tr>
<tr>
<td>GCLRWK</td>
<td>CLR-WK</td>
<td>ma</td>
<td>Clear Workstation</td>
</tr>
<tr>
<td>GCLSG</td>
<td>CL-SG</td>
<td>1a</td>
<td>Close Segment</td>
</tr>
<tr>
<td>GCLWK</td>
<td>CL-WK</td>
<td>ma</td>
<td>Close Workstation</td>
</tr>
<tr>
<td>GCRSG</td>
<td>CR-SG</td>
<td>1a</td>
<td>Create Segment</td>
</tr>
<tr>
<td>GCSGWK</td>
<td>C-SG-WK</td>
<td>2a</td>
<td>Copy Segment to Workstation</td>
</tr>
<tr>
<td>GDWK</td>
<td>DA-WK</td>
<td>ma</td>
<td>Deactivate Workstation</td>
</tr>
<tr>
<td>GDS</td>
<td>D-SG</td>
<td>1a</td>
<td>Delete Segment</td>
</tr>
<tr>
<td>GDSGWK</td>
<td>D-SG-WK</td>
<td>1a</td>
<td>Delete Segment from Workstation</td>
</tr>
<tr>
<td>GECLKS</td>
<td>E-CL-KS</td>
<td>0a</td>
<td>Emergency Close GKS</td>
</tr>
<tr>
<td>GERHND</td>
<td>ER-HND</td>
<td>0a</td>
<td>Error Handling</td>
</tr>
<tr>
<td>GERLOG</td>
<td>ER-LOG</td>
<td>0a</td>
<td>Error Logging</td>
</tr>
<tr>
<td>GESC</td>
<td>ESC</td>
<td>ma</td>
<td>Escape</td>
</tr>
<tr>
<td>GEVTM</td>
<td>EV-T-M</td>
<td>1a</td>
<td>Evaluate Transformation Matrix</td>
</tr>
<tr>
<td>GFA</td>
<td>F-A</td>
<td>ma</td>
<td>Fill Area</td>
</tr>
<tr>
<td>GFUSH</td>
<td>F-LUSH</td>
<td>mc</td>
<td>Flush Device Events</td>
</tr>
<tr>
<td>GGD</td>
<td>G-D-P</td>
<td>ma</td>
<td>Generalized Drawing Primitive</td>
</tr>
<tr>
<td>GGTCH</td>
<td>GT-CH</td>
<td>mc</td>
<td>Get Choice</td>
</tr>
<tr>
<td>GGTITM</td>
<td>GT-ITM</td>
<td>0a</td>
<td>Get Item Type from GKSM</td>
</tr>
<tr>
<td>GGTLC</td>
<td>GT-LC</td>
<td>mc</td>
<td>Get Locator</td>
</tr>
<tr>
<td>GGTTFK</td>
<td>GT-FK</td>
<td>1c</td>
<td>Get Pick</td>
</tr>
<tr>
<td>GGTSK</td>
<td>GT-SK</td>
<td>mc</td>
<td>Get Stroke</td>
</tr>
<tr>
<td>GGTS</td>
<td>GT-ST</td>
<td>mc</td>
<td>Get String</td>
</tr>
<tr>
<td>GGTVL</td>
<td>GT-VL</td>
<td>mc</td>
<td>Get Valuator</td>
</tr>
<tr>
<td>GII</td>
<td>I-ITM</td>
<td>0a</td>
<td>Interpret Item</td>
</tr>
<tr>
<td>GINCH</td>
<td>IN-CH</td>
<td>mb</td>
<td>Initialise Choice</td>
</tr>
<tr>
<td>GINLC</td>
<td>IN-LC</td>
<td>mb</td>
<td>Initialise Locator</td>
</tr>
<tr>
<td>GINFK</td>
<td>IN-FK</td>
<td>1b</td>
<td>Initialise Pick</td>
</tr>
<tr>
<td>GINS</td>
<td>IN-SG</td>
<td>2a</td>
<td>Insert Segment</td>
</tr>
<tr>
<td>GINSK</td>
<td>IN-SK</td>
<td>mb</td>
<td>Initialise Stroke</td>
</tr>
<tr>
<td>GINST</td>
<td>IN-ST</td>
<td>mb</td>
<td>Initialise String</td>
</tr>
<tr>
<td>GINVL</td>
<td>IN-VL</td>
<td>mb</td>
<td>Initialise Valuator</td>
</tr>
<tr>
<td>GMSG</td>
<td>MSG</td>
<td>1a</td>
<td>Message</td>
</tr>
<tr>
<td>GMSGS</td>
<td>MSG-S</td>
<td>1a</td>
<td>Message (FORTRAN 77 Subset)</td>
</tr>
<tr>
<td>GOPKS</td>
<td>OP-KS</td>
<td>ma</td>
<td>Open GKS</td>
</tr>
<tr>
<td>GOPWK</td>
<td>OP-WK</td>
<td>ma</td>
<td>Open Workstation</td>
</tr>
<tr>
<td>GPL</td>
<td>PL</td>
<td>ma</td>
<td>Polyline</td>
</tr>
<tr>
<td>GPM</td>
<td>PM</td>
<td>ma</td>
<td>Polymarker</td>
</tr>
<tr>
<td>GPRE</td>
<td>P-REC</td>
<td>0a</td>
<td>Pack Data Record</td>
</tr>
<tr>
<td>GQA</td>
<td>Q-A-C-WK</td>
<td>1a</td>
<td>Inquire Set Member of Active Workstations</td>
</tr>
<tr>
<td>GQASF</td>
<td>Q-A-S-F</td>
<td>0a</td>
<td>Inquire Aspect Source Flags (Inquire Current Individual Attribute Values)</td>
</tr>
<tr>
<td>GQASWK</td>
<td>Q-A-S-WK</td>
<td>1a</td>
<td>Inquire Set Member of Associated Workstations</td>
</tr>
<tr>
<td>GQC</td>
<td>Q-C-P</td>
<td>ma</td>
<td>Inquire Colour Facilities</td>
</tr>
<tr>
<td>GQCHB</td>
<td>Q-CH-B</td>
<td>0a</td>
<td>Inquire Character Base Vector (Inquire Current Primitive Attribute Values)</td>
</tr>
<tr>
<td>GQCCH</td>
<td>Q-CH-H</td>
<td>ma</td>
<td>Inquire Character Height (Inquire Current Primitive Attribute Values)</td>
</tr>
<tr>
<td>GQCHS</td>
<td>Q-CH-S</td>
<td>mb</td>
<td>Inquire Choice Device State</td>
</tr>
<tr>
<td>GQCHSP</td>
<td>Q-CH-SP</td>
<td>0a</td>
<td>Inquire Character Spacing (Inquire Current Individual Attribute Values)</td>
</tr>
<tr>
<td>Bound Name</td>
<td>Code</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>GQCHUP</td>
<td>Q-CH-UP</td>
<td>INQUIRE CHARACTER UP VECTOR (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQCHW</td>
<td>Q-CH-W</td>
<td>INQUIRE CHARACTER WIDTH (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQCHXP</td>
<td>Q-CH-XP</td>
<td>INQUIRE CHARACTER EXPANSION FACTOR (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQLIP</td>
<td>Q-LIP</td>
<td>INQUIRE CLIPPING INDICATOR</td>
<td></td>
</tr>
<tr>
<td>GQCNTN</td>
<td>Q-C-N-T-N</td>
<td>INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER</td>
<td></td>
</tr>
<tr>
<td>GQCNR</td>
<td>Q-C-R</td>
<td>INQUIRE COLOUR REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQDCD</td>
<td>Q-D-CD</td>
<td>INQUIRE DEFAULT CHOICE DEVICE DATA</td>
<td></td>
</tr>
<tr>
<td>GQDDS</td>
<td>Q-D-DS</td>
<td>INQUIRE DEFAULT DEFFERED STATE VALUES</td>
<td></td>
</tr>
<tr>
<td>GQDLC</td>
<td>Q-D-CLC</td>
<td>INQUIRE DEFAULT LOCATOR DEVICE DATA</td>
<td></td>
</tr>
<tr>
<td>GQDFK</td>
<td>Q-D-FK</td>
<td>INQUIRE DEFAULT PICK DEVICE DATA</td>
<td></td>
</tr>
<tr>
<td>GQDSGA</td>
<td>Q-D-SGA</td>
<td>INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES</td>
<td></td>
</tr>
<tr>
<td>GQDSK</td>
<td>Q-D-SK</td>
<td>INQUIRE DEFAULT STROKE DEVICE DATA</td>
<td></td>
</tr>
<tr>
<td>GQDSP</td>
<td>Q-D-SP</td>
<td>INQUIRE DISPLAY SPACE SIZE</td>
<td></td>
</tr>
<tr>
<td>GQDST</td>
<td>Q-D-ST</td>
<td>INQUIRE DEFAULT STRING DEVICE DATA</td>
<td></td>
</tr>
<tr>
<td>GQDV</td>
<td>Q-D-VL</td>
<td>INQUIRE DEFAULT VALUATOR DEVICE DATA</td>
<td></td>
</tr>
<tr>
<td>GQDWKA</td>
<td>Q-D-WKA</td>
<td>INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES</td>
<td></td>
</tr>
<tr>
<td>GQECI</td>
<td>Q-E-CI</td>
<td>INQUIRE LIST element of COLOUR INDICES</td>
<td></td>
</tr>
<tr>
<td>GQEFIAI</td>
<td>Q-E-FIAI</td>
<td>INQUIRE LIST element of FILL AREA INDICES</td>
<td></td>
</tr>
<tr>
<td>GQEGDP</td>
<td>Q-E-GDP</td>
<td>INQUIRE LIST element of AVAILABLE GENERALIZED DRAWING PRIMITIVES</td>
<td></td>
</tr>
<tr>
<td>GQENTN</td>
<td>Q-E-N-T-N</td>
<td>INQUIRE LIST element of NORMALIZATION TRANSFORMATION NUMBERS</td>
<td></td>
</tr>
<tr>
<td>GQEPAI</td>
<td>Q-E-PAI</td>
<td>INQUIRE LIST element of PATTERN INDICES</td>
<td></td>
</tr>
<tr>
<td>GQEPILI</td>
<td>Q-E-PLI</td>
<td>INQUIRE LIST element of POLYLINE INDICES</td>
<td></td>
</tr>
<tr>
<td>GQEPMI</td>
<td>Q-E-PMI</td>
<td>INQUIRE LIST element of POLYMARKER INDICES</td>
<td></td>
</tr>
<tr>
<td>GQETXI</td>
<td>Q-E-TXI</td>
<td>INQUIRE LIST element of TEXT INDICES</td>
<td></td>
</tr>
<tr>
<td>GQEWK</td>
<td>Q-E-WK</td>
<td>INQUIRE LIST element of AVAILABLE WORKSTATION TYPES</td>
<td></td>
</tr>
<tr>
<td>GQFACI</td>
<td>Q-F-A-CI</td>
<td>INQUIRE FILL AREA COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQFAF</td>
<td>Q-F-AF</td>
<td>INQUIRE FILL AREA FACILITIES</td>
<td></td>
</tr>
<tr>
<td>GQFAI</td>
<td>Q-F-AI</td>
<td>INQUIRE FILL AREA INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQFAIS</td>
<td>Q-F-A-IS</td>
<td>INQUIRE FILL AREA INTERIOR STYLE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQFAR</td>
<td>Q-F-AR</td>
<td>INQUIRE FILL AREA REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQFASI</td>
<td>Q-F-ASI</td>
<td>INQUIRE FILL AREA STYLE INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQGDP</td>
<td>Q-G-D-P</td>
<td>INQUIRE GENERALIZED DRAWING PRIMITIVE</td>
<td></td>
</tr>
<tr>
<td>GQGOV</td>
<td>Q-G-OV</td>
<td>INQUIRE INPUT QUEUE OVERFLOW</td>
<td></td>
</tr>
<tr>
<td>GQLCS</td>
<td>Q-LCS</td>
<td>INQUIRE LOCATOR DEVICE STATE</td>
<td></td>
</tr>
<tr>
<td>GQLI</td>
<td>Q-LI</td>
<td>INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES</td>
<td></td>
</tr>
<tr>
<td>GQLN</td>
<td>Q-LN</td>
<td>INQUIRE LINETYPE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQLVKS</td>
<td>Q-LV-KS</td>
<td>INQUIRE LEVEL OF GKS</td>
<td></td>
</tr>
<tr>
<td>GQLWK</td>
<td>Q-L-WK</td>
<td>INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES</td>
<td></td>
</tr>
<tr>
<td>GQLWSC</td>
<td>Q-LWSC</td>
<td>INQUIRE LINETHICKNESS SCALE FACTOR (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQM</td>
<td>Q-MK</td>
<td>INQUIRE MARKERTYPE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQMKSC</td>
<td>Q-MK-SC</td>
<td>INQUIRE MARKER SIZE SCALE FACTOR (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQMNTN</td>
<td>Q-M-N-T-N</td>
<td>INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER</td>
<td></td>
</tr>
<tr>
<td>GQNT</td>
<td>Q-N-T</td>
<td>INQUIRE NORMALIZATION TRANSFORMATION</td>
<td></td>
</tr>
<tr>
<td>GQOPS</td>
<td>Q-OP-S</td>
<td>INQUIRE OPERATING STATE VALUE</td>
<td></td>
</tr>
<tr>
<td>GQOPS</td>
<td>Q-OP-SG</td>
<td>INQUIRE NAME OF OPEN SEGMENT</td>
<td></td>
</tr>
<tr>
<td>GQOPWK</td>
<td>Q-OP-WK</td>
<td>INQUIRE SET member of OPEN WORKSTATIONS</td>
<td></td>
</tr>
<tr>
<td>GQPA</td>
<td>Q-PA</td>
<td>INQUIRE PATTERN SIZE (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQPAF</td>
<td>Q-P-A-F</td>
<td>INQUIRE PATTERN FACILITIES</td>
<td></td>
</tr>
<tr>
<td>GQPAR</td>
<td>Q-P-A-R</td>
<td>INQUIRE PATTERN REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPARF</td>
<td>INQUIRE PATTERN REFERENCE POINT (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPCR</td>
<td>INQUIRE PATTERN REFERENCE COLOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQFFAR</td>
<td>INQUIRE PREDEFINED FILL AREA REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPKD</td>
<td>INQUIRE PICK IDENTIFIER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPKS</td>
<td>INQUIRE PICK DEVICE STATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPLCI</td>
<td>INQUIRE POLYLINE COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPLF</td>
<td>INQUIRE POLYLINE FACILITIES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPLI</td>
<td>INQUIRE POLYLINE INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPKID</td>
<td>INQUIRE POLYLINE REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPKS</td>
<td>INQUIRE POLYLINE COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPKF</td>
<td>INQUIRE POLYLINE FACILITIES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPKM</td>
<td>INQUIRE POLYLINE INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPKR</td>
<td>INQUIRE POLYLINE REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPMCI</td>
<td>INQUIRE POLYMARKER COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPMF</td>
<td>INQUIRE POLYMARKER FACILITIES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPMI</td>
<td>INQUIRE POLYMARKER INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQPMR</td>
<td>INQUIRE POLYMARKER REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQFFA</td>
<td>INQUIRE PREDEFINED PATTERN REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQFFP</td>
<td>INQUIRE PREDEFINED POLYLINE REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQFFM</td>
<td>INQUIRE PREDEFINED POLYMARKER REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQFPT</td>
<td>INQUIRE PREDEFINED TEXT REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQFX</td>
<td>INQUIRE PIXEL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQFXA</td>
<td>INQUIRE PIXEL ARRAY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQFXAD</td>
<td>INQUIRE PIXEL ARRAY DIMENSIONS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQSGA</td>
<td>INQUIRE SEGMENT ATTRIBUTES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQSGP</td>
<td>INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQSGUS</td>
<td>INQUIRE SET MEMBER OF SEGMENT NAMES IN USE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQSGWK</td>
<td>INQUIRE SET MEMBER OF SEGMENT NAMES ON WORKSTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQSIM</td>
<td>INQUIRE MORE SIMULTANEOUS EVENTS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQSKS</td>
<td>INQUIRE STROKE DEVICE STATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQSST</td>
<td>INQUIRE STRING DEVICE STATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXAL</td>
<td>INQUIRE TEXT ALIGNMENT (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXCI</td>
<td>INQUIRE TEXT COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXF</td>
<td>INQUIRE TEXT FACILITIES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXFP</td>
<td>INQUIRE TEXT FONT AND PRECISION (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXI</td>
<td>INQUIRE TEXT INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXP</td>
<td>INQUIRE TEXT PATH (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXR</td>
<td>INQUIRE TEXT REPRESENTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXX</td>
<td>INQUIRE TEXT EXTENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQTXXS</td>
<td>INQUIRE TEXT EXTENT (FORTRAN 77 SUBSET)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQVLS</td>
<td>INQUIRE VALUATOR DEVICE STATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQWK</td>
<td>INQUIRE WORKSTATION CONNECTION AND TYPE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQWKCA</td>
<td>INQUIRE WORKSTATION CLASSIFICATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQWKDL</td>
<td>INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQWKD</td>
<td>INQUIRE WORKSTATION MAXIMUM NUMBERS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQWKS</td>
<td>INQUIRE WORKSTATION STATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GQWT</td>
<td>INQUIRE WORKSTATION TRANSFORMATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRSITM</td>
<td>READ ITEM FROM GKSM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRENSEG</td>
<td>RENAME SEGMENT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRCQ</td>
<td>REQUEST CHOICE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRQLC</td>
<td>REQUEST LOCATOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRQP</td>
<td>REQUEST PICK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRQSK</td>
<td>REQUEST STROKE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRSST</td>
<td>REQUEST STRING</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRQVL</td>
<td>REQUEST VALUATOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRSGWK</td>
<td>REDRAW ALL SEGMENETS ON WORKSTATION</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GSAF</td>
<td>SET ASPECT SOURCE FLGS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GSCHH</td>
<td>SET CHARACTER HEIGHT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
List Ordered by Bound Names

GSCHM  S-CH-M  mb  SET CHOICE MODE
GSCHSP  S-CH-SP  0a  SET CHARACTER SPACING
GSCHUP  S-CH-UP  ma  SET CHARACTER UP VECTOR
GSCHXP  S-CH-XP  0a  SET CHARACTER EXPANSION FACTOR
GSCLP  S-CLIP  ma  SET CLIPPING INDICATOR
GSCR  S-C-R  ma  SET COLOUR REPRESENTATION
GSDS  S-D-S  la  SET DEFERRAL STATE
GSDTEC  S-DTEC  1b  SET DETECTABILITY
GSELNT  SEL-N-T  ma  SELECT NORMALIZATION TRANSFORMATION
GSFACI  S-F-A-C-I  ma  SET FILL AREA COLOUR INDEX
GSFAI  S-F-A-I  0a  SET FILL AREA INDEX
GSFAIS  S-F-A-I-S  ma  SET FILL AREA INTERIOR STYLE
GSFAR  S-F-A-R  1a  SET FILL AREA REPRESENTATION
GSFASI  S-F-A-S-I  0a  SET FILL AREA STYLE INDEX
GSHLIT  S-HLIT  la  SET HIGHLIGHTING
GSLCM  S-LC-M  mb  SET LOCATOR MODE
GSLN  S-LN  ma  SET LINETYPE
GSLWSC  S-LW-SC  0a  SET LINEWIDTH SCALE FACTOR
GSMCH  SM-CH  mc  SAMPLING MODE
GSMK  SM-K  ma  SET MARKERTYPE
GSMKSC  S-MK-SC  0a  SET MARKER SIZE SCALE FACTOR
GSMLC  SM-LC  mc  SET MARKER LOCATOR
GSMPI  SM-PK  lc  SAMPLE PICK
GSMST  SM-ST  mc  SAMPLE STRING
GSMVL  SM-VL  mc  SAMPLE VALUATOR
GSPA  S-PA  0a  SET PATTERN SIZE
GSPAR  S-PA-R  1a  SET PATTERN REPRESENTATION
GSPARF  S-PA-RF  0a  SET PATTERN REFERENCE POINT
GSPKID  S-PK-ID  1b  SET PICK IDENTIFIER
GSKM  S-PK-M  1b  SET PICK MODE
GSPLCI  S-PL-C-I  ma  SET POLYLINE COLOUR INDEX
GSLI  S-PL-I  0a  SET POLYLINE INDEX
GSLR  S-PL-R  1a  SET POLYLINE REPRESENTATION
GSPMCI  S-PM-C-I  ma  SET POLYMARKER COLOUR INDEX
GSPMI  S-PM-I  0a  SET POLYMARKER INDEX
GSPMR  S-PM-R  1a  SET POLYMARKER REPRESENTATION
GSSGP  S-SG-P  1a  SET SEGMENT PRIORITY
GSSGT  S-SG-T  1a  SET SEGMENT TRANSFORMATION
GSSKM  S-SK-M  mb  SET STROKE MODE
GSTM  S-ST-M  mb  SET STRING MODE
GSTXAL  S-TX-AL  ma  SET TEXT ALIGNMENT
GSTXCI  S-TX-C-I  ma  SET TEXT COLOUR INDEX
GSTXFP  S-TX-F-P  0a  SET TEXT FONT AND PRECISION
GSTXI  S-TX-I  0a  SET TEXT INDEX
GSTXP  S-TX-P  0a  SET TEXT PATH
GSTXR  S-TX-R  1a  SET TEXT REPRESENTATION
GSVIS  S-VIS  1a  SET VISIBILITY
GVL  S-VP  ma  SET VIEWPORT
GSVP  S-VP  ma  SET VIEWPORT
GSVP  S-VP  ma  SET VIEWPORT
GSWKVP  S-WK-VP  ma  SET WORKSTATION VIEWPORT
GSKWN  S-WK-WN  ma  SET WORKSTATION WINDOW
GSWN  S-WN  ma  SET WINDOW
GTX  TX  ma  TEXT
GTXS  TX-S  ma  TEXT (FORTRAN 77 SUBSET)
GUREC  U-REC  0a  UNPACK DATA RECORD
GUW  U-WK  ma  UPDATE WORKSTATION
GWAIT  WAIT  mc  AWAIT EVENT
GWITM  W-ITM  0a  WRITE ITEM TO GKSM
### List Ordered by GKS Function Names

