ECMA

EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

STANDARD ECMA-96

SYNTAX OF GRAPHICAL DATA FOR MULTIPLE-WORKSTATION INTERFACE (GDS)

September 1985

Free copies of this document are available from ECMA,
European Computer Manufacturers Association
114 Rue du Rhône – 1204 Geneva (Switzerland)

ECMA
EUROPEAN COMPUTER MANUFACTURERS ASSOCIATION

STANDARD ECMA-96

SYNTAX OF GRAPHICAL DATA FOR MULTIPLE-WORKSTATION INTERFACE (GDS)

September 1985

F O R E W O R D

Due to particular conditions in the ECMA Headquarters during Summer 1985, this ECMA Standard is not presented in all respects according to the usual editing rules of the ECMA Secretariat. This should be considered exceptional. Future editions of this Standard will be presented in the usual layout.

BRIEF HISTORY

In March 1983 ECMA TC1 set up a Task Group on Picture Coding (TGPC) for the coding of geometric functions to be derived from the Graphical Kernel System (ISO 7942).

In the course of its study the ECMA Task Group decided - taking user requirements into account - to adopt a byte-oriented coding technique for a Multiple-Workstation Interface based on GKS rather than one for the GKS Functional Interface.

It was noted by ECMA that other coding schemes for related computer graphics applications were in development. In order not to introdice unnecessarily a new coding scheme, ECMA proposed to use only one unique coding scheme in all cases.

For this reason ECMA concluded to base its work on the coding scheme already adopted for the Videotex Data Syntax II specified in CCITT Rec. T.101 and in the corresponding CEPT Rec. T/CD 6.1. This decision was communicated to WG-8 of ISO/TC97/SC2 at its meeting of April 1984.

This point of view was stronlgy supported by CEPT. The following close ECMA/CEPT co-operation resulted in this ECMA Standard for a Syntax of Graphical Data for a Multiple-Workstation Interface, GDS (formerly GVDPLPS). CEPT document T/CD 6.1 was revised by CEPT in order to ensure that it constitutes a true sub-set of this ECMA Standard. ECMA and CEPT presented the proposed GDS Standard to WG-8 of ISO/TC97/SC2 during its 4th meeting in December 1984. As a result of detailed discussions a compromise on a unique coding scheme was adopted by ISO/TC97/SC2/WG-8 and by computer graphics experts of ISO/TC97/SC21/WG5-2 participating in this meeting.

This coding scheme, specified in the present ECMA Standard on GDS as well as in the corresponding CEPT Recommendation T/CD 6.1, Part 2, can be applied in both the computer graphics and the Telematic services environments such as:

- byte-coded bindings of ISO 7942 (GKS) and ISO DIS 8632 (CGM),
- Computer Graphics Interface CGI, formerly VDI,
- Videotex,
- a GKS Metafile,
- future Teletex geometric applications, also taking into account office document architecture (Standard ECMA-101).

Based on a 7-bit structure, the coding scheme may be used in both the 7-bit and the 8-bit environments.

Passed as an ECMA Standard at the General Assembly of June 13, 1985.

TABLE OF CONTENTS

Page

Page

1. INTRODUCTION

This standard specifies a Graphics Data Syntax for a multiple workstation interface. It is based on ISO 7942 Information Processing - Graphical Kernel System (GKS) - Functional description, therefore taking advantage of the work already done in the international computer graphics community.

GDS functionalities are based on the concept of a workstation as defined in GKS. Although the full GKS workstation concept can only be realized by GKS itself, this standard provides the capability to communicate groups of GKS functions to graphics devices.
Following the GKS definition GDS allows advantage to be taken of the different capabilities of the specific physical devices.

In order to have one syntax for the functions used in the computer graphics community, the encoding structure of GDS is based on the encoding structure as defined in CCITT Rec. T.101 (Data syntax II). The geometric display as defined in CEPT T/CD 6.1 (part 2) is a proper subset of GDS.

2. SCOPE AND FIELD OF APPLICATION

This document specifies the set of functions to be used in graphics equipement and their encoding in a 7-bit or 8-bit environment. In addition the code tables are structured in accordance with ECMA-6.

The intention of this document is to facilitate data interchange, not to standardize equipment. The specification of the concepts are included only to delimit the field of application. The definitions of the primitives may not be applicable to a physical device which does not conform to the specified concepts.

The graphics primitives contained in this document are derived from GKS. The set of primitives necessary in a physical device depends on the required GKS level.

Figure 1 shows the model describing the GKS environment and its interfaces. The Graphics application in the field of Computer Aided Engineering (CAE), Computer Aided Design (CAD), Business Graphics, Telematic Services etc. can be written in high level languages for which specific bindings with GKS are in the process of standardization.

A graphics application program, using GKS functions, communicates with the physical device through the multiple workstation interface. Above the multiple workstation interface are the GKS normalization transformation which convert world coordinates to normalized device coordinates. Below the multiple workstation interface are the GKS workstations connected to and driven by GKS. The workstations are mapped to the physical device and the normalized device coordinates are transformed to device coordinates by the workstation transformations. Non existent capabilities of the physical device must be simulated by using emulation software.

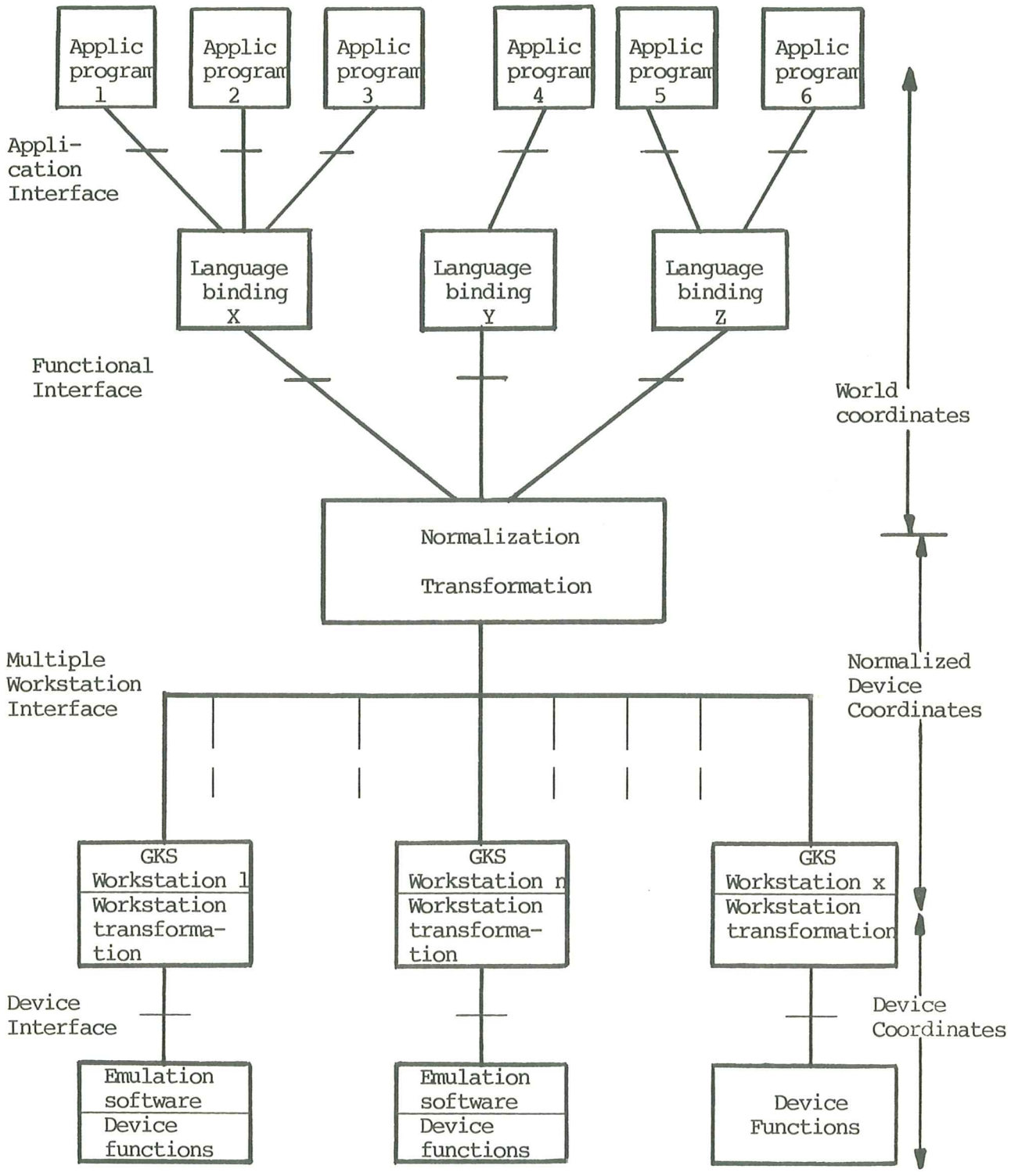


Figure 1

The Model Describing the GKS Environment and its Interfaces

The data syntax and encoding for the GKS multiple workstation interface is provided in this standard.