#### 7.2 List Ordered by GKS Function Names

<table>
<thead>
<tr>
<th>Function Code</th>
<th>Description</th>
<th>Argument(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GACTM</td>
<td>Accumulate Transformation Matrix</td>
<td>la</td>
</tr>
<tr>
<td>GACWK</td>
<td>Activate Workstation</td>
<td>ma</td>
</tr>
<tr>
<td>GASGWK</td>
<td>Associate Segment with Workstation</td>
<td>2a</td>
</tr>
<tr>
<td>GWAIT</td>
<td>Await Event</td>
<td>mc</td>
</tr>
<tr>
<td>GCA</td>
<td>Cell Array</td>
<td>0a</td>
</tr>
<tr>
<td>GCLRWK</td>
<td>Clear Workstation</td>
<td>ma</td>
</tr>
<tr>
<td>GCLKS</td>
<td>Close GKS</td>
<td>ma</td>
</tr>
<tr>
<td>GCLSG</td>
<td>Close Segment</td>
<td>ia</td>
</tr>
<tr>
<td>GCLWK</td>
<td>Close Workstation</td>
<td>ma</td>
</tr>
<tr>
<td>GCSGWK</td>
<td>Copy Segment to Workstation</td>
<td>2a</td>
</tr>
<tr>
<td>GCRSG</td>
<td>Create Segment</td>
<td>ia</td>
</tr>
<tr>
<td>GDAWK</td>
<td>Deactivate Workstation</td>
<td>ma</td>
</tr>
<tr>
<td>GDSG</td>
<td>Delete Segment</td>
<td>ia</td>
</tr>
<tr>
<td>GDSGWK</td>
<td>Delete Segment from Workstation</td>
<td>ia</td>
</tr>
<tr>
<td>GECLKS</td>
<td>Emergency Close GKS</td>
<td>0a</td>
</tr>
<tr>
<td>GERHND</td>
<td>Error Handling</td>
<td>0a</td>
</tr>
<tr>
<td>GERLOG</td>
<td>Error Logging</td>
<td>0a</td>
</tr>
<tr>
<td>GESC</td>
<td>Escape</td>
<td>ma</td>
</tr>
<tr>
<td>GEVTM</td>
<td>Evaluate Transformation Matrix</td>
<td>1a</td>
</tr>
<tr>
<td>GFA</td>
<td>Fill Area</td>
<td>ma</td>
</tr>
<tr>
<td>GFLUSH</td>
<td>Flush Device Events</td>
<td>mc</td>
</tr>
<tr>
<td>GGDP</td>
<td>Generalized Drawing Primitive</td>
<td>ma</td>
</tr>
<tr>
<td>GTCH</td>
<td>Get Choice</td>
<td>mc</td>
</tr>
<tr>
<td>GTITM</td>
<td>Get Item Type from GKS</td>
<td>0a</td>
</tr>
<tr>
<td>GTLC</td>
<td>Get Locator</td>
<td>mc</td>
</tr>
<tr>
<td>GTPK</td>
<td>Get Pick</td>
<td>tc</td>
</tr>
<tr>
<td>GTST</td>
<td>Get String</td>
<td>mc</td>
</tr>
<tr>
<td>GTSK</td>
<td>Get Stroke</td>
<td>mc</td>
</tr>
<tr>
<td>GTVL</td>
<td>Get Valuator</td>
<td>mc</td>
</tr>
<tr>
<td>GINCH</td>
<td>Initialise Choice Device State</td>
<td>mb</td>
</tr>
<tr>
<td>GINLCTYPE</td>
<td>Initialise Locator</td>
<td>mb</td>
</tr>
<tr>
<td>GINP</td>
<td>Initialise Pick</td>
<td>1b</td>
</tr>
<tr>
<td>GINST</td>
<td>Initialise String</td>
<td>mb</td>
</tr>
<tr>
<td>GINSK</td>
<td>Initialise Stroke</td>
<td>mb</td>
</tr>
<tr>
<td>GINV</td>
<td>Initialise Valuator</td>
<td>mb</td>
</tr>
<tr>
<td>GQASF</td>
<td>Inquire Aspect Source Flags (Inquire Current Individual Attribute Values)</td>
<td>0a</td>
</tr>
<tr>
<td>GQCHB</td>
<td>Inquire Character Base Vector (Inquire Current Primitive Attribute Values)</td>
<td>0a</td>
</tr>
<tr>
<td>GQCHXP</td>
<td>Inquire Character Expansion Factor</td>
<td>0a</td>
</tr>
<tr>
<td>GQCHH</td>
<td>Inquire Character Height (Inquire Current Primitive Attribute Values)</td>
<td>ma</td>
</tr>
<tr>
<td>GQCHSP</td>
<td>Inquire Character Spacing (Inquire Current Individual Attribute Values)</td>
<td>0a</td>
</tr>
<tr>
<td>GQCHUP</td>
<td>Inquire Character Up Vector (Inquire Current Primitive Attribute Values)</td>
<td>ma</td>
</tr>
<tr>
<td>GQCHW</td>
<td>Inquire Character Width (Inquire Current Primitive Attribute Values)</td>
<td>0a</td>
</tr>
<tr>
<td>GQCHS</td>
<td>Inquire Choice Device State</td>
<td>mb</td>
</tr>
<tr>
<td>GQCLIP</td>
<td>Inquire Clipping Indicator</td>
<td>ma</td>
</tr>
<tr>
<td>GQCF</td>
<td>Inquire Colour Facilities</td>
<td>ma</td>
</tr>
<tr>
<td>GQCR</td>
<td>Inquire Colour Representation</td>
<td>ma</td>
</tr>
<tr>
<td>GQCNTN</td>
<td>Inquire Current Normalization Transformation Number</td>
<td>ma</td>
</tr>
<tr>
<td>GQDCH</td>
<td>Inquire Default Choice Device Data</td>
<td>mb</td>
</tr>
<tr>
<td>GQDDS</td>
<td>Inquire Default Deferral State Values</td>
<td>ia</td>
</tr>
<tr>
<td>GQDLC</td>
<td>Inquire Default Locator Device Data</td>
<td>mb</td>
</tr>
<tr>
<td>GQDFK</td>
<td>Inquire Default Pick Device Data</td>
<td>1b</td>
</tr>
<tr>
<td>GQDST</td>
<td>Inquire Default String Device Data</td>
<td>mb</td>
</tr>
<tr>
<td>GQDSK</td>
<td>Inquire Default Stroke Device Data</td>
<td>mb</td>
</tr>
<tr>
<td>GQDVL</td>
<td>Inquire Default Valuator Device Data</td>
<td>mb</td>
</tr>
<tr>
<td>GQDSP</td>
<td>Inquire Display Space Size</td>
<td>0a</td>
</tr>
<tr>
<td>GQDSGA</td>
<td>Inquire Dynamic Modification of Segment Attributes</td>
<td>1a</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>GQDWKA</td>
<td>INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES</td>
<td></td>
</tr>
<tr>
<td>GQFAICI</td>
<td>INQUIRE FILL AREA COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQFAF</td>
<td>INQUIRE FILL AREA FACILITIES</td>
<td></td>
</tr>
<tr>
<td>GQFAI</td>
<td>INQUIRE FILL AREA INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQFAIS</td>
<td>INQUIRE FILL AREA INTERIOR STYLE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQFAI</td>
<td>INQUIRE FILL AREA REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQFAI</td>
<td>INQUIRE FILL AREA STYLE INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQGD</td>
<td>INQUIRE GENERALIZED DRAWING PRIMITIVES</td>
<td></td>
</tr>
<tr>
<td>GQIQOV</td>
<td>INQUIRE INPUT QUEUE OVERFLOW</td>
<td></td>
</tr>
<tr>
<td>GQLVKS</td>
<td>INQUIRE LEVEL OF GKS</td>
<td></td>
</tr>
<tr>
<td>GQLN</td>
<td>INQUIRE LINETYPE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQLWSC</td>
<td>INQUIRE LINESWIDTH SCALE FACTOR (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQEGDP</td>
<td>INQUIRE LIST ELEMENT OF AVAILABLE GENERALIZED DRAWING PRIMITIVES</td>
<td></td>
</tr>
<tr>
<td>GQEWK</td>
<td>INQUIRE LIST ELEMENT OF AVAILABLE WORKSTATION TYPES</td>
<td></td>
</tr>
<tr>
<td>GQECI</td>
<td>INQUIRE LIST ELEMENT OF COLOUR INDICES</td>
<td></td>
</tr>
<tr>
<td>GQFAS</td>
<td>INQUIRE LIST ELEMENT OF FILL AREA INDICES</td>
<td></td>
</tr>
<tr>
<td>GQENTN</td>
<td>INQUIRE LIST ELEMENT OF NORMALIZATION TRANSFORMATION NUMBERS</td>
<td></td>
</tr>
<tr>
<td>GQFAS</td>
<td>INQUIRE LIST ELEMENT OF PATTERN INDICES</td>
<td></td>
</tr>
<tr>
<td>GQEPPLI</td>
<td>INQUIRE LIST ELEMENT OF POLYLINE INDICES</td>
<td></td>
</tr>
<tr>
<td>GQEPFM</td>
<td>INQUIRE LIST ELEMENT OF POLYMARKER INDICES</td>
<td></td>
</tr>
<tr>
<td>GQETXI</td>
<td>INQUIRE LIST ELEMENT OF TEXT INDICES</td>
<td></td>
</tr>
<tr>
<td>GQLCS</td>
<td>INQUIRE LOCATOR DEVICE STATE</td>
<td></td>
</tr>
<tr>
<td>GQMKSC</td>
<td>INQUIRE MARKER SIZE SCALE FACTOR (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQMK</td>
<td>INQUIRE MARKERTYPE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQLWK</td>
<td>INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES</td>
<td></td>
</tr>
<tr>
<td>GQMNNTN</td>
<td>INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER</td>
<td></td>
</tr>
<tr>
<td>GQSIM</td>
<td>INQUIRE MORE SIMULTANEOUS EVENTS</td>
<td></td>
</tr>
<tr>
<td>GQOPS</td>
<td>INQUIRE NAME OF OPEN SEGMENT</td>
<td></td>
</tr>
<tr>
<td>GQNT</td>
<td>INQUIRE NORMALIZATION TRANSFORMATION</td>
<td></td>
</tr>
<tr>
<td>GQLI</td>
<td>INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES</td>
<td></td>
</tr>
<tr>
<td>GQSGP</td>
<td>INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED</td>
<td></td>
</tr>
<tr>
<td>GQOPS</td>
<td>INQUIRE OPERATING STATE VALUE</td>
<td></td>
</tr>
<tr>
<td>GQPFARF</td>
<td>INQUIRE PATTERN FACILITIES</td>
<td></td>
</tr>
<tr>
<td>GQPA</td>
<td>INQUIRE PATTERN REFERENCE POINT (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQPFS</td>
<td>INQUIRE PATTERN REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQPA</td>
<td>INQUIRE PATTERN SIZE (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQPKS</td>
<td>INQUIRE PICK DEVICE STATE</td>
<td></td>
</tr>
<tr>
<td>GQPKID</td>
<td>INQUIRE PICK IDENTIFIER</td>
<td></td>
</tr>
<tr>
<td>GQFX</td>
<td>INQUIRE PIXEL</td>
<td></td>
</tr>
<tr>
<td>GQFXA</td>
<td>INQUIRE PIXEL ARRAY</td>
<td></td>
</tr>
<tr>
<td>GQFXAD</td>
<td>INQUIRE PIXEL ARRAY DIMENSIONS</td>
<td></td>
</tr>
<tr>
<td>GQPLCI</td>
<td>INQUIRE POLYLINE COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQPLF</td>
<td>INQUIRE POLYLINE FACILITIES</td>
<td></td>
</tr>
<tr>
<td>GQPLI</td>
<td>INQUIRE POLYLINE INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQPLR</td>
<td>INQUIRE POLYLINE REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQPMCI</td>
<td>INQUIRE POLYMARKER COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQPMF</td>
<td>INQUIRE POLYMARKER FACILITIES</td>
<td></td>
</tr>
<tr>
<td>Function Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>GQPMI</td>
<td>INQUIRE POLYMARKER INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQPMR</td>
<td>INQUIRE POLYMARKER REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQPCR</td>
<td>INQUIRE PREDEFINED COLOUR REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQPFAR</td>
<td>INQUIRE PREDEFINED FILL AREA REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQPPAR</td>
<td>INQUIRE PREDEFINED PATTERN REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQPFPLR</td>
<td>INQUIRE PREDEFINED POLYLINE REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQPFMR</td>
<td>INQUIRE PREDEFINED POLYMARKER REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQPTXR</td>
<td>INQUIRE PREDEFINED TEXT REPRESENTATION</td>
<td></td>
</tr>
<tr>
<td>GQSQA</td>
<td>INQUIRE SEGMENT ATTRIBUTES</td>
<td></td>
</tr>
<tr>
<td>GQACWK</td>
<td>INQUIRE SET OF ACTIVE WORKSTATIONS</td>
<td></td>
</tr>
<tr>
<td>GQASKW</td>
<td>INQUIRE SET OF ASSOCIATED WORKSTATIONS</td>
<td></td>
</tr>
<tr>
<td>GQPWFK</td>
<td>INQUIRE SET OF OPEN WORKSTATIONS</td>
<td></td>
</tr>
<tr>
<td>GQSGUS</td>
<td>INQUIRE SET OF SEGMENT NAMES IN USE</td>
<td></td>
</tr>
<tr>
<td>GQSGWK</td>
<td>INQUIRE SET OF SEGMENT NAMES ON WORKSTATION</td>
<td></td>
</tr>
<tr>
<td>GQSTS</td>
<td>INQUIRE STRING DEVICE STATE</td>
<td></td>
</tr>
<tr>
<td>GQSKS</td>
<td>INQUIRE STROKE DEVICE STATE</td>
<td></td>
</tr>
<tr>
<td>GQTXAL</td>
<td>INQUIRE TEXT ALIGNMENT (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQTXCI</td>
<td>INQUIRE TEXT COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQTXXX</td>
<td>INQUIRE TEXT EXTENT</td>
<td></td>
</tr>
<tr>
<td>GQTXXSA</td>
<td>INQUIRE TEXT EXTENT (FORTRAN 77 SUBSET)</td>
<td></td>
</tr>
<tr>
<td>GQTXF</td>
<td>INQUIRE TEXT FACILITIES</td>
<td></td>
</tr>
<tr>
<td>GQTXFP</td>
<td>INQUIRE TEXT FONT AND PRECISION (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQTXI</td>
<td>INQUIRE TEXT INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQTXP</td>
<td>INQUIRE TEXT PATH (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
<td></td>
</tr>
<tr>
<td>GQVLS</td>
<td>INQUIRE VALUATOR DEVICE STATE</td>
<td></td>
</tr>
<tr>
<td>GQWKCA</td>
<td>INQUIRE WORKSTATION CATEGORY</td>
<td></td>
</tr>
<tr>
<td>GQWKCL</td>
<td>INQUIRE WORKSTATION CLASSIFICATION</td>
<td></td>
</tr>
<tr>
<td>GQWKC</td>
<td>INQUIRE WORKSTATION CONNECTION AND TYPE</td>
<td></td>
</tr>
<tr>
<td>GQWKDU</td>
<td>INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES</td>
<td></td>
</tr>
<tr>
<td>GQWKM</td>
<td>INQUIRE WORKSTATION MAXIMUM NUMBERS</td>
<td></td>
</tr>
<tr>
<td>GQWKS</td>
<td>INQUIRE WORKSTATION STATE</td>
<td></td>
</tr>
<tr>
<td>GQWKT</td>
<td>INQUIRE WORKSTATION TRANSFORMATION</td>
<td></td>
</tr>
<tr>
<td>GINS</td>
<td>INSERT SEGMENT</td>
<td></td>
</tr>
<tr>
<td>GIITM</td>
<td>INTERPRET ITEM</td>
<td></td>
</tr>
<tr>
<td>GMSG</td>
<td>MESSAGE</td>
<td></td>
</tr>
<tr>
<td>GMSGS</td>
<td>MESSAGE (FORTRAN 77 SUBSET)</td>
<td></td>
</tr>
<tr>
<td>GOFKS</td>
<td>OPEN GKS</td>
<td></td>
</tr>
<tr>
<td>GOPWK</td>
<td>OPEN WORKSTATION</td>
<td></td>
</tr>
<tr>
<td>GPREC</td>
<td>PACK DATA RECORD</td>
<td></td>
</tr>
<tr>
<td>GPL</td>
<td>POLYLINE</td>
<td></td>
</tr>
<tr>
<td>GFM</td>
<td>POLYMARKER</td>
<td></td>
</tr>
<tr>
<td>GGRDITM</td>
<td>READ ITEM FROM GKSM</td>
<td></td>
</tr>
<tr>
<td>GGRSGWK</td>
<td>REDRAW ALL SEGMENTS ON WORKSTATION</td>
<td></td>
</tr>
<tr>
<td>GRENSEG</td>
<td>RENAME SEGMENT</td>
<td></td>
</tr>
<tr>
<td>GROCH</td>
<td>REQUEST CHOICE</td>
<td></td>
</tr>
<tr>
<td>GROLC</td>
<td>REQUEST LOCATOR</td>
<td></td>
</tr>
<tr>
<td>GROFK</td>
<td>REQUEST PICK</td>
<td></td>
</tr>
<tr>
<td>GROST</td>
<td>REQUEST STRING</td>
<td></td>
</tr>
<tr>
<td>GROSK</td>
<td>REQUEST STROKE</td>
<td></td>
</tr>
<tr>
<td>GROVL</td>
<td>REQUEST VALUATOR</td>
<td></td>
</tr>
<tr>
<td>GSMCH</td>
<td>SAMPLE CHOICE</td>
<td></td>
</tr>
<tr>
<td>GSMCL</td>
<td>SAMPLE LOCATOR</td>
<td></td>
</tr>
<tr>
<td>GSMFK</td>
<td>SAMPLE PICK</td>
<td></td>
</tr>
<tr>
<td>GSMST</td>
<td>SAMPLE STRING</td>
<td></td>
</tr>
<tr>
<td>GSMSK</td>
<td>SAMPLE STROKE</td>
<td></td>
</tr>
<tr>
<td>GSMVL</td>
<td>SAMPLE VALUATOR</td>
<td></td>
</tr>
<tr>
<td>GSELNT</td>
<td>SELECT NORMALIZATION TRANSFORMATION</td>
<td></td>
</tr>
<tr>
<td>GSASF</td>
<td>SET ASPECT SOURCE FLAGS</td>
<td></td>
</tr>
</tbody>
</table>
### List Ordered by GKS Function Names

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GSCHXP</td>
<td>SET CHARACTER EXPANSION FACTOR</td>
</tr>
<tr>
<td>GSCHH</td>
<td>SET CHARACTER HEIGHT</td>
</tr>
<tr>
<td>GSCHSP</td>
<td>SET CHARACTER SPACING</td>
</tr>
<tr>
<td>GSCHUP</td>
<td>SET CHARACTER UP VECTOR</td>
</tr>
<tr>
<td>GSCHM</td>
<td>SET CHOICE MODE</td>
</tr>
<tr>
<td>GSClIP</td>
<td>SET CLIPPING INDICATOR</td>
</tr>
<tr>
<td>GSCR</td>
<td>SET COLOUR REPRESENTATION</td>
</tr>
<tr>
<td>GSdS</td>
<td>SET DEFERRAL STATE</td>
</tr>
<tr>
<td>GSdTEC</td>
<td>SET DETECTABILITY</td>
</tr>
<tr>
<td>GSFACl</td>
<td>SET FILL AREA COLOUR INDEX</td>
</tr>
<tr>
<td>GSfAI</td>
<td>SET FILL AREA INDEX</td>
</tr>
<tr>
<td>GSfAlS</td>
<td>SET FILL AREA INTERIOR STYLE</td>
</tr>
<tr>
<td>GSfAR</td>
<td>SET FILL AREA REPRESENTATION</td>
</tr>
<tr>
<td>GSfASI</td>
<td>SET FILL AREA STYLE INDEX</td>
</tr>
<tr>
<td>GSbLIT</td>
<td>SET HIGHLIGHTING</td>
</tr>
<tr>
<td>GSLN</td>
<td>SET LINETYPE</td>
</tr>
<tr>
<td>GSLWSc</td>
<td>SET LINEWIDTH SCALE FACTOR</td>
</tr>
<tr>
<td>GSLCM</td>
<td>SET LOCATOR MODE</td>
</tr>
<tr>
<td>GS MKSC</td>
<td>SET MARKER SIZE SCALE FACTOR</td>
</tr>
<tr>
<td>GS MK</td>
<td>SET MARKERTYPE</td>
</tr>
<tr>
<td>GSfARF</td>
<td>SET PATTERN REFERENCE POINT</td>
</tr>
<tr>
<td>GSfAR</td>
<td>SET PATTERN REPRESENTATION</td>
</tr>
<tr>
<td>GSfA</td>
<td>SET PATTERN SIZE</td>
</tr>
<tr>
<td>GSfKId</td>
<td>SET PICK IDENTIFIER</td>
</tr>
<tr>
<td>GSfKm</td>
<td>SET PICK MODE</td>
</tr>
<tr>
<td>GSfLCLI</td>
<td>SET POLYLINE COLOUR INDEX</td>
</tr>
<tr>
<td>GSfLI</td>
<td>SET POLYLINE INDEX</td>
</tr>
<tr>
<td>GSfLR</td>
<td>SET POLYLINE REPRESENTATION</td>
</tr>
<tr>
<td>GS PMC1</td>
<td>SET POLYMARKER COLOUR INDEX</td>
</tr>
<tr>
<td>GSfMI</td>
<td>SET POLYMARKER INDEX</td>
</tr>
<tr>
<td>GSfMR</td>
<td>SET POLYMARKER REPRESENTATION</td>
</tr>
<tr>
<td>GSfGP</td>
<td>SET SEGMENT PRIORITY</td>
</tr>
<tr>
<td>GSfGT</td>
<td>SET SEGMENT TRANSFORMATION</td>
</tr>
<tr>
<td>GSSTM</td>
<td>SET STRING MODE</td>
</tr>
<tr>
<td>GSfKM</td>
<td>SET STROKE MODE</td>
</tr>
<tr>
<td>GSfXAL</td>
<td>SET TEXT ALIGNMENT</td>
</tr>
<tr>
<td>GSfXC1</td>
<td>SET TEXT COLOUR INDEX</td>
</tr>
<tr>
<td>GSfXFP</td>
<td>SET TEXT FONT AND PRECISION</td>
</tr>
<tr>
<td>GSfXI</td>
<td>SET TEXT INDEX</td>
</tr>
<tr>
<td>GSfXP</td>
<td>SET TEXT PATH</td>
</tr>
<tr>
<td>GSfXR</td>
<td>SET TEXT REPRESENTATION</td>
</tr>
<tr>
<td>GSfLM</td>
<td>SET VALUATOR MODE</td>
</tr>
<tr>
<td>GSVP</td>
<td>SET VIEWPORT</td>
</tr>
<tr>
<td>GSfVIP</td>
<td>SET VIEWPORT INPUT PRIORITY</td>
</tr>
<tr>
<td>GSfVIS</td>
<td>SET VISIBILITY</td>
</tr>
<tr>
<td>GSWN</td>
<td>SET WINDOW</td>
</tr>
<tr>
<td>GSWKVP</td>
<td>SET WORKSTATION VIEWPORT</td>
</tr>
<tr>
<td>GSWKWN</td>
<td>SET WORKSTATION WINDOW</td>
</tr>
<tr>
<td>GTX</td>
<td>TEXT</td>
</tr>
<tr>
<td>GTXS</td>
<td>TEXT (FORTRAN 77 SUBSET)</td>
</tr>
<tr>
<td>GUNEC</td>
<td>UNPACK DATA RECORD</td>
</tr>
<tr>
<td>GUWk</td>
<td>UPDATE WORKSTATION</td>
</tr>
<tr>
<td>GWITM</td>
<td>WRITE ITEM TO GKSM</td>
</tr>
</tbody>
</table>