3. REFERENCES

ISO 7942 : INFORMATION PROCESSING
GRAPHICAL KERNEL SYSTEM (GKS) - Functional description.

ECMA-6 : 7-Bit Coded Character Set.

ECMA-35 : Code Extension Techniques.

ISO DP 8632 : INFORMATION PROCESSING SYSTEMS - Computer graphics. Metafile for the storage and transfer of picture description information.

CCITT Recommendation T.101 : International interworking for Videotex services.

4. DEFINITIONS AND ACRONYMS

This clause contains the definitions and acronyms of terms used in this document.

4.1 DEFINITIONS

ASPECT RATIO

The ratio of the width to the height of a rectangular area, such as workstation window or workstation viewport.
Example : an aspect ratio of 2:1 indicates an area twice as wide as it is high.

ASPECT SOURCE FLAG (ASF)

An indicator (flag) controlling whether the value of the associated attribute is obtained from a bundle table (BUNDLED) or from an individual specification (INDIVIDUAL).

ATTRIBUTE

A particular property that applies to a display element (output primitives) or a segment. Examples : highlighting, character height.

BASIC GRID UNIT (BGU)

A binary fraction that identifies the accuracy of coordinates.

BUNDLE

A set of attributes associated with one of the output primitives.

BUNDLE INDEX

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

BUNDLE TABLE

A workstation dependent table associated with a particular output primitive. Entries in the table specify all the workstation-dependent aspects of a display element. Bundle tables exist for the following output display elements : polyline, polymarker, text and fill area.

CELL ARRAY

Display element consisting of a parallelogram subdivided into parallelograms of equal size, each having a single colour. These cells do not necessarily map one-to-one with pixels.

CHOICE DEVICE

A logical input device providing a non negative integer defining one of a set of alternatives.

CLIPPING

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

CLIPPING RECTANGLE

A rectangle defined in NDC space used as a clipping boundary when the display elements have to be clipped.

COLOUR TABLE

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

DETECTABILITY

A segment attribute which makes the selection of a segment possible or not by a pick input device.

DEVICE COORDINATE (DC)

A coordinate expressed in a coordinate system that is device dependent.

DEVICE SPACE

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

DIFFERENTIAL CHAIN CODE (DCC)

A coding method used in incremental mode, identifying differences between steps (increments).

DISPLAY DEVICE

A device on which display images can be represented.

DISPLAY ELEMENT

A basic graphics element that can be used to construct a display image.

DISPLAY ELEMENT ATTRIBUTE

Display element attribute values (for output display elements) are selected by the application in a workstation-independent manner, but can have workstation-dependent effects.

DISPLAY IMAGE ; PICTURE

A collection of display elements that are represented together on a display surface.

DISPLAY SPACE

That portion of the device space corresponding to the area available for displaying images.

DISPLAY SURFACE ; VIEW SURFACE

The medium in a display device on which display images may appear (for example : the screen of a display, the paper in a plotter).

DOMAIN RING

A mechanism for defining the accuracy (number of bits) of the coordinate data.

ECHO

The immediate notification of the current value provided by an input device to the operator at the display surface.

FILL AREA

A display element consisting of a polygon (closed boundary) which may be hollow or may be filled with a uniform colour, a pattern or a hatch style.

FILL AREA BUNDLE TABLE

A table associating specific values of FILL AREA attributes with a fill area bundle index. This table contains entries consisting of fill area colour, fill area interior style and fill area style index.

FONT

A family or assortment of characters of a given size and style.

GDS ESCAPE

A mechanism used to access implementation or device dependent features, other than those for the generation of graphical output, otherwise not addressable by any primitive.

GENERALIZED DRAWING PRIMITIVE (GDP)

A display element (graphics primitive) used to address special geometrical workstation capabilities such as curve drawing.