### 7.8 List Ordered by Levels

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GACWK</td>
<td>ACTIVATE WORKSTATION</td>
</tr>
<tr>
<td>GCLKS</td>
<td>CLOSE GKS</td>
</tr>
<tr>
<td>GCLRWK</td>
<td>CLEAR WORKSTATION</td>
</tr>
<tr>
<td>GCLWK</td>
<td>CLOSE WORKSTATION</td>
</tr>
<tr>
<td>GDAWK</td>
<td>DEACTIVATE WORKSTATION</td>
</tr>
<tr>
<td>GESC</td>
<td>ESCAPE</td>
</tr>
<tr>
<td>GfA</td>
<td>FILL AREA</td>
</tr>
<tr>
<td>GGDp</td>
<td>GENERALIZED DRAWING PRIMITIVE</td>
</tr>
<tr>
<td>GOPKS</td>
<td>OPEN GKS</td>
</tr>
<tr>
<td>GOPWK</td>
<td>OPEN WORKSTATION</td>
</tr>
</tbody>
</table>

---

Page 23
List Ordered by Levels

GPL PL ma POLYLINE
GPM PM ma POLYMARKER
GCQCF Q-C-F ma INQUIRE COLOUR FACILITIES
GCQCHH Q-CH-H ma INQUIRE CHARACTER HEIGHT (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
GCQCHUP Q-CH-UP ma INQUIRE CHARACTER UP VECTOR (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
GCQLIP Q-CLIP ma INQUIRE CLIPPING INDICATOR
GCQNTN Q-C-N-T-N ma INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER
GQCRI Q-C-R ma INQUIRE COLOUR REPRESENTATION
GQEIC Q-E-I ma INQUIRE LIST element OF COLOUR INDICES
GQEGDP Q-E-G-D-P ma INQUIRE LIST element OF AVAILABLE GENERALIZED DRAWING PRIMITIVES
GQFAF Q-F-A-F ma INQUIRE FILL AREA FACILITIES
GQLN Q-LN ma INQUIRE LINETYPE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
GQLVKS Q-LV-KS ma INQUIRE LEVEL OF GKS
GQMK Q-MK ma INQUIRE MARKERTYPE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
GQNT Q-N-T ma INQUIRE NORMALIZATION TRANSFORMATION
GQPLCI Q-PL-C-I ma INQUIRE POLYLINE COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
GQPLF Q-PL-F ma INQUIRE POLYLINE FACILITIES
GQPMF Q-PM-F ma INQUIRE POLYMARKER FACILITIES
GQTXAL Q-TX-AL ma INQUIRE TEXT ALIGNMENT (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
GQTXCI Q-TX-C-I ma INQUIRE TEXT COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
GQTXF Q-TX-F ma INQUIRE TEXT FACILITIES
GQTXX Q-TX-X ma INQUIRE TEXT EXTENT
GQTXXS Q-TX-X-S ma INQUIRE TEXT EXTENT (FORTRAN 77 SUBSET)
GQWKC Q-WK-C ma INQUIRE WORKSTATION CONNECTION AND TYPE
GQWKT Q-WK-T ma INQUIRE WORKSTATION TRANSFORMATION
GSCCH S-CH-H ma SET CHARACTER HEIGHT
GSCHP S-CH-UP ma SET CHARACTER UP VECTOR
GSCILP S-CLIP ma SET CLIPPING INDICATOR
GSCR S-C-R ma SET COLOUR REPRESENTATION
GSELNT SEL-N-T ma SELECT NORMALIZATION TRANSFORMATION
GSFACI S-F-A-C-I ma SET FILL AREA COLOUR INDEX
GSFAIS S-F-A-I-S ma SET FILL AREA INTERIOR STYLE
GSLN S-LN ma SET LINETYPE
GSMK S-MK ma SET MARKERTYPE
GSPLCI S-PL-C-I ma SET POLYLINE COLOUR INDEX
GSFCICI S-FM-C-I ma SET POLYMARKER COLOUR INDEX
GSTXAL S-TX-AL ma SET TEXT ALIGNMENT
GSTXCI S-TX-C-I ma SET TEXT COLOUR INDEX
GSVP S-VP ma SET VIEWPORT
GSWKVP S-WK-VP ma SET WORKSTATION VIEWPORT
GSWKWN S-WK-WN ma SET WORKSTATION WINDOW
GSWN S-WN ma SET WINDOW
GTX TX ma TEXT
GTXS TX-S ma TEXT (FORTRAN 77 SUBSET)
GUWK U-WK ma UPDATE WORKSTATION
GINCH IN-CH mb INITIALISE CHOICE
GINLC IN-LC mb INITIALISE LOCATOR
GINSK IN-SK mb INITIALISE STROKE
GINST IN-ST mb INITIALISE STRING
GINVL IN-VL mb INITIALISE VALUATOR
GQCNS Q-CH-S mb INQUIRE CHOICE DEVICE STATE
GQDCH Q-D-CH mb INQUIRE DEFAULT CHOICE DEVICE DATA
GQDLC Q-D-LC mb INQUIRE DEFAULT LOCATOR DEVICE DATA
GQDSK Q-D-SK mb INQUIRE DEFAULT STROKE DEVICE DATA
GQDST Q-D-ST mb INQUIRE DEFAULT STRING DEVICE DATA
GQDV L Q-D-VL mb INQUIRE DEFAULT VALUATOR DEVICE DATA
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQFAI</td>
<td>INQUIRE FILL AREA COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQFAIS</td>
<td>INQUIRE FILL AREA INTERIOR STYLE (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQLCS</td>
<td>INQUIRE LOCATOR DEVICE STATE</td>
</tr>
<tr>
<td>GQLI</td>
<td>INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES</td>
</tr>
<tr>
<td>GQPMCI</td>
<td>INQUIRE POLYMARKER COLOUR INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQSKS</td>
<td>INQUIRE FILL AREA INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQSTS</td>
<td>INQUIRE STRING DEVICE STATE</td>
</tr>
<tr>
<td>GQLS</td>
<td>INQUIRE VALUATOR DEVICE STATE</td>
</tr>
<tr>
<td>GRCCH</td>
<td>REQUEST CHOICE</td>
</tr>
<tr>
<td>GRLQC</td>
<td>REQUEST LOCATOR</td>
</tr>
<tr>
<td>GRQSK</td>
<td>REQUEST STROKE</td>
</tr>
<tr>
<td>GRQST</td>
<td>REQUEST STRING</td>
</tr>
<tr>
<td>GRLVL</td>
<td>REQUEST VALUATOR</td>
</tr>
<tr>
<td>GSCHM</td>
<td>SET CHOICE MODE</td>
</tr>
<tr>
<td>GSLCM</td>
<td>SET LOCATOR MODE</td>
</tr>
<tr>
<td>GSSKM</td>
<td>SET STRING MODE</td>
</tr>
<tr>
<td>GSSTM</td>
<td>SET STRING MODE</td>
</tr>
<tr>
<td>GSVLM</td>
<td>SET VALUATOR MODE</td>
</tr>
<tr>
<td>GSVLPM</td>
<td>SET VIEWPORT INPUT PRIORITY</td>
</tr>
<tr>
<td>GFLUSH</td>
<td>FLUSH DEVICE EVENTS</td>
</tr>
<tr>
<td>GTCH</td>
<td>GET CHOICE</td>
</tr>
<tr>
<td>GRTL C</td>
<td>GET LOCATOR</td>
</tr>
<tr>
<td>GTSK</td>
<td>GET STROKE</td>
</tr>
<tr>
<td>GTST</td>
<td>GET STRING</td>
</tr>
<tr>
<td>GTVL</td>
<td>GET VALUATOR</td>
</tr>
<tr>
<td>GQIQOV</td>
<td>INQUIRE INPUT QUEUE OVERFLOW</td>
</tr>
<tr>
<td>GQSIM</td>
<td>INQUIRE MORE SIMULTANEOUS EVENTS</td>
</tr>
<tr>
<td>GSMCH</td>
<td>SAMPLE CHOICE</td>
</tr>
<tr>
<td>GSMCL</td>
<td>SAMPLE LOCATOR</td>
</tr>
<tr>
<td>GMSK</td>
<td>SAMPLE STRING</td>
</tr>
<tr>
<td>GSMST</td>
<td>SAMPLE STRING</td>
</tr>
<tr>
<td>GSVL</td>
<td>SAMPLE VALUATOR</td>
</tr>
<tr>
<td>GWAIT</td>
<td>AWAIT EVENT</td>
</tr>
<tr>
<td>GCA</td>
<td>CELL ARRAY</td>
</tr>
<tr>
<td>GECLEKS</td>
<td>EMERGENCY CLOSE GKS</td>
</tr>
<tr>
<td>GERHND</td>
<td>ERROR HANDLING</td>
</tr>
<tr>
<td>GERLOG</td>
<td>ERROR LOGGING</td>
</tr>
<tr>
<td>GTITM</td>
<td>GET ITEM TYPE FROM GKS</td>
</tr>
<tr>
<td>GHTM</td>
<td>INTERPRET ITEM</td>
</tr>
<tr>
<td>GFPREC</td>
<td>PACK DATA RECORD</td>
</tr>
<tr>
<td>GQASF</td>
<td>INQUIRE ASPECT SOURCE FLAGS (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQCHB</td>
<td>INQUIRE CHARACTER BASE VECTOR (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQCHSP</td>
<td>INQUIRE CHARACTER SPACING (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQCHW</td>
<td>INQUIRE CHARACTER WIDTH (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQCHXP</td>
<td>INQUIRE CHARACTER EXPANSION FACTOR (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GDSP</td>
<td>INQUIRE DISPLAY SPACE SIZE</td>
</tr>
<tr>
<td>GQENTN</td>
<td>INQUIRE LIST element of NORMALIZATION TRANSFORMATION NUMBERS</td>
</tr>
<tr>
<td>GQEWK</td>
<td>INQUIRE LIST element of AVAILABLE WORKSTATION TYPES</td>
</tr>
<tr>
<td>GQFAI</td>
<td>INQUIRE FILL AREA INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQFAIS</td>
<td>INQUIRE FILL AREA STYLE INDEX (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQGDPI</td>
<td>INQUIRE GENERALIZED DRAWING PRIMITIVE</td>
</tr>
<tr>
<td>GQLWK</td>
<td>INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES</td>
</tr>
</tbody>
</table>
List Ordered by Levels

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GQWSC</td>
<td>INQUIRE LINEWIDTH SCALE FACTOR (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQMKSC</td>
<td>INQUIRE MARKER SIZE SCALE FACTOR (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQMNTN</td>
<td>INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER</td>
</tr>
<tr>
<td>GQOPS</td>
<td>INQUIRE OPERATING STATE VALUE</td>
</tr>
<tr>
<td>GQOPWK</td>
<td>INQUIRE SET member OF OPEN WORKSTATIONS</td>
</tr>
<tr>
<td>GQPA</td>
<td>INQUIRE PATTERN SIZE (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQPAF</td>
<td>INQUIRE PATTERN FACILITIES</td>
</tr>
<tr>
<td>GQPCR</td>
<td>INQUIRE PATTERN REFERENCE POINT (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQPFA</td>
<td>INQUIRE PATTERN REPRESENTATION</td>
</tr>
<tr>
<td>GQPXM</td>
<td>INQUIRE POLYLINE INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQPMD</td>
<td>INQUIRE POLYMARKER INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQFPA</td>
<td>INQUIRE PREREDEFINED PATTERN REPRESENTATION</td>
</tr>
<tr>
<td>GQFPL</td>
<td>INQUIRE PREREDEFINED POLYLINE REPRESENTATION</td>
</tr>
<tr>
<td>GQFMR</td>
<td>INQUIRE PREREDEFINED POLYMARKER REPRESENTATION</td>
</tr>
<tr>
<td>GQPTX</td>
<td>INQUIRE PREDEFINED TEXT REPRESENTATION</td>
</tr>
<tr>
<td>GQFPR</td>
<td>INQUIRE PREDEFINED FILL AREA REPRESENTATION</td>
</tr>
<tr>
<td>GQPLR</td>
<td>INQUIRE PREDEFINED FILL AREA REPRESENTATION</td>
</tr>
<tr>
<td>GQX</td>
<td>INQUIRE PIXEL</td>
</tr>
<tr>
<td>GQXAI</td>
<td>INQUIRE PIXEL ARRAY</td>
</tr>
<tr>
<td>GQXAD</td>
<td>INQUIRE PIXEL ARRAY DIMENSIONS</td>
</tr>
<tr>
<td>GQTXF</td>
<td>INQUIRE TEXT FONT AND PRECISION (INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQTXI</td>
<td>INQUIRE TEXT INDEX (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQTXP</td>
<td>INQUIRE TEXT PATH (INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)</td>
</tr>
<tr>
<td>GQWKCA</td>
<td>INQUIRE WORKSTATION CATEGORY</td>
</tr>
<tr>
<td>GQWKCL</td>
<td>INQUIRE WORKSTATION CLASSIFICATION</td>
</tr>
<tr>
<td>GQWKDU</td>
<td>INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES</td>
</tr>
<tr>
<td>GQWKS</td>
<td>INQUIRE WORKSTATION STATE</td>
</tr>
<tr>
<td>GRDITM</td>
<td>READ ITEM FROM GKS</td>
</tr>
<tr>
<td>GSASF</td>
<td>SET ASPECT SOURCE FLAGS</td>
</tr>
<tr>
<td>GSCSP</td>
<td>SET CHARACTER SPACING</td>
</tr>
<tr>
<td>GSCXP</td>
<td>SET CHARACTER EXPANSION FACTOR</td>
</tr>
<tr>
<td>GSFAI</td>
<td>SET FILL AREA INDEX</td>
</tr>
<tr>
<td>GSFAS</td>
<td>SET FILL AREA STYLE INDEX</td>
</tr>
<tr>
<td>GSLWSC</td>
<td>SET LINEWIDTH SCALE FACTOR</td>
</tr>
<tr>
<td>GSMKSC</td>
<td>SET MARKER SIZE SCALE FACTOR</td>
</tr>
<tr>
<td>GSPA</td>
<td>SET PATTERN SIZE</td>
</tr>
<tr>
<td>GSPAR</td>
<td>SET PATTERN REFERENCE POINT</td>
</tr>
<tr>
<td>GSPFL</td>
<td>SET POLYLINE INDEX</td>
</tr>
<tr>
<td>GSPMI</td>
<td>SET POLYMARKER INDEX</td>
</tr>
<tr>
<td>GSTXF</td>
<td>SET TEXT FONT AND PRECISION</td>
</tr>
<tr>
<td>GSTXI</td>
<td>SET TEXT INDEX</td>
</tr>
<tr>
<td>GSTXP</td>
<td>SET TEXT PATH</td>
</tr>
<tr>
<td>GUREC</td>
<td>UNPACK DATA RECORD</td>
</tr>
<tr>
<td>GWITM</td>
<td>WRITE ITEM TO GKS</td>
</tr>
<tr>
<td>GACTM</td>
<td>ACCUMULATE TRANSFORMATION MATRIX</td>
</tr>
<tr>
<td>GCLSG</td>
<td>CLOSE SEGMENT</td>
</tr>
<tr>
<td>GCRSG</td>
<td>CREATE SEGMENT</td>
</tr>
<tr>
<td>GDSG</td>
<td>DELETE SEGMENT</td>
</tr>
<tr>
<td>GDSGWK</td>
<td>DELETE SEGMENT FROM WORKSTATION</td>
</tr>
<tr>
<td>GEVTM</td>
<td>EVALUATE TRANSFORMATION MATRIX</td>
</tr>
<tr>
<td>GMSG</td>
<td>MESSAGE</td>
</tr>
<tr>
<td>GMSGS</td>
<td>MESSAGE (FORTRAN 77 SUBSET)</td>
</tr>
<tr>
<td>GQAACWK</td>
<td>INQUIRE SET member OF ACTIVE WORKSTATIONS</td>
</tr>
<tr>
<td>GQAASK</td>
<td>INQUIRE SET member OF ASSOCIATED WORKSTATIONS</td>
</tr>
<tr>
<td>GQDDS</td>
<td>INQUIRE DEFAULT DEFERRAL STATE VALUES</td>
</tr>
<tr>
<td>GQDASG</td>
<td>INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GQDWKA</td>
<td>INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES</td>
</tr>
<tr>
<td>GEFA1</td>
<td>INQUIRE LIST element OF FILL AREA INDICES</td>
</tr>
<tr>
<td>GEPFA1</td>
<td>INQUIRE LIST element OF PATTERN INDICES</td>
</tr>
<tr>
<td>GEPFLI</td>
<td>INQUIRE LIST element OF POLYLINE INDICES</td>
</tr>
<tr>
<td>GEFMPI</td>
<td>INQUIRE LIST element OF POLYMARKER INDICES</td>
</tr>
<tr>
<td>GQETXI</td>
<td>INQUIRE LIST element OF TEXT INDICES</td>
</tr>
<tr>
<td>GQFAR</td>
<td>INQUIRE FILL AREA REPRESENTATION</td>
</tr>
<tr>
<td>GQPSG</td>
<td>INQUIRE NAME OF OPEN SEGMENT</td>
</tr>
<tr>
<td>GQPAR</td>
<td>INQUIRE PATTERN REPRESENTATION</td>
</tr>
<tr>
<td>GQPLR</td>
<td>INQUIRE POLYLINE REPRESENTATION</td>
</tr>
<tr>
<td>GQFMR</td>
<td>INQUIRE POLYMARKER REPRESENTATION</td>
</tr>
<tr>
<td>GQSGA</td>
<td>INQUIRE SEGMENT ATTRIBUTES</td>
</tr>
<tr>
<td>GQSGP</td>
<td>INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED</td>
</tr>
<tr>
<td>GQSGUS</td>
<td>INQUIRE SET member OF SEGMENT NAMES IN USE</td>
</tr>
<tr>
<td>GQSGWK</td>
<td>INQUIRE SET member OF SEGMENT NAMES ON WORKSTATION</td>
</tr>
<tr>
<td>GQTXR</td>
<td>INQUIRE TEXT REPRESENTATION</td>
</tr>
<tr>
<td>GRENSG</td>
<td>REDRAW ALL SEGMENTS ON WORKSTATION</td>
</tr>
<tr>
<td>GRSWK</td>
<td>RENAME SEGMENT</td>
</tr>
<tr>
<td>GSFAR</td>
<td>SET DEFERRAL STATE</td>
</tr>
<tr>
<td>GSHLIT</td>
<td>SET HIGHLIGHTING</td>
</tr>
<tr>
<td>GSFAR</td>
<td>SET PATTERN REPRESENTATION</td>
</tr>
<tr>
<td>GSFMR</td>
<td>SET POLYLINE REPRESENTATION</td>
</tr>
<tr>
<td>GSGP</td>
<td>SET POLYMARKER REPRESENTATION</td>
</tr>
<tr>
<td>GSSGP</td>
<td>SET SEGMENT PRIORITY</td>
</tr>
<tr>
<td>GSSGT</td>
<td>SET SEGMENT TRANSFORMATION</td>
</tr>
<tr>
<td>GSTXR</td>
<td>SET TEXT REPRESENTATION</td>
</tr>
<tr>
<td>GSVIS</td>
<td>SET VISIBILITY</td>
</tr>
<tr>
<td>GINFK</td>
<td>INITIALISE PICK</td>
</tr>
<tr>
<td>GQDPK</td>
<td>INQUIRE DEFAULT PICK DEVICE DATA</td>
</tr>
<tr>
<td>GQFKID</td>
<td>INQUIRE PICK IDENTIFIER</td>
</tr>
<tr>
<td>GQFKS</td>
<td>INQUIRE PICK DEVICE STATE</td>
</tr>
<tr>
<td>GQFKM</td>
<td>REQUEST PICK</td>
</tr>
<tr>
<td>GQFKM</td>
<td>SET DETECTABILITY</td>
</tr>
<tr>
<td>GQFKID</td>
<td>SET PICK IDENTIFIER</td>
</tr>
<tr>
<td>GSFKM</td>
<td>SET PICK MODE</td>
</tr>
<tr>
<td>GTFK</td>
<td>GET PICK</td>
</tr>
<tr>
<td>GSFKM</td>
<td>SAMPLE PICK</td>
</tr>
<tr>
<td>GASPWK</td>
<td>ASSOCIATE SEGMENT WITH WORKSTATION</td>
</tr>
<tr>
<td>GCSPWK</td>
<td>COPIY SEGMENT TO WORKSTATION</td>
</tr>
<tr>
<td>GINS SG</td>
<td>INSERT SEGMENT</td>
</tr>
</tbody>
</table>
8. GKS Errors Specific to the FORTRAN Binding

Certain features of the FORTRAN language make additional errors (beyond the ones described in the GKS document) possible. Specifically, these new errors are defined:

2000  Enumeration type out of range -- the INTEGER passed as a GKS enumerated type is not within the range of valid values.