GRAPHICAL KERNEL SYSTEM (GKS)

The application programmer's interface to graphics defined in ISO 7942.

GKS INSTANCE

A combination of GKS and one or more graphics equipment.

GRAPHICS DEVICE

An output device (for example : refresh display, storage tube display or plotter) on which display images can be represented.

HIGHLIGHTING

A device dependent way of emphasizing a segment by modifying its visual attributes (a generalisation of blinking).

INPUT CLASS

A set of input devices that are logically equivalent with respect to their function. The input classes are : LOCATOR, STROKE, VALUATOR, CHOICE, PICK and STRING.

LOCATOR DEVICE

A logical input device providing a position in normalized device coordinates.

LOGICAL INPUT DEVICE

A logical input device is an abstraction of one or more physical devices delivering a logical input value. Logical input devices can be of type LOCATOR, STROKE, VALUATOR, CHOICE, PICK and STRING.

LOGICAL INPUT VALUE

A value delivered by a logical input device.

MARKER

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

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.

NORMALIZED DEVICE COORDINATE (NDC)

A coordinate specified in a device independent intermediate coordinate system, normalized to some range.

OPERATOR

A person manipulating physical input devices in order to validate the measures of logical input devices.

OUTPUT PRIMITIVE

A display element (graphics primitive) that actually generates (parts of) a display image. These are : POLYLINE, FILL AREA, POLYMARKER, CELL ARRAY, TEXT and GDP's.

PICK DEVICE

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

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.

PICTURE

See display image.

PICTURE ELEMENT

See pixel.

PIXEL ; PICTURE ELEMENT

The smallest element of a display surface that can be independently assigned a colour or intensity.

POLYLINE

A display element consisting of a set of connected lines.

POLYLINE BUNDLE TABLE

A table associating specific values for all workstation dependent aspects of a polyline display element with a polyline bundle index. This table contains entries consisting of line type, line width and colour index.

POLYMARKER

A display element consisting of a set of locations, each to be indicated by a marker.

POLYMARKER BUNDLE TABLE

A table associating specific values for all workstation-dependent aspects of a polymarker display element with a polymarker bundle index. This table contains entries consisting of marker type, marker size and colour index.

PRIMITIVE ATTRIBUTE

Primitive attribute values (for output primitives) are selected in a workstation independent manner, but can have workstation dependent effects.

PROMPT

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

RING

A square defined by its radius and angular resolution factor, used for encoding increments in the incremental mode.

ROTATION

Turning all or part of a display image about an axis. Rotation is restricted to segments.

SCALING

Enlarging or reducing all or part of a display image by multiplying the coordinates of the display elements by a constant value. Scaling is restricted to segments.

SEGMENT

A collection of display elements that can be manipulated as a unit.

SEGMENT ATTRIBUTES

Attributes that apply only to segments. They are visibility, highlighting, detectability, segment priority and segment transformation.

SEGMENT PRIORITY

A segment attribute used to determine which of several overlapping segments take precedence for graphics output and input.

SEGMENT TRANSFORMATION

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

STRING DEVICE

A logical input device providing a character string.

STROKE DEVICE

A logical input device providing a sequence of points in normalized device coordinates.

TEXT

A display element consisting of a character string.

TEXT BUNDLE TABLE

A table associating specific values for all workstation-dependent aspects of a text display element with a text bundle index. This table contains entries consisting of text font and precision, character expansion factor, character spacing and colour index.

TEXT FONT AND PRECISION

An aspect of text 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, CHARACTER and STROKE.

TRANSLATION

The application of a constant displacement to the position of all or part of a display image. Translation is restricted to segments.

TRIGGER

A physical input device or set of devices which an operator can use to indicate significant moments in time.

VALUATOR DEVICE

A logical input device providing a real number.

VIEW SURFACE

See display surface.

VISIBILITY

A segment attribute which makes a segment to be displayed or not.

WORKSTATION

The concept of an abstract graphics device which provides the logical interface through which the physical devices are controlled.

WORKSTATION TRANSFORMATION

A transformation that maps the boundary and interior of a workstation window to the boundary and interior of a workstation viewport preserving aspect ratio. It maps positions in normalized device coordinates to device coordinates.

WORKSTATION VIEWPORT

A portion of device coordinate space currently selected for both input and output operations.