2001  Output parameter size insufficient -- a FORTRAN array or string being passed as an output parameter is too small to contain the returned information.

2002  List element or set member not available -- for a non-empty list or set, a value less than zero or greater than the size of a list or set was passed as the requested list element or set member in an inquiry routine.

2003  Invalid data record -- the data record cannot be decoded, or there was a problem encountered when GKS was creating a data record, making the result invalid.
9. The GKS Function Interface

9.1 General Principles

For each GKS function the corresponding FORTRAN SUBROUTINE declaration is given. The name of the GKS function is listed, followed by its FORTRAN name and the corresponding parameters. After that, the list of parameters is described by type and a brief identifying phrase.

For the mapping of ENUMERATION types see above.

For GENERALIZED DRAWING PRIMITIVE and ESCAPE, subroutines GGDP and GESC are defined. Each GDP identifier that is in the ISO International Register of Graphical Items may, in addition, be accessed by a subroutine of the form GDpqrs, where pqrs is related to the appropriate GDP identifier. The parameters are derived from those of GGDP as follows: N, PXA, PYA are required; PRIMID is absent; the data record is specified by parameters appropriate to the particular GDP identifier. Similarly, each specific ESCAPE function that is in the ISO International Register of Graphical Items may, in addition, be accessed by a subroutine of the form GEpqrs. The parameters are derived from those of GESC as follows: FCTID is absent and the data record is specified by parameters appropriate to the specific ESCAPE function. For both GDP and ESCAPE, it is possible to define each separate subroutine using GGDP and GESC.
Control Functions

9.2 Control Functions

OPEN GKS

SUBROUTINE GOPKS (ERRFIL,BUFA)

Input Parameters:
INTEGER ERRFIL
INTEGER BUFA

error message file
amount of memory units (implementation
dependent; if -1, use implementation dependent default)

CLOSE GKS

SUBROUTINE GCLKS

OPEN WORKSTATION

SUBROUTINE GOPWK (WKID,CONID,WTYPE)

Input Parameters:
INTEGER WKID
INTEGER CONID
INTEGER WTYPE

workstation identifier
connection identifier
workstation type

CLOSE WORKSTATION

SUBROUTINE GCLWK (WKID)

Input Parameters:
INTEGER WKID

workstation identifier

ACTIVATE WORKSTATION

SUBROUTINE GACWK (WKID)

Input Parameters:
INTEGER WKID

workstation identifier

DEACTIVATE WORKSTATION

SUBROUTINE GDAWK (WKID)

Input Parameters:
INTEGER WKID

workstation identifier
CONTROL FUNCTIONS

CLEAR WORKSTATION

SUBROUTINE GCLRWK (WKID, COFL)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER COFL  control flag (GCOND1, GALWAY)

REDRAW ALL SEGMENTS ON WORKSTATION

SUBROUTINE GRSGWK (WKID)

Input Parameters:
INTEGER WKID  workstation identifier

UPDATE WORKSTATION

SUBROUTINE GUWK (WKID, REGFL)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER REGFL  update regeneration flag (GPOSTP, GPERFO)

SET DEFERRAL STATE

SUBROUTINE GSDS (WKID, DEFMOD, REGMOD)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER DEFMOD  deferral mode (GASAP, GBNIG, GBNIL, GASTI)
INTEGER REGMOD  implicit regeneration mode (GSUPPD, GALLOW)

MESSAGE

Full FORTRAN 77 version

SUBROUTINE GMSG (WKID, MESS)

Input Parameters:
INTEGER WKID  workstation identifier
CHARACTER*(*) MESS  message
MESSAGE
FORTRAN 77 Subset version

SUBROUTINE GMSGS (WKID, LSTR, MESS)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER LSTR  length of string (in characters)
CHARACTER*80 MESS  message

ESCAPE

SUBROUTINE GESC (FCTID, LIDR, IDR, MLODR, LODR, ODR)

Input Parameters:
INTEGER FCTID  function identification
INTEGER LIDR  dimension of input data record array
CHARACTER*80 IDR (LIDR)  input data record
INTEGER MLODR  maximum length of output data record

Output Parameters:
INTEGER LODR  number of array elements occupied in ODR
CHARACTER*80 ODR(MLODR)  output data record
9.3 Output Functions

POLYLINE

SUBROUTINE GPL (N,PXA, PYA)

Input Parameters:
INTEGER N
REAL PXA (N), PYA (N)

number of points
coordinates of points in world coordinates

POLYMARKER

SUBROUTINE GPM (N,PXA, PYA)

Input Parameters:
INTEGER N
REAL PXA (N), PYA (N)

number of points
coordinates of points in world coordinates

TEXT

Full FORTRAN 77 version

SUBROUTINE GTX (PX, PY, CHAR)

Input Parameters:
REAL PX, PY
CHARACTER*(*) CHAR

text position in world coordinates
string of characters

TEXT

FORTRAN 77 Subset version

SUBROUTINE GTXS (PX, PY, LSTR, CHAR)

Input Parameters:
REAL PX, PY
INTEGER LSTR
CHARACTER*80 CHAR

text position in world coordinates
length of string (in characters)
string of characters

FILL AREA

SUBROUTINE GFA (N,PXA, PYA)

Input Parameters:
INTEGER N
REAL PXA (N), PYA (N)

number of points
coordinates of points in world coordinates
Output Functions

CELL ARRAY

SUBROUTINE GCA (PX, PY, QX, QY, DIMX, DIMY, ISC, ISR, DX, DY, COLIA)

Input Parameters:
REAL PX, PY, QX, QY
INTEGER DIMX, DIMY
two points (P, Q) in world coordinates
the dimensions of COLIA which contains
the cell array
INTEGER ISC, ISR
indices of start column, start row
INTEGER DX, DY
number of columns, number of rows
INTEGER COLIA (DIMX,DIMY)
colour index array

GENERALIZED DRAWING PRIMITIVE

SUBROUTINE GGDP (N, PXA, PYA, PRIMID, LDR, DATREC)

Input Parameters:
INTEGER N
number of points (≥0)
REAL PXA (*), PYA (*)
coordinates of points in world coordinates
INTEGER PRIMID
GDP identifier
INTEGER LDR
dimension of data record array
CHARACTER*80 DATREC(LDR)
data record
9.4 Output Attributes

9.4.1 Workstation Independent Primitive Attributes

SET POLYLINE INDEX

SUBROUTINE GSPLI (PLI)

Input Parameters:
INTEGER PLI  polyline index

SET LINETYPE

SUBROUTINE GSLN (LTYPE)

Input Parameters:
INTEGER LTYPE  linetype

SET LINEWIDTH SCALE FACTOR

SUBROUTINE GSLWSC (LWIDTH)

Input Parameters:
REAL LWIDTH  linewidth scale factor

SET POLYLINE COLOUR INDEX

SUBROUTINE GSPLCI (COLI)

Input Parameters:
INTEGER COLI  polyline colour index

SET POLYMARKER INDEX

SUBROUTINE GSPMI (PMI)

Input Parameters:
INTEGER PMI  polymarker index

SET MARKER TYPE

SUBROUTINE GSMK (MTYPE)

Input Parameters:
INTEGER MTYPE  marker type
Output Attributes

**SET MARKER SIZE SCALE FACTOR**

SUBROUTINE GSMKSC (MSZSF)

Input Parameters:
REAL MSZSF

marker size scale factor

**SET POLYMARKER COLOUR INDEX**

SUBROUTINE GSPMCI (COLI)

Input Parameters:
INTEGER COLI

polymarker colour index

**SET TEXT INDEX**

SUBROUTINE GSTXI (TXI)

Input Parameters:
INTEGER TXI

text index

**SET TEXT FONT AND PRECISION**

SUBROUTINE GSTXFP (FONT,PREC)

Input Parameters:
INTEGER FONT
INTEGER PREC
text font
text precision (GSTRP,GCHARP,GSTRKP)

**SET CHARACTER EXPANSION FACTOR**

SUBROUTINE GSCHXP (CHXP)

Input Parameters:
REAL CHXP

character expansion factor

**SET CHARACTER SPACING**

SUBROUTINE GSCHSP (CHSP)

Input Parameters:
REAL CHSP

character spacing
SET TEXT COLOUR INDEX

SUBROUTINE GSTXCI (COLI)

Input Parameters:
INTEGER COLI  text colour index

SET CHARACTER HEIGHT

SUBROUTINE GSCHH (CHH)

Input Parameters:
REAL CHH  character height

SET CHARACTER UP VECTOR

SUBROUTINE GSCHUP (CHUX,CHUY)

Input Parameters:
REAL CHUX, CHUY  character up vector (WC)

SET TEXT PATH

SUBROUTINE GSTXP (TXP)

Input Parameters:
INTEGER TXP  text path (GRIGHT,GLEFT,GUP,GDOWN)

SET TEXT ALIGNMENT

SUBROUTINE GSTXAL (TXALH,TXALV)

Input Parameters:
INTEGER TXALH  text alignment horizontal
INTEGER TXALV  text alignment vertical
(GAHNOR,GALEFT,GACENT,GARITE)
(GAVNOR,GA TOP,GACAP,GAHALF,GABASE,GABOTT)

SET FILL AREA INDEX

SUBROUTINE GSFAI (FAI)

Input Parameters:
INTEGER FAI  fill area index
Output Attributes

**SET FILL AREA INTERIOR STYLE**

SUBROUTINE GSFAIS (INTS)

Input Parameters:
INTEGER INTS

fill area interior style
(GHOLLO, GSOLID, GPATTR, GHATCH)

**SET FILL AREA STYLE INDEX**

SUBROUTINE GSFASI (STYLI)

Input Parameters:
INTEGER STYLI

fill area style index

**SET FILL AREA COLOUR INDEX**

SUBROUTINE GSFACI (COLI)

Input Parameters:
INTEGER COLI

fill area colour index

**SET PATTERN SIZE**

SUBROUTINE GSPA (SZX, SZY)

Input Parameters:
REAL SZX, SZY

pattern size

**SET PATTERN REFERENCE POINT**

SUBROUTINE GSPARF (RFX, RFY)

Input Parameters:
REAL RFX, RFY

pattern reference point
SET ASPECT SOURCE FLAGS

SUBROUTINE GSASF (LASF)

Input Parameters:
INTEGER LASF (13)

list of aspect source flags
(GBUNDL,GINDIV)
1 linetype ASF
2 linewidth scale factor ASF
3 polyline colour index ASF
4 marker type ASF
5 marker size scale factor ASF
6 polymarker colour index ASF
7 text font and precision ASF
8 character expansion factor ASF
9 character spacing ASF
10 text colour index ASF
11 fill area interior style ASF
12 fill area style index ASF
13 fill area colour index ASF

SET PICK IDENTIFIER

SUBROUTINE GSPKID (PKID)

Input Parameters:
INTEGER PKID

pick identifier

9.4.2 Workstation Attributes (Representations)

SET POLYLINE REPRESENTATION

SUBROUTINE GSPLR (WKID,PLI,LTYPE,LWIDTH,COLI)

Input Parameters:
INTEGER WKID
INTEGER PLI
INTEGER LTYPE
REAL LWIDTH
INTEGER COLI

workstation identifier
polyline index
linetype
linewidth scale factor
colour index

SET POLYMARKER REPRESENTATION

SUBROUTINE GSPMR (WKID,PMI,MTYPE,MSZSF,COLI)

Input Parameters:
INTEGER WKID
INTEGER PMI
INTEGER MTYPE
REAL MSZSF
INTEGER COLI

workstation identifier
polymarker index
marker type
marker size scale factor
colour index
Output Attributes

SET TEXT REPRESENTATION

SUBROUTINE GSTXR (WKID, TXI, FONT, PREC, CHXP, CHSP, COLI)

Input Parameters:
INTEGER WKID  
INTEGER TXI  
INTEGER FONT  
INTEGER PREC  
REAL CHXP  
REAL CHSP  
INTEGER COLI  

workstation identifier  
text index  
text font  
text precision (GSTRP, GCHARP, GSTRKP)  
character expansion factor  
character spacing  
colour index

SET FILL AREA REPRESENTATION

SUBROUTINE GSFAR (WKID, FAI, INTS, STYLI, COLI)

Input Parameters:
INTEGER WKID  
INTEGER FAI  
INTEGER INTS  
INTEGER STYLI  
INTEGER COLI  

workstation identifier  
fill area index  
interior style  
(style index)

SET PATTERN REPRESENTATION

SUBROUTINE GSPAR (WKID, PAI, DIMX, DIMY, ISC, ISR, DX, DY, COLIA)

Input Parameters:
INTEGER WKID  
INTEGER PAI  
INTEGER DIMX, DIMY  
INTEGER ISC, ISR  
INTEGER DX, DY  
INTEGER COLIA (DIMX, DIMY)  

workstation identifier  
pattern index  
the dimensions of COLIA which contains the pattern array  
indices to start column, start row  
number of columns, number of rows used pattern array

SET COLOUR REPRESENTATION

SUBROUTINE GSCR (WKID, CI, CR, CG, CB)

Input Parameters:
INTEGER WKID  
INTEGER CI  
REAL CR, CG, CB  

workstation identifier  
colour index  
colour intensities (red/green/blue)
9.5 Transformation Functions

9.5.1 Normalization Transformation

SET WINDOW

SUBROUTINE GSWN (TNR,XMIN,XMAX,YMIN,YMAX)

Input Parameters:
INTEGER TNR   transformation number
REAL XMIN,XMAX,YMIN,YMAX window limits in world coordinates

SET VIEWPORT

SUBROUTINE GSVP (TNR,XMIN,XMAX,YMIN,YMAX)

Input Parameters:
INTEGER TNR   transformation number
REAL XMIN,XMAX,YMIN,YMAX viewport limits in normalized device coordinates

SET VIEWPORT INPUT PRIORITY

SUBROUTINE GSVPIP (TNR,RTNR,RELPRI)

Input Parameters:
INTEGER TNR   transformation number
INTEGER RTNR reference transformation number
INTEGER RELPRI relative priority (GHIGHR,GLOWER)

SELECT NORMALIZATION TRANSFORMATION

SUBROUTINE GSELNT (TNR)

Input Parameters:
INTEGER TNR   transformation number

SET CLIPPING INDICATOR

SUBROUTINE GSCLIP (CLSW)

Input Parameters:
INTEGER CLSW clipping indicator (GNCLIP,GCLIP)
9.5.2 Workstation Transformation

**SET WORKSTATION WINDOW**

SUBROUTINE GSWKWN (WKID, XMIN, XMAX, YMIN, YMAX)

Input Parameters:
- INTEGER WKID  
- REAL XMIN, XMAX, YMIN, YMAX  
  workstation window limits in normalized device coordinates

**SET WORKSTATION VIEWPORT**

SUBROUTINE GSWKVP (WKID, XMIN, XMAX, YMIN, YMAX)

Input Parameters:
- INTEGER WKID  
- REAL XMIN, XMAX, YMIN, YMAX  
  workstation viewport limits in device coordinates
9.6 Segment Functions

9.6.1 Segment Manipulation Functions

CREATE SEGMENT

SUBROUTINE GCRSG (SGNA)

Input Parameters:
INTEGER SGNA  \( \text{segment name} \)

CLOSE SEGMENT

SUBROUTINE GCLSG

RENAME SEGMENT

SUBROUTINE GRENSG (OLD,NEW)

Input Parameters:
INTEGER OLD  \( \text{old segment name} \)
INTEGER NEW  \( \text{new segment name} \)

DELETE SEGMENT

SUBROUTINE GDSG (SGNA)

Input Parameters:
INTEGER SGNA  \( \text{segment name} \)

DELETE SEGMENT FROM WORKSTATION

SUBROUTINE GDSGWK (WKID,SGNA)

Input Parameters:
INTEGER WKID  \( \text{workstation identifier} \)
INTEGER SGNA  \( \text{segment name} \)

ASSOCIATE SEGMENT WITH WORKSTATION

SUBROUTINE GASGWK (WKID,SGNA)

Input Parameters:
INTEGER WKID  \( \text{workstation identifier} \)
INTEGER SGNA  \( \text{segment name} \)
Segment Functions

COPY SEGMENT TO WORKSTATION

SUBROUTINE GCSEWK (WKID,SGNA)

Input Parameters:
INTEGER WKID
INTEGER SGNA

workstation identifier
segment name

INSERT SEGMENT

SUBROUTINE GINS (SGNA,M)

Input Parameters:
INTEGER SGNA
REAL M(2,3)

segment name
transformation matrix
(M(1,1) M(1,2) M(1,3))
(M(2,1) M(2,2) M(2,3))

9.6.2 Segment Attributes

SET SEGMENT TRANSFORMATION

SUBROUTINE GSSGT (SGNA,M)

Input Parameters:
INTEGER SGNA
REAL M(2,3)

segment name
transformation matrix
(M(1,1) M(1,2) M(1,3))
(M(2,1) M(2,2) M(2,3))

SET VISIBILITY

SUBROUTINE GSVIS (SGNA, VIS)

Input Parameters:
INTEGER SGNA
INTEGER VIS

segment name
visibility (GINVIS, GVISI)

SET HIGHLIGHTING

SUBROUTINE GSHLIT (SGNA, HIL)

Input Parameters:
INTEGER SGNA
INTEGER HIL

segment name
highlighting (GNORML, GHILIT)
SET SEGMENT PRIORITY

SUBROUTINE GSSGP (SGNA,PRIOR)

Input Parameters:
INTEGER SGNA
REAL PRIOR

segment name
segment priority

SET DETECTABILITY

SUBROUTINE GSDTEC (SGNA,DET)

Input Parameters:
INTEGER SGNA
INTEGER DET

segment name
detectability (GUNDET,GDETEC)
SUBROUTINE GINLC (WKID, LCDNR, TNR, IPX, IPY, PET, XMIN, XMAX, YMIN, YMAX, LDR, DATREC)

Input Parameters:
INTEGER WKID
INTEGER LCDNR
INTEGER TNR
REAL IPX, IPY
INTEGER PET
REAL XMIN, XMAX, YMIN, YMAX
INTEGER LDR
CHARACTER*80 DATREC(LDR)

workstation identifier
locator device number
initial normalization transformation number
initial locator position (WC)
prompt/echo type
echo area in device coordinates
dimension of data record array
data record

GPREC parameters to build LOCATOR INPUT DATA RECORD for PET = 4:
IL=7 (or 1 if attribute control flag=GCURNT)
IA=(attribute control flag[, linetype ASF, linewidth scale])
RL=1 (or 0 if attribute control flag=GCURNT)
RA=([linewidth scale factor])
SL=0
LSTR=()
STR=()

GPREC parameters to build LOCATOR INPUT DATA RECORD for PET = 5 and polyline/fill area flag=GPLINE:
IL=8 (or 2 if attribute control flag=GCURNT)
IA=(polyline/fill area control flag, attribute control flag)
RL=1 (or 0 if attribute control flag=GCURNT)
RA=(linewidth scale factor)
SL=0
LSTR=()
STR=()

GPREC parameters to build LOCATOR INPUT DATA RECORD for PET = 5 and polyline/fill area flag=GFILLA:
IL=8 (or 2 if attribute control flag=GCURNT)
IA=(polyline/fill area control flag, attribute control flag)
RL=0
RA=()
SL=0
LSTR=()
STR=()
SUBROUTINE GINSK (WKID, SKDNR, TNR, N, IPX, IPY, PET, XMIN, XMAX, *YMIN, YMAX, BUFLEN, LDR, DATREC)

Input Parameters:
INTEGER WKID
INTEGER SKDNR
INTEGER TNR
INTEGER N
REAL IPX (*), IPY (*)

INTEGER PET
REAL XMIN, XMAX, YMIN, YMAX
INTEGER BUFLEN
INTEGER LDR
CHARACTER*80 DATREC(LDR)

workstation identifier
stroke device number
initial normalization transformation number
number of points in initial stroke
points in initial stroke (WC) (the actual arguments are dimensioned by at least MAX(1,N))
prompt/echo type
echo area in device coordinates
input buffer size
dimension of data record array
data record

GPREC parameters to build STROKE INPUT DATA RECORD for PET = 1 or 2:
IL = 0
IA = ()
RL = 3
RA = (x interval, y interval, time interval in seconds)
SL = 0
LSTR = ()
STR = ()

GPREC parameters to build STROKE INPUT DATA RECORD for PET = 3:
IL = 5 (or 1 if attribute control flag = GCURNT)
IA = (attribute control flag, marker size scale factor ASF)
RL = 4 (or 3 if attribute control flag = GCURNT)
RA = (x interval, y interval, time interval in seconds)
SL = 0
LSTR = ()
STR = ()

GPREC parameters to build STROKE INPUT DATA RECORD for PET = 4:
IL = 7 (or 1 if attribute control flag = GCURNT)
IA = (attribute control flag, linetype ASF, linewidth scale)
RL = 4 (or 3 if attribute control flag = GCURNT)
RA = (x interval, y interval, time interval in seconds)
SL = 0
LSTR = ()
STR = ()
**INITIALISE VALUATOR**

SUBROUTINE GINVIL (WKID, VLDNR, IVAL, PET, XMIN, XMAX, YMIN, YMAX, LOVAL, HIVAL, LDR, DATREC)

Input Parameters:
- INTEGER WKID: workstation identifier
- INTEGER VLDNR: valuator device number
- REAL IVAL: initial value
- INTEGER PET: prompt/echo type
- REAL XMIN, XMAX, YMIN, YMAX: echo area in device coordinates
- REAL LOVAL, HIVAL: minimal and maximal value
- INTEGER LDR: dimension of data record array
- CHARACTER*80 DATREC(LDR): data record
INITIALISE CHOICE

SUBROUTINE GINCH (WKID, CHDNR, ISTAT, ICHNR, PET, XMIN, XMAX, YMIN, YMAX, LDR, DATREC)

Input Parameters:
INTEGER WKID
INTEGER CHDNR
INTEGER ISTAT
INTEGER ICHNR
INTEGER PET
REAL XMIN, XMAX, YMIN, YMAX
INTEGER LDR
CHARACTER*80 DATREC(LDR)

workstation identifier
choice device number
initial status (GOK, GNCHOI)
initial choice number
prompt/echo type
echo area in device coordinates
dimension of data record array
data record

GPREC parameters to build CHOICE INPUT DATA RECORD for PET = 2:
IL = number of choice alternatives
IA = (array of prompts)
RL = 0
RA = ()
SL = 0
LSTR = ()
STR = ()

GPREC parameters to build CHOICE INPUT DATA RECORD for PET = 3 or 4:
IL = 0
IA = ()
RL = 0
RA = ()
SL = number of choice strings
LSTR = (array of lengths of choice strings)
STR = (array of strings)

GPREC parameters to build CHOICE INPUT DATA RECORD for PET = 5:
IL = 1
IA = (segment name)
RL = 0
RA = ()
SL = 0
LSTR = ()
STR = ()
INITIALISE PICK

SUBROUTINE GINPK (WKID, PKDNR, ISTAT, ISGNA, IPKID, PET, XMIN, XMAX, YMIN, YMAX, LDR, DATREC)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER PKDNR  pick device number
INTEGER ISTAT  initial status (GOK, GNPKICK)
INTEGER ISGNA  initial segment name
INTEGER IPKID  initial pick identifier
INTEGER PET  prompt/echo type
REAL XMIN, XMAX, YMIN, YMAX  echo area in device coordinates
INTEGER LDR  dimension of data record array
CHARACTER*80 DATREC(LDR)  data record

INITIALISE STRING

Full FORTRAN 77 version

SUBROUTINE GINST (WKID, STDNR, LSTR, ISTR, PET, XMIN, XMAX, YMIN, YMAX, BUFLEN, INIPOS, LDR, DATREC)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER STDNR  string device number
INTEGER LSTR  length of the initial string (≥ 0). The number of characters actually used is the minimum of LSTR and the length of ISTR.
CHARACTER*(*) ISTR  initial string
INTEGER PET  prompt/echo type
REAL XMIN, XMAX, YMIN, YMAX  echo area in device coordinates
INTEGER BUFLEN  input buffer size
INTEGER INIPOS  initial cursor position
INTEGER LDR  dimension of data record array
CHARACTER*80 DATREC(LDR)  data record
**INITIALISE STRING**

FORTRAN 77 Subset version

SUBROUTINE GINST (WKID, STDNR, LSTR, ISTR, PET, XMIN, XMAX, YMIN, YMAX, BUFLEN, INIPOS, LDR, DATREC)

Input Parameters:
- INTEGER WKID: workstation identifier
- INTEGER STDNR: string device number
- INTEGER LSTR: length of the initial string
- CHARACTER*80 ISTR: initial string
- INTEGER PET: prompt/echo type
- REAL XMIN, XMAX, YMIN, YMAX: echo area in device coordinates
- INTEGER BUFLEN: buffer length of string
- INTEGER INIPOS: initial cursor position
- INTEGER LDR: dimension of data record array
- CHARACTER*80 DATREC(LDR): data record

9.7.2 Setting Mode of Input Devices

**SET LOCATOR MODE**

SUBROUTINE GSLCM (WKID, LCDNR, MODE, ESW)

Input Parameters:
- INTEGER WKID: workstation identifier
- INTEGER LCDNR: locator device number
- INTEGER MODE: operating mode (GREQU, GSAMPL, GEVENT)
- INTEGER ESW: echo switch (GNECHO, GECHO)

**SET STROKE MODE**

SUBROUTINE GSSKM (WKID, SKDNR, MODE, ESW)

Input Parameters:
- INTEGER WKID: workstation identifier
- INTEGER SKDNR: stroke device number
- INTEGER MODE: operating mode (GREQU, GSAMPL, GEVENT)
- INTEGER ESW: echo switch (GNECHO, GECHO)

**SET VALUATOR MODE**

SUBROUTINE GSVLM (WKID, VLDNR, MODE, ESW)

Input Parameters:
- INTEGER WKID: workstation identifier
- INTEGER VLDNR: valuator device number
- INTEGER MODE: operating mode (GREQU, GSAMPL, GEVENT)
- INTEGER ESW: echo switch (GNECHO, GECHO)
Input Functions

SET CHOICE MODE

SUBROUTINE GSCHM (WKID, CHDNR, MODE, ESW)

Input Parameters:
INTEGER WKID workstation identifier
INTEGER CHDNR choice device number
INTEGER MODE operating mode (GREQU, GSAMPL, GEVENT)
INTEGER ESW echo switch (GNECHO, GECHO)

SET PICK MODE

SUBROUTINE GSPKM (WKID, PKDNR, MODE, ESW)

Input Parameters:
INTEGER WKID workstation identifier
INTEGER PKDNR pick device number
INTEGER MODE operating mode (GREQU, GSAMPL, GEVENT)
INTEGER ESW echo switch (GNECHO, GECHO)

SET STRING MODE

SUBROUTINE GSSTM (WKID, STD NR, MODE, ESW)

Input Parameters:
INTEGER WKID workstation identifier
INTEGER STD NR string device number
INTEGER MODE operating mode (GREQU, GSAMPL, GEVENT)
INTEGER ESW echo switch (GNECHO, GECHO)

9.7.3 Request Input Functions

REQUEST LOCATOR

SUBROUTINE GRQLC (WKID, LCDNR, STAT, TNR, PX, PY)

Input Parameters:
INTEGER WKID workstation identifier
INTEGER LCDNR locator device number

Output Parameters:
INTEGER STAT status (GNONE, GOK)
INTEGER TNR normalization transformation number
REAL PX, PY locator position
REQUEST STROKE

SUBROUTINE GRQSK (WKID, SKDNR, N, STAT, TNR, NP, PXA, PYA)

Input Parameters:
INTEGER WKID
INTEGER SKDNR
INTEGER N

Output Parameters:
INTEGER STAT
INTEGER TNR
INTEGER NP
REAL PXA (N), PYA (N)

Input Parameters:
INTEGER WKID
INTEGER SKDNR
INTEGER N

Output Parameters:
INTEGER STAT
INTEGER TNR
INTEGER NP
REAL PXA (N), PYA (N)

REQUEST VALUATOR

SUBROUTINE GRQVL (WKID, VLDNR, STAT, VAL)

Input Parameters:
INTEGER WKID
INTEGER VLDNR

Output Parameters:
INTEGER STAT
REAL VAL

REQUEST CHOICE

SUBROUTINE GRQCH (WKID, CHDNR, STAT, CHNR)

Input Parameters:
INTEGER WKID
INTEGER CHDNR

Output Parameters:
INTEGER STAT
INTEGER CHNR

REQUEST PICK

SUBROUTINE GRQPK (WKID, PKDNR, STAT, SGNA, PKID)

Input Parameters:
INTEGER WKID
INTEGER PKDNR

Output Parameters:
INTEGER STAT
INTEGER SGNA
INTEGER PKID
INPUT FUNCTIONS

REQUEST STRING
Full FORTRAN 77 version

SUBROUTINE GRQST (WKID, STD NR, STAT, LOSTR, STR)

Input Parameters:
INTEGER WKID
INTEGER STD NR

workstation identifier
string device number

Output Parameters:
INTEGER STAT
INTEGER LOSTR
CHARACTER(*) STR

status (GNONE, GOK)
number of characters returned
character string

REQUEST STRING
FORTRAN 77 Subset version

SUBROUTINE GRQST (WKID, STD NR, STAT, LOSTR, STR)

Input Parameters:
INTEGER WKID
INTEGER STD NR

workstation identifier
string device number

Output Parameters:
INTEGER STAT
INTEGER LOSTR
CHARACTER*80 STR

status (GNONE, GOK)
number of characters returned
character string

9.7.4 Sample Input Functions

SAMPLE LOCATOR

SUBROUTINE GSMLC (WKID, LCD NR, TNR, LPX, LPY)

Input Parameters:
INTEGER WKID
INTEGER LCD NR

workstation identifier
locator device number

Output Parameters:
INTEGER TNR
REAL LPX, LPY

normalization transformation number
locator position in WC
SAMPLE STROKE

SUBROUTINE GSMSK (WKID,SKDNR,N,TNR,NP,PXA,PYA)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER SKDNR  stroke device number
INTEGER N  maximum number of points

Output Parameters:
INTEGER TNR  normalization transformation number
INTEGER NP  number of points
REAL PXA (N), PYA (N)  points in stroke (WC)

SAMPLE VALUATOR

SUBROUTINE GSMVL (WKID,VLDNR,VAL)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER VLDNR  valuator device number

Output Parameters:
REAL VAL  value

SAMPLE CHOICE

SUBROUTINE GSMCH (WKID,CHDNR,STAT,CHNR)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER CHDNR  choice device number

Output Parameters:
INTEGER STAT  status (GOK,GNCHOI)
INTEGER CHNR  choice number

SAMPLE PICK

SUBROUTINE GSMPK (WKID,PKDNR,STAT,SGNA,PKID)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER PKDNR  pick device number

Output Parameters:
INTEGER STAT  status (GOK,GNPICK)
INTEGER SGNA  segment name
INTEGER PKID  pick identifier
### Input Functions

#### SAMPLE STRING

**Full FORTRAN 77 version**

```fortran
SUBROUTINE GSMST (WKID,STDNR,LOSTR,STR)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER STDNR  string device number

Output Parameters:
INTEGER LOSTR  number of characters returned
CHARACTER*(*) STR  string
```

**FORTRAN 77 Subset version**

```fortran
SUBROUTINE GSMST (WKID,STDNR,LOSTR,STR)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER STDNR  string device number

Output Parameters:
INTEGER LOSTR  number of characters returned
CHARACTER*80 STR  string
```

#### 9.7.5 Event Input Functions

##### AWAIT EVENT

```fortran
SUBROUTINE GWAIT (TOUT,WKID,ICL,ID NR)

Input Parameters:
REAL TOUT  time out (seconds)

Output Parameters:
INTEGER WKID  workstation identifier
INTEGER ICL  input class (GNCLAS,GLOCAT,GSTROK, GVALUA,GCHOIC,GPICK,GSTRIN)
INTEGER ID NR  logical input device number
```

##### FLUSH DEVICE EVENTS

```fortran
SUBROUTINE GFLUSH (WKID,ICL,ID NR)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER ICL  input class (GLOCAT,GSTROK,GVALUA, GCHOIC,GPICK,GSTRIN)
INTEGER ID NR  logical input device number
```
GET LOCATOR

SUBROUTINE GGTLC (TNR,LPX,LPY)

Output Parameters:
INTEGER TNR  normalization transformation number
REAL LPX,LPY  locator position in world coordinates

GET STROKE

SUBROUTINE GGTSDK (N,TNR,NP,PXA,PYA)

Input Parameters:
INTEGER N  maximum number of points

Output Parameters:
INTEGER TNR  normalization transformation number
INTEGER NP  number of points
REAL PXA (N), PYA (N)  points in stroke in world coordinates

GET VALUATOR

SUBROUTINE GGTVAL (VAL)

Output Parameters:
REAL VAL  value

GET CHOICE

SUBROUTINE GGTCH (STAT,CHNR)

Output Parameters:
INTEGER STAT  status (GOK,GNCHOI)
INTEGER CHNR  choice number

GET PICK

SUBROUTINE GGTPK (STAT,SGNA,PKID)

Output Parameters:
INTEGER STAT  status (GOK,GNPICK)
INTEGER SGNA  segment name
INTEGER PKID  pick identifier
Input Functions

**GET STRING**
Full FORTRAN 77 version

```fortran
SUBROUTINE GGTST (LOSTR, STR)

Output Parameters:
INTEGER LOSTR  number of characters returned
CHARACTER*(*) STR  string
```

**GET STRING**
FORTRAN 77 Subset version

```fortran
SUBROUTINE GGTST (LOSTR, STR)

Output Parameters:
INTEGER LOSTR  number of characters returned
CHARACTER*80 STR  string
```
9.8 Metafile Functions

WRITE ITEM TO GKSM

SUBROUTINE GWITM (WKID, TYPE, IDRL, LDR, DATREC)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER TYPE  item type
INTEGER IDRL  item data record length (number of characters in the data record array)
INTEGER LDR  dimension of data record array
CHARACTER*80 DATREC(LDR)  data record

GET ITEM TYPE FROM GKSM

SUBROUTINE GGTITM (WKID, TYPE, IDRL)

Input Parameters:
INTEGER WKID  workstation identifier

Output Parameters:
INTEGER TYPE  item type
INTEGER IDRL  item data record length (this may be passed to GRDITM (MIDRL)

READ ITEM FROM GKSM

SUBROUTINE GRDITM (WKID, MIDRL, MLDL, DATREC)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER MIDRL  maximum item data record length (number of characters in the data record array). Range is 0...IDRL. If =0, then skip the record; if < IDRL, the excess is lost; if = IDRL, a full record is read. (IDRL is returned by GGTITM.)
INTEGER MLDL  dimension of item data record

Output Parameters:
CHARACTER*80 DATREC(MLDL)  data record
**SUBROUTINE GITM (TYPE,IDRL,LDR,DATREC)**

**Input Parameters:**
- **INTEGER TYPE**
- **INTEGER IDRL**
- **INTEGER LDR**
- **CHARACTER*80 DATREC(LDR)**

**Definitions:**
- **item type**
- **item data record length (number of characters in the data record array)**
- **dimension of data record array**
- **data record**
9.9 Inquiry Functions

9.9.1 Inquiry Function for Operating State Value

INQUIRE OPERATING STATE VALUE

SUBROUTINE GQOPS (OPSTA)

Output Parameters:
INTEGER OPSTA operating state value
(GGKCL,GGKOP,GWSOP,GWSAC,GSGOP)

9.9.2 Inquiry Functions for GKS Description Table

INQUIRE LEVEL OF GKS

SUBROUTINE GQLVKS (ERRIND,LEVEL)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER LEVEL level of GKS (LMA,LMB,LMC,
L0A,L0B,L0C,L1A,L1B,L1C,L2A,L2B,L2C)

INQUIRE LIST element OF AVAILABLE WORKSTATION TYPES

SUBROUTINE GQEWK (N,ERRIND,NUMBER,WKTyp)

Input Parameters:
INTEGER N list element requested

Output Parameters:
INTEGER ERRIND error indicator
INTEGER NUMBER number of workstation types
INTEGER WKTyp Nth element of list of av. workstation types

INQUIRE WORKSTATION MAXIMUM NUMBERS

SUBROUTINE GQWKM (ERRIND,MXOPWK,MXACWK,MXWKAS)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER MXOPWK maximum number of simultaneously open
workstations
INTEGER MXACWK maximum number of simultaneously active
workstations
INTEGER MXWKAS maximum number of workstations associated
with segment
**Inquiry Functions**

**INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER**

SUBROUTINE GQMNTN (ERRIND, MAXTNR)

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER MAXTNR: maximum normalization transformation number

**Inquiry Functions for GKS State List**

**INQUIRE SET member OF OPEN WORKSTATIONS**

SUBROUTINE GQOPWK (N, ERRIND, OL, WKID)

Input Parameters:
- INTEGER N: set member requested

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER OL: number of open workstations
- INTEGER WKID: Nth member of set of open workstations

**INQUIRE SET member OF ACTIVE WORKSTATIONS**

SUBROUTINE GQACWK (N, ERRIND, OL, WKID)

Input Parameters:
- INTEGER N: set member requested

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER OL: number of active workstations
- INTEGER WKID: Nth member of set of active workstations

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)

**INQUIRE POLYLINE INDEX**

SUBROUTINE GQPLI (ERRIND, PLI)

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER PLI: polyline index
Inquiry Functions

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE POLYMARKER INDEX

SUBROUTINE GQPMI (ERRIND,PMI)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER PMI     polymarker index

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE TEXT INDEX

SUBROUTINE GQTXI (ERRIND,TXI)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER TXI     text index

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE CHARACTER HEIGHT

SUBROUTINE GQCHH (ERRIND,CHH)

Output Parameters:
INTEGER ERRIND  error indicator
REAL CHH        character height

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE CHARACTER UP VECTOR

SUBROUTINE GQCHUP (ERRIND,CHUX,CHUY)

Output Parameters:
INTEGER ERRIND  error indicator
REAL CHUX, CHUY character up vector (WC)

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE CHARACTER WIDTH

SUBROUTINE GQCHW (ERRIND,CHW)

Output Parameters:
INTEGER ERRIND  error indicator
REAL CHW        character width
Inquiry Functions

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE CHARACTER BASE VECTOR

SUBROUTINE GQCHB (ERRIND, CHBX, CHBY)

Output Parameters:
INTEGER ERRIND  error indicator
REAL CHBX, CHBY  character base vector

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE TEXT PATH

SUBROUTINE GQTXP (ERRIND, TXP)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER TXP  text path (GRIGHT, GLEFT, GUP, GDOWN)

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE TEXT ALIGNMENT

SUBROUTINE GQTXAL (ERRIND, TXALH, TXALV)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER TXALH  text alignment horizontal
INTEGER TXALV  text alignment vertical

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE FILL AREA INDEX

SUBROUTINE GQFAI (ERRIND, FAI)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER FAI  fill area index

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE PATTERN SIZE

SUBROUTINE GQPA (ERRIND, PWX, PWY, PHX, PHY)

Output Parameters:
INTEGER ERRIND  error indicator
REAL PWX, PWY  pattern width vector
REAL PHX, PHY  pattern height vector
Inquiry Functions

(INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES)
INQUIRE PATTERN REFERENCE POINT

SUBROUTINE GQPARF (ERRIND,RFX,RFY)

Output Parameters:
INTEGER ERRIND error indicator
REAL RFX, RFY pattern reference point

(INQUIRE CURRENT PICK IDENTIFIER)

SUBROUTINE GQPKID (ERRIND,PKID)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER PKID pick identifier

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE LINETYPE

SUBROUTINE GQLN (ERRIND,LTYPE)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER LTYPE linetype

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE LINE WIDTH SCALE FACTOR

SUBROUTINE GQLWSC (ERRIND,LWIDTH)

Output Parameters:
INTEGER ERRIND error indicator
REAL LWIDTH linewidth scale factor

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE POLYLINE COLOUR INDEX

SUBROUTINE GQPLCI (ERRIND,COLI)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER COLI polyline colour index
Inquiry Functions

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)

INQUIRE MARKERTYPE

SUBROUTINE GQMK (ERRIND, MTYPE)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER MTYPE  marker type

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)

INQUIRE MARKER SIZE SCALE FACTOR

SUBROUTINE GQMKSC (ERRIND, MSZSF)

Output Parameters:
INTEGER ERRIND  error indicator
REAL MSZSF  marker size scale factor

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)

INQUIRE POLYMARKER COLOUR INDEX

SUBROUTINE GQPMCI (ERRIND, COLI)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER COLI  polymarker colour index

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)

INQUIRE TEXT FONT AND PRECISION

SUBROUTINE GQTXFP (ERRIND, FONT, PREC)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER FONT  text font
INTEGER PREC  text precision (GSTRP, GCHARP, GSTRKP)

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)

INQUIRE CHARACTER EXPANSION FACTOR

SUBROUTINE GQCHXP (ERRIND, CHXP)

Output Parameters:
INTEGER ERRIND  error indicator
REAL CHXP  character expansion factor
(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE CHARACTER SPACING

SUBROUTINE GQCHSP (ERRIND, CHSP)

Output Parameters:
INTEGER ERRIND error indicator
REAL CHSP character spacing

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE TEXT COLOUR INDEX

SUBROUTINE GQTXCI (ERRIND, COLI)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER COLI text colour index

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE FILL AREA INTERIOR STYLE

SUBROUTINE GQFAIS (ERRIND, INTS)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER INTS fill area interior style
(GHOLLO, GSOLID, GPATTR, GHATCH)

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE FILL AREA STYLE INDEX

SUBROUTINE GQFASI (ERRIND, STYLI)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER STYLI fill area style index

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE FILL AREA COLOUR INDEX

SUBROUTINE GQFACI (ERRIND, COLI)

Output Parameters:
INTEGER ERRIND error indicator
INTEGER COLI fill area colour index
Inquiry Functions

(INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES)
INQUIRE ASPECT SOURCE FLAGS

SUBROUTINE GQASF (ERRIND, LASF)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER LASF (13)  list of aspect source flags
(GBUNDL,GINDIV)
1  linetype ASF
2  linewidth scale factor ASF
3  polyline colour index ASF
4  marker type ASF
5  marker size scale factor ASF
6  polymarker colour index ASF
7  text font and precision ASF
8  character expansion factor ASF
9  character spacing ASF
10  text colour index ASF
11  fill area interior style ASF
12  fill area style index ASF
13  fill area colour index ASF

INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER

SUBROUTINE GQCNTN (ERRIND, CTNR)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER CTNR  current transformation number

INQUIRE LIST element OF NORMALIZATION TRANSFORMATION NUMBERS

SUBROUTINE GQENTN (N, ERRIND, OL, TNR)

Input Parameters:
INTEGER N  list element requested

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER OL  length of list
INTEGER TNR  Nth element of list of transformation numbers, ordered by decreasing viewport input priority
INQUIRE NORMALIZATION TRANSFORMATION

SUBROUTINE GQNT (NTNR,ERRIND,WINDOW,VIEWPT)

Input Parameters:
INTEGER NTNR  
Output Parameters:
INTEGER ERRIND
REAL WINDOW(4)
REAL VIEWPT(4)