WORKSTATION WINDOW

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

4.2 ACRONYMS

ASAP	As Soon As Possible.
ASF	Aspect Source Flag.
ASTI	At Some Time.
BGU	Basic Grid Unit.
BNIG	Before the Next Interaction Globally
BNIL	Before the Next Interaction Locally
CDP	Conjugate Diameter Pair
DC	Device Coordinates
DCC	Differential Chain Code
GDP	Generalized Drawing Primitive
GDS	Graphics Data Syntax for a multiple workstation interface
GKS	Graphical Kernel System
IMM	IMMediately
INPUT	Input only workstation
IRG	Implicit ReGeneration
NDC	Normalized Device Coordinates
OUTIN	Output and Input workstation
OUTPUT	Output only workstation
WDSS	Workstation Dependent Segment Storage.
WISS	Workstation Independent Segment Storage.

5. GENERAL DESCRIPTION

This clause provides a description of all the concepts involved in graphics operations.

For this purpose two groups of basic elements are introduced : PRIMITIVES and ATTRIBUTES.

The PRIMITIVES are abstractions of basic actions a graphics device can perform, such as drawing lines. The PRIMITIVE ATTRIBUTES specify the characteristics of the OUTPUT PRIMITIVES on a display device, such as colour and line thickness.

Another main concept is that of GRAPHICS WORKSTATION which can be regarded as the abstraction of the collection of graphics input and output devices.

Two coordinate systems are provided :

a) Normalized Device Coordinates (NDC) used to define a uniform coordinate system for all graphics workstations.

b) Device coordinates (DC), the actual coordinate system of the physical device representing its addressable space.

OUTPUT PRIMITIVES and their ATTRIBUTES may be grouped together in SEGMENTS.

SEGMENTS are collections of display elements that can be manipulated and changed as a unit.

In a GDS implementation some primitives are mandatory whereas others are not. Appendix A lists all the primitives defined on the standard and specifies which are mandatory and which are optional.

5.1 GRAPHICAL OUTPUT

The basic display elements from which a picture is built up are defined by output primitives. The display elements are specified by their geometry and by their appearance on the display surface of a workstation. These aspects are controlled by a set of attributes that belong to a display element. Certain attributes may vary from one workstation to another ; therefore they are called workstation-dependent attributes.

There are primitives for the creation of display elements and for the setting of attributes. Examples of different display elements are shown in figure 2.

5.1.1 OUTPUT PRIMITIVES

The following primitives for the creation of display elements are provided :

- POLYLINE

The display element to be created is a set of connected straight lines defined by a sequence of points.

- POLYMARKER

The display element consists of symbols centred at given positions. The symbols, called markers, are glyphs with specified appearances which are used to identify a set of locations.

- TEXT

The display element is a character string placed at a given position.

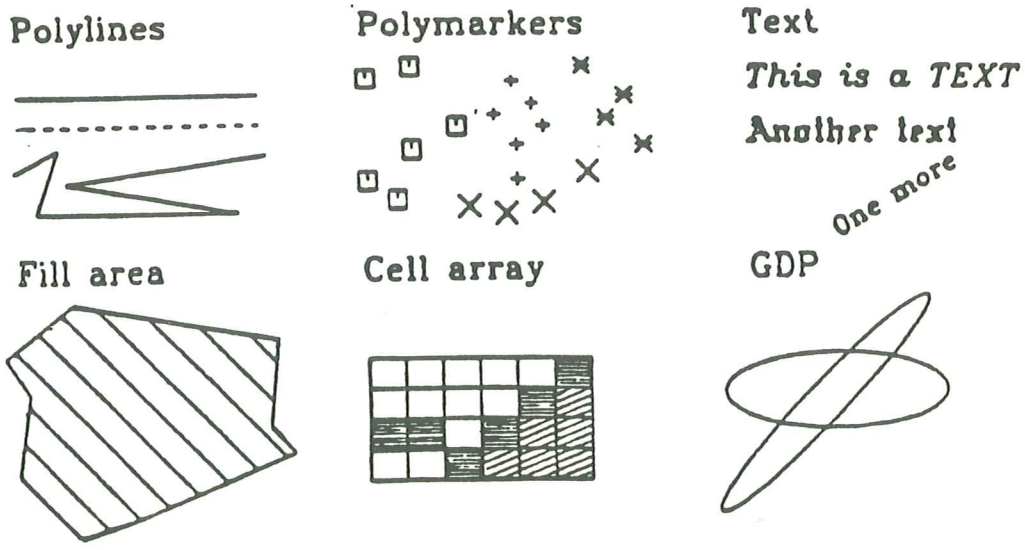


Fig. 2 - Examples of Display elements

- FILL AREA

The display element is an area closed by a set of connected straight lines that may be hollow or filled with uniform colour, a pattern or a hatch style.