Input Parameters:
INTEGER NTNR  
Output Parameters:
INTEGER ERRIND
REAL WINDOW(4)
REAL VIEWPT(4)

INQUIRE CLIPPING

SUBROUTINE GQCLIP (ERRIND,CLSW,CLRECT)

Output Parameters:
INTEGER ERRIND
INTEGER CLSW
REAL CLRECT(4)

Output Parameters:
INTEGER ERRIND
INTEGER CLSW
REAL CLRECT(4)

INQUIRE NAME OF OPEN SEGMENT

SUBROUTINE GQOPSG (ERRIND,SGNA)

Output Parameters:
INTEGER ERRIND
INTEGER SGNA

Output Parameters:
INTEGER ERRIND
INTEGER SGNA

INQUIRE SET member OF SEGMENT NAMES IN USE

SUBROUTINE GQSGUS (N,ERRIND,OL,SGNA)

Input Parameters:
INTEGER N
Output Parameters:
INTEGER ERRIND
INTEGER OL
INTEGER SGNA

Input Parameters:
INTEGER N
Output Parameters:
INTEGER ERRIND
INTEGER OL
INTEGER SGNA

Inquiry Functions
INQUIRY MORE SIMULTANEOUS EVENTS

SUBROUTINE GQSIM (ERRIND,FLAG)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER FLAG more simultaneous events (GNMORE,GMORE)

9.9.4 Inquiry Functions for Workstation State List

INQUIRE WORKSTATION CONNECTION AND TYPE

SUBROUTINE GQVKC (WKID,ERRIND,CONID,WTYPE)

Input Parameters:
INTEGER WKID workstation identifier

Output Parameters:
INTEGER ERRIND error indicator
INTEGER CONID connection identifier
INTEGER WTYPE workstation type

INQUIRE WORKSTATION STATE

SUBROUTINE GQWKS (WKID,ERRIND,STATE)

Input Parameters:
INTEGER WKID workstation identifier

Output Parameters:
INTEGER ERRIND error indicator
INTEGER STATE workstation state (GINACT,GACTIV)

INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES

SUBROUTINE GQWKDU (WKID,ERRIND,DEFMOD,REGMOD,DEMPH,
*NFRAME)

Input Parameters:
INTEGER WKID workstation identifier

Output Parameters:
INTEGER ERRIND error indicator
deferral mode (GASAP,GBNIG,GBNIL,GASTI)
INTEGER REGMOD implicit regeneration mode (GSUPPD,GALLOW)
INTEGER DEMPTY display surface empty (GNEMPT,GEMPTY)
INTEGER NFRAME new frame action necessary at update (GNO,GYES)
INQUIRE LIST element OF POLYLINE INDICES

SUBROUTINE GQEPLI (WKID,N,ERRIND,OL,PLI)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER N  list element requested

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER OL  number of polyline bundle table entries
INTEGER PLI  Nth element of list of defined polyline indices

INQUIRE POLYLINE REPRESENTATION

SUBROUTINE GQPLR (WKID,PLI,TYPE,ERRIND,LTYPE,LWIDTH,COLI)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER PLI  polyline index
INTEGER TYPE  type of returned values (GSET,GREALI)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER LTYPE  linetype
REAL LWIDTH  linewidth scale factor
INTEGER COLI  polyline colour index

INQUIRE LIST element OF POLYMARKER INDICES

SUBROUTINE GQEPMI (WKID,N,ERRIND,OL,PMI)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER N  list element requested

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER OL  number of polymarker bundle table entries
INTEGER PMI  Nth element of list of defined polymarker indices
INQUIRY FUNCTIONS

INQUIRE POLYMARKER REPRESENTATION

SUBROUTINE GQPMR (WKID, PMI, TYPE, ERRIND, MTYPE, MSZSF, COLI)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER PMI  polymarker index
INTEGER TYPE  type of returned values (GSET, GREALI)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER MTYPE  marker type
REAL MSZSF  marker size scale factor
INTEGER COLI  polymarker colour index

INQUIRE LIST element OF TEXT INDICES

SUBROUTINE GQETXI (WKID, N, ERRIND, OL, TXI)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER N  list element requested

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER OL  number of text bundle table entries
INTEGER TXI  Nth element of list of defined text indices

INQUIRE TEXT REPRESENTATION

SUBROUTINE GQTXR (WKID, TXI, TYPE, ERRIND, FONT, PREC, CHXP, CHSP, COLI)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER TXI  text index
INTEGER TYPE  type of returned values (GSET, GREALI)

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER FONT  text font
INTEGER PREC  text precision (GSTRP, GCHARP, GSTRKP)
REAL CHXP  character expansion factor
REAL CHSP  character spacing
INTEGER COLI  text colour index
INQUIRE TEXT EXTENT
Full FORTRAN 77 version

SUBROUTINE GQTXX (WKID,PX,PY,STR,ERRIND,CPX,CPY,
+TXEXPX,TXEXPY)

Input Parameters:
INTEGER WKID
REAL PX,PY
CHARACTER*(*) STR

Output Parameters:
INTEGER ERRIND
REAL CPX,CPY
REAL TXEXPX (4), TXEXPY (4)

workstation identifier
character string
concatenation point in world coordinates
text extent rectangle

INQUIRE TEXT EXTENT
FORTRAN 77 Subset version

SUBROUTINE GQTXXS (WKID,PX,PY,LSTR,STR,ERRIND,CPX,CPY,
+TXEXPX,TXEXPY)

Input Parameters:
INTEGER WKID
REAL PX,PY
INTEGER LSTR
CHARACTER*80 STR

Output Parameters:
INTEGER ERRIND
REAL CPX,CPY
REAL TXEXPX, TXEXPY (4)

error indicator
concatenation point in world coordinates
text extent rectangle

INQUIRE LIST element OF FILL AREA INDICES

SUBROUTINE GQEFAI (WKID,N,ERRIND,OL,FAI)

Input Parameters:
INTEGER WKID
INTEGER N

Output Parameters:
INTEGER ERRIND
INTEGER OL
INTEGER FAI

error indicator
number of fill area bundle table entries
Nth element of list of defined fill area indices
INQUIRY FUNCTIONS

INQUIRE FILL AREA REPRESENTATION

SUBROUTINE GQFAR (WKID, FAI, TYPE, ERRIND, INTS, STYLI, COLI)

Input Parameters:
INTEGER WKID                         workstation identifier
INTEGER FAI                          fill area index
INTEGER TYPE                         type of returned values (GSET, GREALI)

Output Parameters:
INTEGER ERRIND                       error indicator
INTEGER INTS                         fill area interior style
(INHOLLO, GSOLID, GPATTR, GHATCH)
INTEGER STYLI                         fill area style index
INTEGER COLI                         fill area colour index

INQUIRE LIST element OF PATTERN INDICES

SUBROUTINE GQEPAI (WKID, N, ERRIND, OL, PAI)

Input Parameters:
INTEGER WKID                         workstation identifier
INTEGER N                            list element requested

Output Parameters:
INTEGER ERRIND                       error indicator
INTEGER OL                           number of pattern table entries
INTEGER PAI                          Nth element of list of pattern indices

INQUIRE PATTERN REPRESENTATION

SUBROUTINE GQPAR (WKID, PAI, TYPE, DIMX, DIMY, ERRIND, DX, DY, COLIA)

Input Parameters:
INTEGER WKID                         workstation identifier
INTEGER PAI                          pattern index
INTEGER TYPE                         type of returned values (GSET, GREALI)
INTEGER DIMX, DIMY                   maximum pattern array dimensions

Output Parameters:
INTEGER ERRIND                       error indicator
INTEGER DX, DY                       pattern array dimensions
INTEGER COLIA (DIMX, DIMY)           pattern array
INQUIRE LIST element OF COLOUR INDICES

SUBROUTINE GQECI (WKID, N, ERRIND, OL, COLI)

Input Parameters:
INTEGER WKID
INTEGER N

Output Parameters:
INTEGER ERRIND
INTEGER OL
INTEGER COLI

INQUIRE COLOUR REPRESENTATION

SUBROUTINE GQCR (WKID, COLI, TYPE, ERRIND, CR, CG, CB)

Input Parameters:
INTEGER WKID
INTEGER COLI
INTEGER TYPE

Output Parameters:
INTEGER ERRIND
REAL CR, CG, CB

INQUIRE WORKSTATION TRANSFORMATION

SUBROUTINE GQWKT (WKID, ERRIND, TUS, RWINDO, CWINDO, RVIEWP, CVIEWP)

Input Parameters:
INTEGER WKID

Output Parameters:
INTEGER ERRIND
INTEGER TUS
REAL RWINDO(4)
REAL CWINDO(4)
REAL RVIEWP(4)
REAL CVIEWP(4)
INQUIRY FUNCTIONS

INQUIRE SET member OF SEGMENT NAMES ON WORKSTATION

SUBROUTINE GQSGWK (WKID, N, ERRIND, OL, SGNA)

Input Parameters:
INTEGER WKID
INTEGER N

Output Parameters:
INTEGER ERRIND
INTEGER OL
INTEGER SGNA

Workstation identifier
set member requested

error indicator
number of segment names
Nth set member of set of stored segments for workstation

INQUIRE LOCATOR DEVICE STATE

SUBROUTINE GQLCS (WKID, LCD NR, TYPE, MLD R, ERRIND, MODE, ESW, TNR, *IPX, IPY, PET, EAREA, LDR, DATREC)

Input Parameters:
INTEGER WKID
INTEGER LCD NR
INTEGER TYPE
INTEGER MLD R

Output Parameters:
INTEGER ERRIND
INTEGER MODE
INTEGER ESW
INTEGER TNR
REAL IPX, IPY
INTEGER PET
REAL EAREA(4)
INTEGER LDR
CHARACTER*80 DATREC(MLD R)

Error indicator
operating mode (GREQU, GREALI)
echo switch (GNECHO, GECHO)
initial normalization transformation number
initial locator position
in world coordinates
prompt/echo type
echo area in device coordinates
XMIN, XMAX, YMIN, YMAX
number of array elements used in data record
data record
INQUIRE STROKE DEVICE STATE

SUBROUTINE GQS KS (WKID, SKDNR, TYPE, N, MLDR, ERRIND, MODE, ESW, ITNR, *NP, PXA, PYA, PET, EAREA, BUFLEN, LDR, DATREC)

Input Parameters:
INTEGER WKID
INTEGER SKDNR
INTEGER TYPE
INTEGER N
INTEGER MLDR

Output Parameters:
INTEGER ERRIND
INTEGER MODE
INTEGER ESW
INTEGER ITNR
INTEGER NP
REAL PXA(N), PYA(N)
INTEGER PET
REAL EAREA(4)
INTEGER BUFLEN
INTEGER LDR
CHARACTER*80 DATREC(MLDR)

workstation identifier
stroke device number
type of returned values (GSET, GREAL1)
maximum number of points
dimension of data record array

error indicator
operating mode (GREQU, GSAMPL, GEVENT)
echo switch (GNECHO, GECHO)
initial normalization transformation number
number of points
initial points in stroke in world coordinates
prompt/echo type
echo area in device coordinates
XMIN, XMAX, YMIN, YMAX
input buffer size
number of array elements used in data record
data record

INQUIRE VALUATOR DEVICE STATE

SUBROUTINE GQV LS (WKID, VLDNR, MLDR, ERRIND, MODE, ESW, IVAL, PET, +EAREA, LOVAL, HIVAL, LDR, DATREC)

Input Parameters:
INTEGER WKID
INTEGER VLDNR
INTEGER MLDR

Output Parameters:
INTEGER ERRIND
INTEGER MODE
INTEGER ESW
REAL IVAL
INTEGER PET
REAL EAREA(4)
REAL LOVAL, HIVAL
INTEGER LDR
CHARACTER*80 DATREC(MLDR)

workstation identifier
valuator device number
dimension of data record array

error indicator
operating mode (GREQU, GSAMPL, GEVENT)
echo switch (GNECHO, GECHO)
initial value
prompt/echo type
echo area in device coordinates
XMIN, XMAX, YMIN, YMAX
minimal and maximal value
number of array elements used in data record
data record
Inquiry Functions

**INQUIRE CHOICE DEVICE STATE**

SUBROUTINE GQCHS (WKID, CHDNR, MLDR, ERRIND, MODE, ESW, ISTAT, *ICHNR, PET, EAREA, LDR, DATREC)

Input Parameters:
- INTEGER WKID: workstation identifier
- INTEGER CHDNR: choice device number
- INTEGER MLDR: dimension of data record array

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER MODE: operating mode (GREQU, GSAMPL, GEVENT)
- INTEGER ESW: echo switch (GNECHO, GECHO)
- INTEGER ISTAT: initial status (GOK, GNCHOI)
- INTEGER ICHNR: initial choice number
- INTEGER PET: prompt/echo type
- REAL EAREA(4): echo area in device coordinates
- INTEGER LDR: number of array elements used in data record
- CHARACTER*80 DATREC(MLDR): data record

**INQUIRE PICK DEVICE STATE**

SUBROUTINE GQPKS (WKID, PKDNR, TYPE, MLDR, ERRIND, MODE, ESW, ISTAT, *ISGNA, IPKID, PET, EAREA, LDR, DATREC)

Input Parameters:
- INTEGER WKID: workstation identifier
- INTEGER PKDNR: pick device number
- INTEGER TYPE: type of returned values (GSET, GREALI)
- INTEGER MLDR: dimension of data record array

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER MODE: operating mode (GREQU, GSAMPL, GEVENT)
- INTEGER ESW: echo switch (GNECHO, GECHO)
- INTEGER ISTAT: initial status (GOK, GNPK)
- INTEGER ISGNA: initial segment
- INTEGER IPKID: initial pick identifier
- INTEGER PET: prompt/echo type
- REAL EAREA(4): echo area in device coordinates
- INTEGER LDR: number of array elements used in data record
- CHARACTER*80 DATREC(MLDR): data record
INQUIRE STRING DEVICE STATE
Full FORTRAN 77 version

SUBROUTINE GQSTS (WKID, STD NR, MLDR, ERRIND, MODE, ESW, LOSTR, ISTR, PET, EAREA, BUFLEN, INIPOS, LDR, DATREC)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER STD NR  string device number
INTEGER MLDR  dimension of data record array

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER MODE  operating mode (GREQU, GSAMPL, GEVENT)
INTEGER ESW  echo switch (GNECHO, GECHO)
INTEGER LOSTR  number of characters returned
CHARACTER*80 ISTR  initial string
INTEGER PET  prompt/echo type
REAL EAREA(4)  echo area in device coordinates
INTEGER BUFLEN  input buffer size
INTEGER INIPOS  initial cursor position
INTEGER LDR  number of array elements used in data record
CHARACTER*80 DATREC(MLDR)  data record

INQUIRE STRING DEVICE STATE
FORTRAN 77 Subset version

SUBROUTINE GQSTS (WKID, STD NR, MLDR, ERRIND, MODE, ESW, LOSTR, ISTR, PET, EAREA, BUFLEN, INIPOS, LDR, DATREC)

Input Parameters:
INTEGER WKID  workstation identifier
INTEGER STD NR  string device number
INTEGER MLDR  dimension of data record array

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER MODE  operating mode (GREQU, GSAMPL, GEVENT)
INTEGER ESW  echo switch (GNECHO, GECHO)
INTEGER LOSTR  number of characters returned
CHARACTER*80 ISTR  initial string
INTEGER PET  prompt/echo type
REAL EAREA(4)  echo area in device coordinates
INTEGER BUFLEN  input buffer size
INTEGER INIPOS  initial cursor position
INTEGER LDR  number of array elements used in data record
CHARACTER*80 DATREC(MLDR)  data record
Inquiry Functions

9.9.5 Inquiry Functions for Workstation Description Table

INQUIRE WORKSTATION CATEGORY

SUBROUTINE GQWKCA (WTYPE, ERRIND, WKCAT)

Input Parameters:
INTEGER WTYPE    workstation type

Output Parameters:
INTEGER ERRIND   error indicator
INTEGER WKCAT    workstation category (GOUTPT, GINPUT, GOUTIN, GWISS, GMO, GMI)

INQUIRE WORKSTATION CLASSIFICATION

SUBROUTINE GQWKCL (WTYPE, ERRIND, VRTYPE)

Input Parameters:
INTEGER WTYPE    workstation type

Output Parameters:
INTEGER ERRIND   error indicator
INTEGER VRTYPE   vector/raster/other type (GVECTR, GRASTR, GOTHWK)

INQUIRE DISPLAY SPACE SIZE

SUBROUTINE GQDSP (WTYPE, ERRIND, DCUNIT, RX, RY, LX, LY)

Input Parameters:
INTEGER WTYPE    workstation type

Output Parameters:
INTEGER ERRIND   error indicator
INTEGER DCUNIT   device coordinate units (GMETRE, GOTHU)
REAL RX, RY      maximum display surface size (DC)
INTEGER LX, LY   maximum display surface size (raster units)
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES

SUBROUTINE GQDWKA (WTYPE, ERRIND, PLBUN, PMBUN, TXBUN, FABUN, PAREP, COLREP, WKTR)

Input Parameters:
INTEGER WTYPE  
   workstation type

Output Parameters:
INTEGER ERRIND  
   error indicator
INTEGER PLBUN  
   polyline representation changeable (GIRG,GIMM)
INTEGER PMBUN  
   polymarker representation changeable (GIRG,GIMM)
INTEGER TXBUN  
   text representation changeable (GIRG,GIMM)
INTEGER FABUN  
   fill area representation changeable (GIRG,GIMM)
INTEGER PAREP  
   pattern representation changeable (GIRG,GIMM)
INTEGER COLREP  
   colour representation changeable (GIRG,GIMM)
INTEGER WKTR  
   workstation transformation changeable (GIRG,GIMM)

INQUIRE DEFAULT DEFERRAL STATE VALUES

SUBROUTINE GQDDS (WTYPE, ERRIND, DEFMOD, REGMOD)

Input Parameters:
INTEGER WTYPE  
   workstation type

Output Parameters:
INTEGER ERRIND  
   error indicator
INTEGER DEFMOD  
   default value for deferral mode (GASAP,GBNIG,GBNIL,GASTI)
INTEGER REGMOD  
   default value for implicit regeneration mode (GSUPPD,GALLOW)
### INQUIRE POLYLINE FACILITIES

SUBROUTINE GQPLF (WTYPE, N, ERRIND, NLT, LT, NLW, NOMLW, RLWMIN, RLWMAX, NPPLI)

**Input Parameters:**
- INTEGER WTYPE: workstation type
- INTEGER N: list element requested

**Output Parameters:**
- INTEGER ERRIND: error indicator
- INTEGER NLT: number of available linetypes
- INTEGER LT: Nth element of list of available linetypes
- INTEGER NLW: number of available linewidths
- REAL NOMLW: nominal linewidth
- REAL RLWMIN, RLWMAX: range of linewidths
- INTEGER NPPLI: number of predefined polyline indices

### INQUIRE PREDEFINED POLYLINE REPRESENTATION

SUBROUTINE GQPPLR (WTYPE, PLI, ERRIND, LTYPE, LWIDTH, COLI)

**Input Parameters:**
- INTEGER WTYPE: workstation type
- INTEGER PLI: predefined polyline index

**Output Parameters:**
- INTEGER ERRIND: error indicator
- INTEGER LTYPE: linetype
- REAL LWIDTH: linewidth scale factor
- INTEGER COLI: polyline colour index

### INQUIRE POLYMARKER FACILITIES

SUBROUTINE GQPMF (WTYPE, N, ERRIND, NMT, MT, NMS, NOMMS, RMSMIN, RMSMAX, NPPMI)

**Input Parameters:**
- INTEGER WTYPE: workstation type
- INTEGER N: list element requested

**Output Parameters:**
- INTEGER ERRIND: error indicator
- INTEGER NMT: number of available marker types
- INTEGER MT: Nth element of list of available marker types
- INTEGER NMS: number of available marker sizes
- REAL NOMMS: nominal marker size
- REAL RMSMIN, RMSMAX: range of marker sizes
- INTEGER NPPMI: number of predefined polymarker indices
INQUIRE PREDEFINED POLYMARKER REPRESENTATION

SUBROUTINE GQPPMR (WTYPE, PMI, ERRIND, MTYPE, MSZSF, COLI)

Input Parameters:
INTEGER WTYPE         workstation type
INTEGER PMI           predefined polymarker index

Output Parameters:
INTEGER ERRIND        error indicator
INTEGER MTYPE         marker type
REAL MSZSF            marker size scale factor
INTEGER COLI          polymarker colour index

INQUIRE TEXT FACILITIES

SUBROUTINE GQTXF (WTYPE, N, ERRIND, NFPP, FONT, PREC, NCHH, MINCHH, MAXCHH, NCHX, MINCHX, MAXCHX, NPTXI)

Input Parameters:
INTEGER WTYPE         workstation type
INTEGER N             list element requested

Output Parameters:
INTEGER ERRIND        error indicator
INTEGER NFPP          number of text font and precision pairs
INTEGER FONT          Nth element of list of text fonts
INTEGER PREC          Nth element of list of text precisions
(INTEGER NCHH, REAL MINCHH, REAL MAXCHH, INTEGER NCHX, REAL MINCHX, REAL MAXCHX, INTEGER NPTXI)

number of available character heights
minimum character height (DC)
maximum character height (DC)
number of available character
expansion factors
minimum character expansion factor
maximum character expansion factor
number of predefined text indices
Inquiry Functions

INQUIRE PREDEFINED TEXT REPRESENTATION

SUBROUTINE GQPTXR (WTYPE,PTXI,ERRIND,FONT,PREC,CHXP,CHSP,*COLI)

Input Parameters:
INTEGER WTYPE  workstation type
INTEGER PTXI  predefined text index

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER FONT  text font
INTEGER PREC  text precision (GSTRP,GCHARP,GSTRKP)
REAL CHXP  character expansion factor
REAL CHSP  character spacing
INTEGER COLI  text colour index

INQUIRE FILL AREA FACILITIES

SUBROUTINE GQFAF (WTYPE,NI,NH,ERRIND,NIS,IS,NHS,HS,NPFAI)

Input Parameters:
INTEGER WTYPE  workstation type
INTEGER NI  list element of interior styles requested
INTEGER NH  list element of hatch styles requested

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER NIS  number of available fill area interior styles
INTEGER IS  N1th element of list of available fill area interior styles
(GHOLLO,G SOLID,GPATTR,G HATCH)
INTEGER NHS  number of available fill area hatch styles
INTEGER HS  NHth element of list of available fill area hatch style indices
INTEGER NPFAI  number of predefined fill area indices

INQUIRE PREDEFINED FILL AREA REPRESENTATION

SUBROUTINE GQPFAR (WTYPE,PFAI,ERRIND,INTS,STYLI,COLI)

Input Parameters:
INTEGER WTYPE  workstation type
INTEGER PFAI  predefined fill area index

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER INTS  fill area interior style
(GHOLLO,G SOLID,GPATTR,G HATCH)
INTEGER STYLI  fill area style index
INTEGER COLI  fill area colour index
INQUIRE PATTERN FACILITIES

SUBROUTINE GQPAF (WTYPE, ERRIND, NPPAI)

Input Parameters:
INTEGER WTYPE  
workstation type

Output Parameters:
INTEGER ERRIND  
error indicator
INTEGER NPPAI  
number of predefined pattern indices

INQUIRE PREDEFINED PATTERN REPRESENTATION

SUBROUTINE GQPPAR (WTYPE, PPAI, DIMX, DIMY, ERRIND, DX, DY, COLIA)

Input Parameters:
INTEGER WTYPE  
workstation type
INTEGER PPAI  
predefined pattern index
INTEGER DIMX, DIMY  
maximum pattern array dimensions

Output Parameters:
INTEGER ERRIND  
error indicator
INTEGER DX, DY  
pattern array dimensions
INTEGER COLIA (DIMX, DIMY)  
pattern array

INQUIRE COLOUR FACILITIES

SUBROUTINE GQCF (WTYPE, ERRIND, NCOLI, COLA, NPCI)

Input Parameters:
INTEGER WTYPE  
workstation type

Output Parameters:
INTEGER ERRIND  
error indicator
INTEGER NCOLI  
number of colours
INTEGER COLA  
colour available (GMONOC, GCOLOR)
INTEGER NPCI  
number of predefined colour indices

INQUIRE PREDEFINED COLOUR REPRESENTATION

SUBROUTINE GQPCR (WTYPE, PCI, ERRIND, CR, CG, CB)

Input Parameters:
INTEGER WTYPE  
workstation type
INTEGER PCI  
predefined colour index

Output Parameters:
INTEGER ERRIND  
error indicator
REAL CR, CG, CB  
colour intensities (red/green/blue)
INQUIRY FUNCTIONS

INQUIRE LIST element OF AVAILABLE GENERALIZED DRAWING PRIMITIVES:

SUBROUTINE GQEGDP (WTYPE,N,ERRIND,NGDP,GDPL)

Input Parameters:
INTEGER WTYPE  
workstation type
INTEGER N  
list element requested

Output Parameters:
INTEGER ERRIND  
error indicator
INTEGER NGDP  
number of available generalized drawing primitives
INTEGER GDPL  
Nth element of list of GDP identifiers

INQUIRE GENERALIZED DRAWING PRIMITIVE

SUBROUTINE GQGDP (WTYPE,GDP,ERRIND,NBND,BNDL)

Input Parameters:
INTEGER WTYPE  
workstation type
INTEGER GDP  
GDP identifier

Output Parameters:
INTEGER ERRIND  
error indicator
INTEGER NBND  
number of sets of attributes used
INTEGER BNDL(4)  
list of sets of attributes used
(GPLBND,GPMBND,GTXBND,GFABND)

INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES

SUBROUTINE GQLWK (WTYPE,ERRIND ,MPLBTE,MPMBTE,MTXBTE,MFABTE, 
*MPAI,MCOLI)

Input Parameters:
INTEGER WTYPE  
workstation type

Output Parameters:
INTEGER ERRIND  
error indicator
INTEGER MPLBTE  
maximum number of polyline bundle table entries
INTEGER MPMBTE  
maximum number of polymarker bundle table entries
INTEGER MTXBTE  
maximum number of text bundle table entries
INTEGER MFABTE  
maximum number of fill area bundle table entries
INTEGER MPAI  
maximum number of pattern indices
INTEGER MCOLI  
maximum number of colour indices
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED

SUBROUTINE GQSGP (WTYPE, ERRIND, NSG)

Input Parameters:
INTEGER WTYPE  workstation type

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER NSG  number of segment priorities supported

INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES

SUBROUTINE GQDSGA (WTYPE, ERRIND, SGTR, VONOFF, VOFFON, HIGH, SGPR, ADD, SGDEL)

Input Parameters:
INTEGER WTYPE  workstation type

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER SGTR  segment transformation changeable (GIRG, GIMM)
INTEGER VONOFF  visibility changeable from on to off (GIRG, GIMM)
INTEGER VOFFON  visibility changeable from off to on (GIRG, GIMM)
INTEGER HIGH  segment priority changeable (GIRG, GIMM)
INTEGER SGPR  adding primitives to the open segment (GIRG, GIMM)
INTEGER ADD  segment deletion immediately visible (GIRG, GIMM)

INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES

SUBROUTINE GQLI (WTYPE, ERRIND, NLCD, NSKD, NVLD, NCHD, NPKD, NSTD)

Input Parameters:
INTEGER WTYPE  workstation type

Output Parameters:
INTEGER ERRIND  error indicator
INTEGER NLCD  number of locator devices
INTEGER NSKD  number of stroke devices
INTEGER NVLD  number of valuator devices
INTEGER NCHD  number of choice devices
INTEGER NPKD  number of pick devices
INTEGER NSTD  number of string devices
INQUIRY DEFAULT LOCATOR DEVICE DATA

SUBROUTINE GQDLC (WTYPE, DEVNO, N, MLDR, ERRIND, DPX, DPY, *OL, PET, EAREA, LDR, DATREC)

Input Parameters:
INTEGER WTYPE
INTEGER DEVNO
INTEGER N
INTEGER MLDR

Output Parameters:
INTEGER ERRIND
REAL DPX, DPY
INTEGER OL
INTEGER PET
REAL EAREA[4]
INTEGER LDR
CHARACTER*80 DATREC(MLDR)

INQUIRY DEFAULT STROKE DEVICE DATA

SUBROUTINE GQDSK (WTYPE, DEVNO, N, MLDR, ERRIND, DBUFSK, *OL, PET, EAREA, BUFLEN, LDR, DATREC)

Input Parameters:
INTEGER WTYPE
INTEGER DEVNO
INTEGER N
INTEGER MLDR

Output Parameters:
INTEGER ERRIND
INTEGER DBUFSK
INTEGER OL
INTEGER PET
REAL EAREA[4]
INTEGER BUFLEN
INTEGER LDR
CHARACTER*80 DATREC(MLDR)
INQUIRE DEFAULT VALUATOR DEVICE DATA

SUBROUTINE GQDVL (WTYPE, DEVNO, N, MLDR, ERRIND, DVAL,
* OL, PET, EAREA, LOVAL, HIVAL, LDR, DATREC)

Input Parameters:
INTEGER WTYPE 
INTEGER DEVNO 
INTEGER N 
INTEGER MLDR 

Output Parameters:
INTEGER ERRIND 
REAL DVAL 
INTEGER OL 
INTEGER PET 
REAL EAREA(4) 
REAL LOVAL, HIVAL 
INTEGER LDR 
CHARACTER*80 DATREC(MLDR)

INQUIRE DEFAULT CHOICE DEVICE DATA

SUBROUTINE GQDCH (WTYPE, DEVNO, N, MLDR, ERRIND, MALT,
* OL, PET, EAREA, LDR, DATREC)

Input Parameters:
INTEGER WTYPE 
INTEGER DEVNO 
INTEGER N 
INTEGER MLDR 

Output Parameters:
INTEGER ERRIND 
INTEGER MALT 
INTEGER OL 
INTEGER PET 
REAL EAREA(4) 
REAL LOVAL, HIVAL 
INTEGER LDR 
CHARACTER*80 DATREC(MLDR)
**INQUIRE DEFAULT PICK DEVICE DATA**

SUBROUTINE GQDPK (WTYPE, DEVNO, N, MLDR, ERRIND, *OL, PET, EAREA, LDR, DATREC)

**Input Parameters:**
- INTEGER WTYPE: workstation type
- INTEGER DEVNO: logical input device number
- INTEGER N: list element requested
- INTEGER MLDR: dimension of data record array

**Output Parameters:**
- INTEGER ERRIND: error indicator
- INTEGER OL: number of available prompt/echo types
- INTEGER PET: Nth element of list of available prompt/echo types
- REAL EAREA(4): default echo area in device coordinates
- INTEGER LDR: number of array elements used in data record
- CHARACTER*80 DATREC(MLDR): data record

---

**INQUIRE DEFAULT STRING DEVICE DATA**

SUBROUTINE GQDST (WTYPE, DEVNO, N, MLDR, ERRIND, MBUFF, *OL, PET, EAREA, BUFLEN, LDR, DATREC)

**Input Parameters:**
- INTEGER WTYPE: workstation type
- INTEGER DEVNO: logical input device number
- INTEGER N: list element requested
- INTEGER MLDR: dimension of data record array

**Output Parameters:**
- INTEGER ERRIND: error indicator
- INTEGER MBUFF: maximum string buffer size
- INTEGER OL: number of available prompt/echo types
- INTEGER PET: Nth element of list of available prompt/echo types
- REAL EAREA(4): default echo area in device coordinates
- INTEGER BUFLEN: buffer length of string
- INTEGER LDR: number of array elements used in data record
- CHARACTER*80 DATREC(MLDR): data record
9.9.6 Inquiry Functions for Segment State List

**INQUIRE SET member OF ASSOCIATED WORKSTATIONS**

SUBROUTINE GQASWK (SGNA,N,ERRIND,OL,WKID)

Input Parameters:
INTEGER SGNA segment name
INTEGER N set member requested

Output Parameters:
INTEGER ERRIND error indicator
INTEGER OL number of associated workstations
INTEGER WKID Nth member of set of associated workstations

**INQUIRE SEGMENT ATTRIBUTES**

SUBROUTINE GQSGA (SGNA,ERRIND,SEGTM,VIS,HIGH,SGPR,DET)

Input Parameters:
INTEGER SGNA segment name

Output Parameters:
INTEGER ERRIND error indicator
REAL SEGTM(2,3) segment transformation matrix
INTEGER VIS visibility (GINVIS,GVISI)
INTEGER HIGH highlighting (GNORML,GHILIT)
REAL SGPR segment priority
INTEGER DET detectability (GUNDET,GDETEC)

9.9.7 Pixel Inquiries

**INQUIRE PIXEL ARRAY DIMENSIONS**

SUBROUTINE GQPXAD (WKID,PX,PY,QX,QY,ERRIND,N,M)

Input Parameters:
INTEGER WKID workstation identifier
REAL PX, PY, QX, QY upper left, lower right corners in world coordinates

Output Parameters:
INTEGER ERRIND error indicator
INTEGER N,M dimensions of pixel array
Inquiry Functions

**INQUIRE PIXEL ARRAY**

SUBROUTINE GQPXA (WKID, PX, PY, DIMX, DIMY, ISC, ISR, DX, DY, ERRIND, INVVAL, COLIA)

Input Parameters:
- INTEGER WKID: workstation identifier
- REAL PX, PY: upper left corner (WC)
- INTEGER DIMX, DIMY: the dimensions of colour index array
- INTEGER ISC, ISR: start column, start row
- INTEGER DX, DY: size of requested pixel array

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER INVVAL: presence of invalid values (GABSNT, GPRSNT)
- INTEGER COLIA (DIMX, DIMY): colour index array

**INQUIRE PIXEL**

SUBROUTINE GQPX (WKID, PX, PY, ERRIND, COLI)

Input Parameters:
- INTEGER WKID: workstation identifier
- REAL PX, PY: point in world coordinates

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER COLI: colour index

9.9.8 Inquiry Function for GKS Error State List

**INQUIRE INPUT QUEUE OVERFLOW**

SUBROUTINE GQIQOV (ERRIND, WKID, ICL, IDN)

Output Parameters:
- INTEGER ERRIND: error indicator
- INTEGER WKID: workstation identifier
- INTEGER ICL: input class
  - (GLOCAT, GSTROK, GVALUA, GCHOIC, GPICK, GSTRIN)
- INTEGER IDN: input device number
9.10 Utility Functions

EVALUATE TRANSFORMATION MATRIX

SUBROUTINE GEVTM (X0, Y0, DX, DY, PHI, FX, FY, SW, MOUT)

Input Parameters:
REAL X0, Y0  fixed point
REAL DX, DY  shift vector
REAL PHI  rotation angle in radians
REAL FX, FY  scale factors
INTEGER SW  coordinate switch (GWC, GNDC)

Output Parameters:
REAL MOUT(2,3)  segment transformation matrix
(MOUT(1,1) MOUT(1,2) MOUT(1,3))
(MOUT(2,1) MOUT(2,2) MOUT(2,3))

ACCUMULATE TRANSFORMATION MATRIX

SUBROUTINE GACTM (MINP, X0, Y0, DX, DY, PHI, FX, FY, SW, MOUT)

Input Parameters:
REAL MINP(2,3)  segment transformation matrix
(MINP(1,1) MINP(1,2) MINP(1,3))
(MINP(2,1) MINP(2,2) MINP(2,3))
REAL X0, Y0  fixed point
REAL DX, DY  shift vector
REAL PHI  rotation angle in radians
REAL FX, FY  scale factors
INTEGER SW  coordinate switch (GWC, GNDC)

Output Parameters:
REAL MOUT(2,3)  segment transformation matrix
(MOUT(1,1) MOUT(1,2) MOUT(1,3))
(MOUT(2,1) MOUT(2,2) MOUT(2,3))
9.11 Error Handling

EMERGENCY CLOSE GKS

SUBROUTINE GECLKS

ERROR HANDLING

SUBROUTINE GERHND (ERRNR,FCTID,ERRFIL)

Input Parameters:
INTEGER ERRNR
INTEGER FCTID
INTEGER ERRFIL

error number
function identification (see Section 6)
error file

ERROR LOGGING

SUBROUTINE GERLOG (ERRNR,FCTID,ERRFIL)

Input Parameters:
INTEGER ERRNR
INTEGER FCTID
INTEGER ERRFIL

error number
function identification (see Section 6)
error file
Utility Functions not defined in GKS

The mechanism used for providing the error indicator of these utilities is the same as that used by the inquiry functions. The following language binding dependent errors may occur: 2001 and 2003. Note: implementation dependent errors may occur also.

PACK DATA RECORD
Full FORTRAN 77 version

SUBROUTINE GPREC(IL,IA,RL,RA,SL,LSTR,STR,MLDR,ERRIND,LDR,DATREC)

Input Parameters:
INTEGER IL
INTEGER IA (*)
INTEGER RL
REAL RA (*)
INTEGER SL
INTEGER LSTR(*)
CHARACTER*(*) STR(*)
INTEGER MLDR

Output Parameters:
INTEGER ERRIND
INTEGER LDR
CHARACTER*80 DATREC(MLDR)

PACK DATA RECORD
FORTRAN 77 Subset version

SUBROUTINE GPREC (IL,IA,RL,RA,SL,LSTR,STR,MLDR,ERRIND,LDR,DATREC)

Input Parameters:
INTEGER IL
INTEGER IA (*)
INTEGER RL
REAL RA (*)
INTEGER SL
INTEGER LSTR(*)
CHARACTER*80 STR(*)
INTEGER MLDR

Output Parameters:
INTEGER ERRIND
INTEGER LDR
CHARACTER*80 DATREC(MLDR)
Utility Functions not defined in GKS

UNPACK DATA RECORD
Full FORTRAN 77 version

SUBROUTINE GUREC (LDR, DATREC, IIL, IRL, ISL, ERRIND, IL, IA, RL, RA, *SL, LSTR, STR)

Input Parameters:
INTEGER LDR
CHARACTER*80 DATREC(LDR)
INTEGER IIL
INTEGER IRL
INTEGER ISL

Output Parameters:
INTEGER ERRIND
INTEGER IL
INTEGER IA (IIL)
INTEGER RL
REAL RA (IRL)
INTEGER SL
INTEGER LSTR(ISL)
CHARACTER*80 STR(ISL)

number of array elements used in DATREC
data record
dimension of integer array
dimension of real array
dimension of character array
error indicator (zero if no error)
number of integer entries
array containing integer entries
number of real entries
array containing real entries
number of character string entries
length of each character string entry
character string entries

UNPACK DATA RECORD
FORTRAN 77 Subset version

SUBROUTINE GUREC (LDR, DATREC, IIL, IRL, ISL, ERRIND, IL, IA, RL, RA, *SL, LSTR, STR)

Input Parameters:
INTEGER LDR
CHARACTER*80 DATREC(LDR)
INTEGER IIL
INTEGER IRL
INTEGER ISL

Output Parameters:
INTEGER ERRIND
INTEGER IL
INTEGER IA (IIL)
INTEGER RL
REAL RA (IRL)
INTEGER SL
INTEGER LSTR(ISL)
CHARACTER*80 STR(ISL)

number of array elements used in DATREC
data record
dimension of integer array
dimension of real array
dimension of character array
error indicator (zero if no error)
number of integer entries
array containing integer entries
number of real entries
array containing real entries
number of character string entries
length of each character string entry
character string entries
Appendix A FORTRAN Examples

This appendix is not part of the Standard, but provides additional information.
The following sample programs, using the GKS FORTRAN binding, illustrate the use of GKS functions.
Example 1:

C PROGRAM STAR
C
C DESCRIPTION:
C This program draws a yellow star on a blue background and writes
C the title 'Star' in green in under the star.
C
C CONFORMANCE:
C GKS level: ma
C FORTRAN-77 binding with FORTRAN-77 Subset comments.
C At least one output or outin workstation.

C Define GKS constants.
C
INTEGER GSOLID
PARAMETER ( GSOLID = 1 )
INTEGER GAGENT,GAGHALF
PARAMETER ( GAGENT = 2, GAGHALF = 3 )

C Implementation dependent constants.
C
INTEGER ERROUT,TTOUT,WSTYPE,NBYTES
PARAMETER ( ERROUT = 1, TTOUT = 5, WSTYPE = 0, NBYTES = -1 )

C Define coordinates for drawing the star.
C
REAL STARX( 5 ),STARY( 5 )
DATA STARX / 0.951057, -0.951057, 0.587785, 0, -0.587785 /
DATA STARY / 0.309017, 0.309017, -0.951057, 1.0, -0.951057 /

C Perform implementation dependent initialization.
C
OPEN ( TTOUT,STATUS='NEW' )

C Open GKS and activate a workstation.
C
CALL GOPKS ( ERROUT,NBYTES )
CALL GOPWK ( 1,TTOUT,WSTYPE )
CALL GACWK ( 1 )

C Center the window around the origin.
C
CALL GSWN ( 1,-1.25,1.25,-1.25,1.25 )
CALL GSELNT ( 1 )

C Define the colors we'll be using.
C
CALL GSCR ( 1,0,0.0,0.0,1.0 )
CALL GSCR ( 1,1,1.0,1.0,0.0 )
CALL GSCR ( 1,2,1.0,1.0,1.0 )
C Fill the star with solid yellow.
C CALL GSFAIS ( GSOLID )
C CALL GSFACI ( 1 )
C
C Draw the star.
C CALL GFA ( 5,STARX,STARY )
C C Select large characters centered under the star.
C CALL GSCHH ( 0.15 )
C CALL GSTXAL ( GACENT,GAHALF )
C CALL GSTXCI ( 2 )
C
C Draw the title.
C CALL GTX ( 0.0,-1.1,'Star' )
C
C FORTRAN 77 subset version.
C CALL GTXS ( 0.0,-1.1,4,'Star' )
C
C Close the workstation and shut down GKS.
C CALL GDAWK ( 1 )
C CALL GCLWK ( 1 )
C CALL GCLKS
C STOP
C END
Example 2:

C PROGRAM IRON
C
C DESCRIPTION:
C This program draws a horizontal bar chart illustrating costs within
C the iron industry. The user can select the data to be displayed using
C a GKS choice device. The plot is adapted from 'Scientific American'
C May 1984 page 39.
C
CONFORMANCE:
C GKS level: 0b
C Choice device must support prompt and echo type 3
C
C GKS constants:
C
INTEGER GBUNDL,GINDIV
PARAMETER ( GBUNDL = 0, GINDIV = 1 )
INTEGER GOK,GNCHOI
PARAMETER ( GOK = 0, GNCHOI = 1 )

C Implementation dependent constants:
C
INTEGER ERROUT,TTOUT,WSTYPE,NBYTES
PARAMETER ( ERROUT = 5,TTOUT = 5, WSTYPE = 0, NBYTES = -1 )

C Aspect source flags:
C
INTEGER ASFLST( 13 )

C Returned data from inquire choice device state:
C
INTEGER ERRIND,MODE,ESW,ISTAT,ICHNR,PET,LDR
REAL EAREA( 4 )
CHARACTER*80 DATREC( 10 )

C Data for initializing choice device state:
C
CHARACTER*12 CHSTRS( 4 )
INTEGER CHLENS( 4 )
INTEGER IDUMMY
REAL RDUMMY
INTEGER STATUS,CHOICE

C Arrays for plot data:
C
REAL USDAT1( 6 ),USDAT2( 6 ),GDAT1( 6 ),GDAT2( 6 ),JDAT1( 6 ),
1 JDAT2( 6 )

C Initialize the prompt data:
C
DATA CHSTRS / 'U.S.', 'W. GERMANY', 'JAPAN', 'EXIT' /
DATA CHLENS / 4, 10, 5, 4 /

C Set the asf list to individual except for fill area interiors.
FORTRAN Example

DATA ASFLST / GINDIV,GINDIV,GINDIV,GINDIV,GINDIV,GINDIV,
1 GINDIV,GINDIV,GINDIV,GINDIV,GBUNDL,GBUNDL,GINDIV /

Data to plot:

DATA USDAT1 / 69.0, 50.0, 15.0, 53.0, 57.0, 150.0 /
DATA USDAT2 / 72.0, 50.0, 103.0, 0.0, 0.0, 56.0 /
DATA GDAT1 / 65.0, 42.0, 3.0, 89.0, 52.0, 93.0 /
DATA GDAT2 / 70.0, 53.0, 102.0, 0.0, 0.0, 49.0 /
DATA JDAT1 / 65.0, 47.0, 2.0, 60.0, 52.0, 55.0 /
DATA JDAT2 / 70.0, 57.0, 105.0, 0.0, 0.0, 41.0 /

Open GKS and activate a workstation.