- CELL ARRAY

The display element defines a parallelogram of equal sized parallelogram cells with individual colours.

- Generalized drawing primitives (GDP)

A general display element addressing special geometrical drawing capabilities of a graphics terminal. All transformations are applied to the points of GDP but the interpretation is workstation-dependent.

The standardized GDPs are :

. RECTANGLE

The display element is an area with a rectangular boundary that may be hollow or filled.

. CIRCLE

The display element is an area with a circle as boundary that may be hollow or filled.

. CIRCULAR ARC 3 POINT

The display element is a circular arc drawn from the starting point, through the specified intermediate point, to the specified ending point.

. CIRCULAR ARC 3 POINT CHORD

The display element is an area closed by a circular arc and a chord that may be hollow or filled.

. CIRCULAR ARC 3 POINT PIE

The display element is an area closed by a circular arc and the two radii that may be hollow or filled.

. CIRCULAR ARC CENTRE

The display element is a circular arc defined by a centre, a radius, a starting vector and a ending vector.

. CIRCULAR ARC CENTRE CHORD

The display element is an area closed by a circular arc and a chord that may be hollow or filled. The circular arc is defined as a circular arc centre.

. CIRCULAR ARC CENTRE PIE

The display element us is area closed by a circular arc and the two radii that may be hollow or filled. The circular arc is defined as a circular arc centre.

. ELLIPSE

The display element is an area with an ellipse as a boundary that may be hollow or filled.

. ELLIPTIC ARC

The display element is an elliptical arc.

. ELLIPTIC ARC CHORD

The display element is an area closed by an elliptic arc and a chord that may be hollow or filled.

. ELLIPTIC ARC PIE

The display element is an area closed by an elliptic arc and the two radii that may be hollow or filled.

. SPLINE

The display element is a smooth curve drawn through a series of control points (uniform quadratic B-SPLINE).

Non standardized GDP : non standardized GDPs may be defined for private use ; each GDP is specified by a negative valued identifier, a set of points and additional data.

5.1.2 Output primitive attributes

Three types of attributes (geometric attributes, non-geometric attributes and identification) can potentially be specified for each display element. The first two attributes determine the appearance of the display elements while the third is used in connection with graphical input. The values of the attributes can be set modally. During creation of a display element these values are bound to the display element and cannot be changed afterwards.

Geometric attributes control the geometric aspect of display elements which affect shape or size. Hence they are workstation independent. Nongeometric attributes merely affect the appearance (for example line type for POLYLINE) of the display elements. The non-geometric attributes for each primitive may be specified by means of a bundle or individually. For specification of aspects by means of a bundle, there is one attribute per display element which is an index into the bundle table.

For each display element (except for generalized display elements and CELL ARRAY) there is a bundle table ; an entry of such a table contains all the non-geometric aspects of a display element.

In this specification mode, the non-geometric attributes are workstation dependent and each workstation has its own set of bundle tables with different values in a particular bundle for different workstations. For individual specification of aspects, there is a separate attribute for each non-geometric aspect. With this specification mode these attributes are workstation-independent.

The values that can be assigned to a non-geometric attribute are the same in both specification modes, but in bundled mode the values are restricted to the valid value of each particular workstation. In individual modes if an invalid value of an attribute is set, default actions for the display element are defined to occur.

Generalized drawing primitives and CELL ARRAY do not have associated bundle tables or corresponding individually specified attributes. For each generalized drawing primitive, the bundle tables and sets of individually specified attributes are specified to be used when creating the display element. CELL ARRAY contains colour index information but no other non-geometric aspects.

The method of specification of the non-geometric aspect of a display element may be chosen separately for each aspect. For each non-geometric aspect of each display element exists the attribute ASPECT SOURCE FLAG that takes the values INDIVIDUAL and BUNDLED to specify the choice. The initial values of flags are defined in clause 9 (defaults). The values of the flags may be changed by the primitive SET ASPECT SOURCE FLAGS. This enables some non-geometric aspects of a primitive to be specified individually and others bundled.

When a display element is created, the values of the non-geometric attributes with which it is displayed are determined as follows :

- if the ASF of an aspect is INDIVIDUAL, the value used on all workstations is the value of the corresponding individually specified attribute of that display element ;

- if the ASF of an aspect is BUNDLED, the value used on a workstation is obtained via the bundle table for that display element on the workstation ; the corresponding component of the bundle table entry, pointed to by the bundle index, is used.

If colour is a non-geometric aspect of a display element, it is specified as an index into an unique colour table of each workstation. Similarly other attributes values may be indices into specific workstation tables or fixed lists.

There is one attribute of the third type per display element, called PICK IDENTIFIER which is used for identifying a display element or a group of display elements in a segment, when that display element or group is picked.

The PICK IDENTIFIER is only used when workstations support input facilites.

The attributes which apply to each display element are :

a) POLYLINE
- POLYLINE INDEX
- LINE TYPE
- LINE WIDTH SCALE FACTOR
- POLYLINE COLOUR INDEX
- LINE TYPE ASF
- LINE WIDTH ASF
- POLYLINE COLOUR INDEX ASF
- PICK IDENTIFIER

b) POLYMARKER
- POLYMARKER INDEX
- MARKER TYPE
- MARKER SIZE SCALE FACTOR
- POLYMARKER COLOUR INDEX
- MARKER TYPE ASF
- MARKER SIZE ASF
- POLYMARKER COLOUR INDEX ASF
- PICK IDENTIFIER

c) 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 VECTORS
- TEXT PATH
- TEXT ALIGNMENT
- PICK IDENTIFIER

d) 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 VECTORS
- PATTERN REFERENCE POINT
- PICK IDENTIFIER

e) CELL ARRAY
- PICK IDENTIFIER

f) GENERALIZED DISPLAY ELEMENT — Zero or more of the sets a) to e) except that PICK IDENTIFIER is always an attribute.

RECTANGLE	set d)
CIRCLE	set d)
CIRCULAR ARC 3 POINT CHORD	set d)
CIRCULAR ARC 3 POINT PIE	set d)
CIRCULAR ARC CENTRE CHORD	set d)
CIRCULAR ARC CENTRE PIE	set d)
ELLIPSE	set d)
ELLIPTIC ARC CHORD	set d)
ELLIPTIC ARC PIE	set d)
CIRCULAR ARC 3 POINT	set a)
CIRCULAR ARC CENTRE	set a)
ELLIPTIC ARC	set a)
SPLINE	set a)

The entries in the bundle, pattern and colour tables may be set separately for each workstation. The tables, which are on every workstation with output facilities, 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 changed. The criterium 'dynamic modification accepted' associated with each aspect in a workstation indicates which changes :

- lead to an implicit regeneration (may be deferred);
- can be performed immediately.

Some standard definitions for table entries are contained in a workstation and are used as initial values. Only the most commonly used combinations of values need to be predefined for each output type workstation. Predefined entries with indices up to the minimum number of predefined entries at a given level (see 5.10) must be distingishable from each other.

5.1.2.1 POLYLINE attributes

POLYLINE has no geometric attribute. The following attributes control the representation of a polyline :

- POLYLINE COLOUR INDEX
  Determines the colour index with which the lines will be drawn. It is controlled with SET POLYLINE COLOUR INDEX and is used if the 'Polyline colour' ASF is INDIVIDUAL.

- LINE WIDTH SCALE FACTOR
  Determines the scale factor applied to the nominal line width of a workstation. It is controlled with SET LINE WIDTH SCALE FACTOR and used if the 'line width' ASF is INDIVIDUAL.

- LINE TYPE
  Determines the type of the line: solid, dashed, dotted or dash-dotted etc.. The LINE TYPE is selected with SET LINE TYPE and is used if the 'line type' ASF is INDIVIDUAL.

- POLYLINE INDEX
  Determines the entry of the polyline bundle table to be used in drawing lines. The POLYLINE INDEX is specified with SET POLYLINE INDEX and used for BUNDLED ASFs.

- POLYLINE REPRESENTATION
  Determines the attribute values to be loaded in the specified entry of the polyline bundle table. The POLYLINE REPRESENTATION is specified with SET POLYLINE REPRESENTATION and contains the attributes: POLYLINE INDEX, LINE TYPE, POLYLINE COLOUR INDEX and LINE WIDTH SCALE FACTOR.