CALL GOPKS ( ERROUT,NBYTES )
CALL GOPWK ( 1,TTOUT,WSTYPE )
CALL GACWK ( 1 )

Specify the window onto chart.

CALL GSWN ( 1,-115.0,160.0,-2.0,14.0 )
CALL GSELNT ( 1 )

Define the colors we'll be using.

CALL GSCR ( 1,1,0.0,1.0,0.0 )
CALL GSCR ( 1,2,1.0,0.0,0.0 )

Use bundled attributes except for fill area interior style and index.

CALL GSASF ( ASFLST )

Initialize the choice device.

CALL GQCHS ( 1,1,10,ERRIND,MODE,ESW,ISTAT,ICHNR,PET,EAREA, 1 LDR,DATREC )
CALL GPREC ( 0,IDUMMY,0,RDUMMY,4,CHLENS,CHSTRS,10,ERRIND, 1 LDR,DATREC )
CALL GINCH ( 1,1,GOK,1,3,EAREA( 1 ),EAREA( 2 ),EAREA( 3 ),
1 EAREA( 4 ),LDR,DATREC )

Get the users choice ( u.s, w. germany, japan, or exit ).

20 CONTINUE
CALL GRQCH ( 1,1,STATUS,CHOICE )

Display data depending on the choice.

IF ( CHOICE.EQ.1 ) THEN
   CALL DRAW ( USDAT1,USDAT2,CHSTRS(1) )
ELSE IF ( CHOICE.EQ.2 ) THEN
   CALL DRAW ( GDAT1,GDAT2,CHSTRS(2) )
ELSE IF ( CHOICE.EQ.3 ) THEN
CALL DRAW ( JDAT1, JDAT2, CHSTRS(3) )
ELSE
   GO TO 29
ENDIF
GO TO 20
29 CONTINUE
C
C Close the workstation and shut down GKS.
C
CALL GDAWK ( 1 )
CALL GCLKS
CALL GCLKS
STOP
END

SUBROUTINE DRAW ( DAT1, DAT2, NATION )
CHARACTER*(*) NATION
REAL DAT1(6), DAT2(6)
C
C Draw the border of the graph and plot the two sets of data associated
C with this country.
C
C Y-coordinate of the current data bar:
C
REAL POS
C
C Draw the border.
C
CALL BORDER( NATION )
C
C Draw the black bars.
C
CALL GSFAI ( 1 )
CALL GSTXCI ( 1 )
DO 11 I = 1,6
   POS = 2.0 * FLOAT( I - 1 ) + 1.6
   CALL BAR( DAT1( I ), POS )
11 CONTINUE
C
C Draw the red bars.
C
CALL GSFAI ( 2 )
CALL GSTXCI ( 2 )
CALL GSFAI ( 2 )
DO 21 I = 1,6
   POS = 2.0 * FLOAT( I - 1 ) + 0.8
   CALL BAR( DAT2( I ), POS )
21 CONTINUE
C
RETURN
END
SUBROUTINE BORDER( NATION )
CHARACTER(*) NATION
C
C Draw the border surrounding the data.
C
C GKS constants:
C
INTEGER GCONDI
PARAMETER ( GCONDI = 0 )
INTEGER GALEFT, GACENT, GAHALF, GATOP, GABOTT, GACAP
PARAMETER ( GALEFT = 1, GACENT = 2, GAHALF = 3, GATOP = 1 )
PARAMETER ( GABOTT = 5, GACAP = 2 )
C
C Coordinates of box surrounding bars:
C
REAL BOXX( 5 ), BOXY( 5 )
C
C Labels for the bars:
C
CHARACTER*16 LABEL( 6 )
DATA LABEL / 'LABOR', 'IRON ORE', 'COKE OR COAL',
1        'PURCHASED SCRAP', 'OTHER COSTS', 'OTHER ENERGY' /
C
C Initialize the box surrounding the bars:
C
DATA BOXX / 0.0, 150.0, 150.0, 0.0, 0.0 /
DATA BOXY / 0.0, 0.0, 12.0, 12.0, 0.0 /
C
C Clear the screen (if not already clear).
C
CALL GCLRWK ( 1, GCONDI )
C
C Draw the box surrounding the chart area.
C
CALL GPL ( 5, BOXX, BOXY )
C
C Draw the labels centered on the bar and flush left.
C
CALL GSTXAL ( GALEFT, GAHALF )
CALL GSCHH ( 0.5 )
CALL GSTXCI ( 1 )
DO 11 I = 1, 6
   POSY = 2.0 * FLOAT( I - 1 ) + 1.2
   CALL GTX (-114.0, POSY, LABEL( I ) )
11 CONTINUE
C
C Draw the top and bottom tick marks (bottom in red).
C
CALL GSTXAL ( GACENT, GABOTT )
CALL TICKS ( 12.0, 12.2 )
CALL GSTXAL ( GACENT, GATOP )
CALL GSTXCI ( 2 )
CALL TICKS ( 0.0, -0.2 )
CALL GSTXCI (1)
CALL GSTXAL (GACENT,GABOTT)
CALL GTX (0.0,-2.0,'PRODUCTION COST')
CALL GSCHH (0.7)
CALL GSTXAL (GACENT,GATOP)
CALL GTX (0.0,14.0,NATION)
CALL GSCHH (0.5)
RETURN
END

SUBROUTINE TICKS (TSTART,TEND)
REAL TSTART,TEND

CALL GSTXCI (1)
CALL GSTXAL (GACENT,GABOTT)
CALL GTX (0.0,-2.0,'PRODUCTION COST')
CALL GSCHH (0.7)
CALL GSTXAL (GACENT,GATOP)
CALL GTX (0.0,14.0,NATION)
CALL GSCHH (0.5)
RETURN
END

SUBROUTINE BAR (LENGTH,POS)
REAL LENGTH,POS
C
C This subroutine draws a horizontal bar of the specified length and
C at position pos.
C
C GKS constants:
INTEGER GALEFT, GACENT, GAHALF, GATOP, GABOTT
PARAMETER ( GALEFT = 1, GACENT = 2, GAHALF = 3, GATOP = 1 )
PARAMETER ( GABOTT = 5 )

Coordinates of the bar:

REAL BARX( 4 ), BARY( 4 )

If the value is too small, print a zero.

IF ( LENGTH.LE.0.0 ) THEN
  CALL GSTXAL ( GALEFT, GAHALF )
  CALL GTX ( 0.0, POS, '0' )
ELSE

Otherwise draw the bar.

  BARX( 1 ) = 0.0
  BARX( 2 ) = LENGTH
  BARX( 3 ) = LENGTH
  BARX( 4 ) = 0.0
  BARY( 1 ) = POS + 0.4
  BARY( 2 ) = POS + 0.4
  BARY( 3 ) = POS - 0.4
  BARY( 4 ) = POS - 0.4
  CALL GFA ( 4, BARX, BARY )
END IF

RETURN
END
Example 3:

C PROGRAM USMAP
C
C DESCRIPTION:
C This program reads a GKS metafile to draw a map of the U.S. The
C primitives in each state are in a separate segment. The user can
C then use a pick device to select the various states. A sampled
C choice device determines the action taken with the selected state.
C The choice number assignments are:
C 1. highlight the state
C 2. turn off highlighting
C 3. make the state visible
C 4. make the state invisible
C 5. exit
C
C CONFORMANCE:
C GKS level: lc
C The implementation must support at least one workstation of
C category outin and one of category mi (metafile input). The
C default choice device must support at least five choices.
C
C GKS constants:
C
INTEGER GOK,GNECHO,GSAMPL
PARAMETER ( GOK = 1, GNECHO = 0, GSAMPL = 1 )
INTEGER GNORML,GHILIT,GINVIS,GVISI
PARAMETER ( GNORML = 0, GHILIT = 1, GINVIS = 0, GVISI = 1 )

C Implementation dependent constants:
C
INTEGER ERROUT,TTOUT,WSTYPE,NBYTES
PARAMETER ( ERROUT = 5,TTOUT = 5, WSTYPE = 0, NBYTES = -1 )
INTEGER MI,MILUN
PARAMETER ( MI = 7, MILUN = 2 )

C Metafile item type, length and data record:
C
INTEGER MLDR,MAXLEN
PARAMETER ( MAXLEN = 500,MLDR = 500 )
INTEGER ITYPE,IDRL
CHARACTER*80 DATREC(MLDR)

C Input device status and values:
C
INTEGER STAT,SEG,PKID,CHNR

C Implementation dependent file open.
C
OPEN ( MILUN,STATUS='OLD' )
C
Open GKS and activate a workstation.
C
CALL GOPKS ( ERROUT,NBYTES )
CALL GOPWK ( 1,TTOUT,WSTYPE )
CALL GACWK ( 1 )
C
Set the choice device to sample mode.
C
CALL GSCHM ( 1,1,GSAMPL,GNECHO )
C
Open the metafile input workstation.
C
CALL GOPWK ( 2,MILUN,MI )
C
Interpret items until eof item is read.
C
10 CONTINUE
   CALL GGTITM ( 2,ITYPE,IDRL )
   IF ( ITYPE.EQ.0 ) GO TO 19
   CALL GRDITM ( 2,IDRL,MLDR,DATREC )
   CALL GIITM ( ITYPE,IDRL,DATREC )
   GO TO 10
19 CONTINUE
C
Close the metafile.
C
CALL GCLWK ( 2 )
C
Allow the user to select states until they select the 'exit' choice.
C
20 CONTINUE
   CALL GRQPK ( 1,1,STAT,SEG,PKID )
   IF ( STAT.NE.GOK ) GO TO 20
C
See which choice is in effect.
C
   CALL GSMCH ( 1,1,STAT,CHNR )
   IF ( STAT.NE.GOK ) GO TO 20
   GO TO ( 21,22,23,24,29 ) CHNR
21 CALL GSHLIT ( SEG,GHILIT )
   GO TO 20
22 CALL GSHLIT ( SEG,GNORM )
   GO TO 20
23 CALL GSVIS ( SEG,GINVIS )
   GO TO 20
24 CALL GSVIS ( SEG,GVISI )
29 CONTINUE
C
Close the workstation and shut down GKS.
C
CALL GDAWK ( 1 )
CALL GCLWK ( 1 )
CALL GCLKS
STOP
END
X3.113-1987 Programming Language FULL BASIC
X3.115-1984 Unformatted 80 Megabyte Trident Pack for Use at 370 tpi and 6000 bpi (General, Physical, and Magnetic Characteristics)
X3.116-1986 Recorded Magnetic Tape Cartridge, 4-Track, Serial 0.250 Inch (6.30 mm) 6400 bpi (252 bppm), Inverted Modified Frequency Modulation Encoded
X3.117-1984 Printable/Image Areas for Text and Facsimile Communication Equipment
X3.118-1984 Financial Services — Personal Identification Number — PIN Pad
X3.119-1984 Contact Start/Stop Storage Disk, 158361 Flux Transitions per Track, 8.268 Inch (210 mm) Outer Diameter and 3.937 inch (100 mm) Inner Diameter
X3.120-1984 Contact Start/Stop Storage Disk
X3.121-1984 Two-Sided, Unformatted, 8 Inch (200-mm), 48-tpi, Double-Density, Flexible Disk Cartridge for 13 262 fpp Two-Headed Application
X3.122-1986 Computer Graphics Metafile for the Storage and Transfer of Picture Description Information
X3.124-1985 Graphical Kernel System (GKS) Functional Description
X3.124.1-1985 Graphical Kernel System (GKS) FORTRAN Binding
X3.124.2-1988 Graphical Kernel System (GKS) PASCAL Binding
X3.125-1985 Two-Sided, Double-Density, Unformatted 5.25-Inch (130-mm), 48-tpi, Double-Density, Flexible Disk Cartridge for 7958 bpi Use
X3.126-1986 One-or Two-Sided Double-Density Unformatted 5.25-Inch (130-mm), 96 Tracks per Inch, Flexible Disk Cartridge
X3.127-1987 Unrecorded Magnetic Tape Cartridge for Information Interchange
X3.128-1986 Contact Start-Stop Storage Disk — 83 000 Flux Transitions per Track, 130-mm (5.118-Inch) Outer Diameter and 40-mm (1.575-Inch) Inner Diameter
X3.129-1986 Intelligent Peripheral Interface, Physical Level
X3.130-1986 Intelligent Peripheral Interface, Logical Device Specific Command Sets for Magnetic Disk Drive
X3.131-1986 Small Computer Systems Interface
X3.132-1987 intelligent Peripheral Interface — Logical Device Generic Command Set for Optical and Magnetic Disks
X3.133-1986 Database Language — NDL
X3.135-1986 Database Language — SQL
X3.136-1988 Serial Recorded Magnetic Tape Cartridge for Information Interchange, Four and Nine Track
X3.137-1988 Unformatted Flexible Disk Cartridge, 90 mm (3.5 Inch) 5.3 bpi (135 tpi) for 7958 bpi Use
X3.139-1987 Fiber Distributed Data Interface (FDDI) Token Ring Media Access Control (MAC)
X3.146-1987 Device Level Interface for Streaming Cartridge and Cassette Tape Drives
X3.147-1988 Intelligent Peripheral Interface — Device Generic Command Set for Magnetic Tape Drives
X3.148-1988 Fiber Distributed Data Interface (FDDI) — Token Ring Physical Layer Protocol (PHY)
X3.153-1987 Open Systems Interconnection — Basic Connection Oriented Session Protocol Specification
X3.156-1987 Nominal B-Inch Rigid Disk Removable Cartridge
X3.157-1987 Recorded Magnetic Tape for Information Interchange, 3200 CPI
X3.158-1987 Serial Recorded Magnetic Tape Cassette for Information Interchange, 0.150 Inch (3.81 mm), 8000 bpi (315 bippm), Group Code Recording
X3.162-1988 Two-Sided, High-Density, Unformatted, 5.25-Inch (130-mm), 96 Tacks per Inch, Flexible Disk Cartridge
X3.163-1988 Contact Start-Stop Metallic Film Storage Disk — 83 333 Flux Transitions per Track, 130 mm (5.118-in) Outer Diameter and 40-mm (1.575-in) Inner Diameter
X3.165-1988 Programming Language DIBOL
X11.1-1977 Programming Language MUMPS
IEEE 416-1978 Abbreviated Test Language for All Systems (ATLAS)
IEEE 716-1982 Standard C/ATLAS Language
IEEE 717-1982 Standard C/ATLAS Syntax
IEEE 770X3.97-1983 Programming Language PASCAL
ISO 8211-1986 Specifications for a Data Descriptive File for Information Interchange
NBS-ICST 1-1986 Fingerprint Identification — Data Format for Information Interchange

American National Standards for Information Processing

X3.1-1987 Synchronous Signaling Rates for Data Transmission
X3.4-1986 Coded Character Sets — 7-Bit ASCII
X3.5-1970 Flowchart Symbols and Their Usage
X3.6-1965 Perforated Tape Code
X3.9-1978 Programming Language FORTRAN
X3.11-1969 General Purpose Paper Cards
X3.14-1983 Recorded Magnetic Tape (200 CPI, NRZI)
X3.16-1976 Character Structure and Character Parity Sense for Serial-by-Bit Data Communication in the American National Standard Code for Information Interchange
X3.17-1981 Character Set for Optical Character Recognition (OCR-A)
X3.18-1974 One-Inch Perforated Paper Tape
X3.19-1974 Eleven-Sixteenths-Inch Perforated Paper Tape
X3.20-1967 Take-Up Reels for One-Inch Perforated Tape
X3.21-1967 Rectangular Holes in Twelve-Row Punched Cards
X3.22-1983 Recorded Magnetic Tape (800 CPI, NRZI)
X3.23-1985 Programming Language COBOL
X3.25-1976 Character Structure and Character Parity Sense for Parallel-by-Bit Data Communication in the American National Standard Code for Information Interchange
X3.27-1987 Magnetic Tape Labels and File Structure
X3.29-1971 Specifications for Properties of Unpunched Oiled Paper Perforator Tape
X3.30-1988 Representation for Calendar Date and Ordinal Date
X3.31-1988 Identification of the Counties of the United States
X3.34-1972 Interchange Rolls of Perforated Tape
X3.37-1987 Programming Language PTP
X3.38-1988 Identification of States of the United States (including the District of Columbia)
X3.39-1986 Recorded Magnetic Tape (1600 CPI, PE)
X3.40-1983 Unrecorded Magnetic Tape (9-Track 800 CPI, NRZI; 1600 CPI, PE; and 6250 CPI, GCR)
X3.41-1974 Code Extension Techniques for Use with the 7-Bit Coded Character Set of American National Standard Code for Information Interchange
X3.42-1975 Representation of Numeric Values in Character Strings
X3.43-1986 Representations of Local Time of Day
X3.44-1974 Determination of the Performance of Data Communication Systems
X3.45-1982 Character Set for Handprinting
X3.46-1974 Unrecorded Magnetic Six-Disk Pack (General, Physical, and Magnetic Characteristics)
X3.47-1988 Identification of Named Populated Places, Primary County Divisions, and Other Locational Entities of the United States
X3.48-1988 Magnetic Tape Cassettes (3.81 mm [0.150 Inch] Tape at 32 bpi [800 bpi], PE)
X3.49-1975 Character Set for Optical Character Recognition (OCR-B)
X3.50-1986 Representations for U.S. Coast, SI, and Other Units to Be Used in Scientific and Technical Systems with United Nations Units
X3.51-1986 Representations of Universal Time, Local Time, and Universal Time-Differential, and United States Time Zone References
X3.52-1976 Unrecorded Single-Disk Cartridge (Front Loading, 2200 BPI) (General, Physical, and Magnetic Requirements)
X3.53-1976 Programming Language PL/I
X3.54-1988 Recorded Magnetic Tape (6250 CPI, Group Coded Recording)
X3.55-1982 Unrecorded Magnetic Tape Cartridge, 0.250 Inch (6.30 mm), 1600 bpi (63 bps), Phase Encoded
X3.56-1986 Recorded Magnetic Tape Cartridge, 4 Track, 0.250 Inch (6.30 mm), 1600 bpi (63 bps), Phase Encoded
X3.58-1977 Unrecorded Eleven-Disk Pack (General, Physical, and Magnetic Requirements)
X3.60-1978 Programming Language Minimal BASIC
X3.61-1986 Representation of Geographic Point Locations
X3.62-1987 Paper Used in Optical Character Recognition (OCR) Systems
X3.63-1981 Unrecorded Twelve-Disk Pack (100 Megabytes) (General, Physical, and Magnetic Requirements)
X3.64-1979 Additional Controls for Use with American National Standard Code for Information Interchange
X3.65-1979 Advanced Data Communication Control Procedures (ADCCP)
X3.72-1981 Parallel Recorded Magnetic Tape Cartridge, 4 Track, 0.250 Inch (6.30 mm), 1600 bpi (63 bps), Phase Encoded
X3.73-1980 Single-Sided Unformatted Flexible Disk Cartridge
X3.74-1981 Programming Language PL/I, General-Purpose Subset
X3.76-1981 Unformatted Single-Disk Cartridge (Top Loading, 200 tpi 4400 bpi) (General, Physical, and Magnetic Requirements)
X3.77-1980 Representation of Pocket Select Characters
X3.78-1981 Representation of Vertical Carriage Positioning Characters in Information Interchange
X3.79-1981 Determination of Performance of Data Communications Systems That Use Bit-Oriented Communication Procedures
X3.80-1988 Interface between Flexible Disk Cartridge Drives and Their Host Controllers
X3.84-1981 Unformatted Twelve-Disk Pack (200 Megabytes) (General, Physical, and Magnetic Requirements)
X3.85-1981 1/2-Inch Magnetic Tape Interchange Using a Self Loading Cartridge
X3.86-1980 Optical Character Recognition (OCR) Inks
X3.89-1981 Unrecorded Single-Disk, Double-Density Cartridge (Front Loading, 2200 bpi, 200 tpi) (General, Physical, and Magnetic Requirements)
X3.91M-1987 Storage Module Interfaces
X3.92-1981 Data Encryption Algorithm
X3.93M-1981 OCR Character Positioning
X3.94-1985 Programming Language PANCM
X3.95-1982 Microprocessor – Hexadecimal Input/Output, Using 5-Bit and 7-Bit Teleprinters
X3.96-1983 Continuous Business Forms (Single-Part)
X3.98-1983 Text Information Interchange in Page Image Format (PIF)
X3.99-1983 Print Quality Guideline for Optical Character Recognition (OCR)
X3.100-1983 Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment for Packet Mode Operation with Packet Switched Data Communications Network
X3.101-1984 Interfaces Between Rigid Disk Drive(s) and Host(s)
X3.102-1983 Data Communication Systems and Services — User-Oriented Performance Parameters
X3.103-1983 Unrecorded Magnetic Tape Minicassette for Information Interchange, Coplanar 3.81 mm (0.150 Inch)
X3.104-1983 Recorded Magnetic Tape Minicassette for Information Interchange, Coplanar 3.81 mm (0.150 in), Phase Encoded
X3.105-1983 Data Link Encryption
X3.106-1983 Modes of Operation for the Data Encryption Algorithm
X3.108-1988 Physical Layer Interface for Local Distributed Data Interfaces to a Nonbranching Coaxial Cable Bus
X3.110-1983 Videotex/Teletext Presentation Level Protocol Syntax
X3.111-1986 Optical Character Recognition (OCR) Matrix Character Sets for OCR-M
X3.112-1984 14-in (356-mm) Diameter Low-Surface-Friction Magnetic Storage Disk

(Continued on reverse)

January 1989