The polyline bundle table contains three attributes per entry : POLYLINE COLOUR INDEX, LINE TYPE, LINE WIDTH SCALE FACTOR.

5.1.2.2 POLYMARKER attributes

POLYMARKER has no geometric attributes. The following attributes control the representation of a polymarker :

- POLYMARKER COLOUR INDEX
  Determines the POLYMARKER COLOUR INDEX to be used in drawing the centred markers. It is controlled with SET POLYMARKER COLOUR INDEX and used if the 'polymarker colour' ASF is INDIVIDUAL

- MARKER TYPE
  Determines the type of the marker : a dot, a plus, a star, a circle or a diagonal cross, etc. The MARKER TYPE is selected with SET MARKER TYPE and is used if the 'marker type' ASF is INDIVIDUAL.

- MARKER SIZE SCALE FACTOR
  Determines the scale factor applied to the nominal marker size. The size is defined with SET MARKER SIZE SCALE FACTOR and is used when the 'marker size' ASF is INDIVIDUAL.

- POLYMARKER INDEX
  Determines the entry of the polymarker bundle table to be used in drawing markers. The POLYMARKER INDEX is specified with SET POLYMARKER INDEX and used for BUNDLED ASFs.

- POLYMARKER REPRESENTATION
  Determines the attribute values to be loaded in the specified entry of the polymarker bundle table. The POLYMARKER REPRESENTATION is specified with SET POLYMARKER REPRESENTATION and contains the attributes : POLYMARKER INDEX, MARKER TYPE, POLYMARKER COLOUR INDEX and MARKER SIZE SCALE FACTOR.

The polymarker bundle table contains three attributes per entry : POLYMARKER COLOUR INDEX, MARKER TYPE and MARKER SIZE SCALE FACTOR.

5.1.2.3 TEXT attributes

Text has the geometric attributes CHARACTER VECTORS, TEXT PATH and TEXT ALIGNMENT.

- CHARACTER VECTORS
  It represents the character height vector and width vector (defined by direction and length) determining the orientation, skew and distortion of the characters.

  The direction of the height vector fixes the character up vector; the distance from baseline to capline along the character up vector is the length of the height vector. The direction of the width vector fixes the baseline direction of the character. The length of the character width vector specifies the nominal width of the character. The actual width is the product of the length of the character width vector times the character expansion factor times the width to heigth ratio of the character.

  The CHARACTER VECTORS are specified with SET CHARACTER VECTORS.

- TEXT PATH
  Determines the writing direction of a text string. Up (respectively down) means in the direction (resp. opposite direction) of the height vector; right (resp. left) means in the direction (resp. op posite direction) of the width vector. The TEXT PATH is specified with SET TEXT PATH.

- TEXT ALIGNMENT
  Has two components which are horizontal and vertical alignments.

  .HORIZONTAL ALIGNMENT
  Determines the horizontal positioning of the text string in relation to the text position: Normal, Left, Centre or Right. The HORIZONTAL ALIGNMENT is specified with SET TEXT ALIGNMENT.

  .VERTICAL ALIGNMENT
  Determines the vertical positioning of the text string in relation to the text position: Normal, Top, Cap, Half, Base or Bottom. The VERTICAL ALIGNMENT is specified with SET TEXT ALIGNMENT.

The representation of text at a workstation is controlled by :

- TEXT COLOUR INDEX
  Determines the COLOUR INDEX of the generated text string. SET TEXT COLOUR INDEX specifies the COLOUR INDEX and is used if the 'text colour' ASF is INDIVIDUAL.

- TEXT INDEX
  Determines the entry of the text bundle table to be used in drawing strings. The TEXT INDEX is specified with SET TEXT INDEX and used for BUNDLED ASFs

- CHARACTER SPACING
  Determines how much additional space is to be inserted between characters. If the value of CHARACTER SPACING is zero, the characters are arranged one after each other along the TEXT PATH.

The CHARACTER SPACING may be negative or positive. The CHARACTER SPACING is specified as a fraction of the length of the character height vector. The CHARACTER SPACING is specified with SET CHARACTER SPACING and is used if the 'character spacing' ASF is INDIVIDUAL.

- TEXT FONT AND PRECISION
  Has two components which are TEXT FONT and TEXT PRECISION :
  - . TEXT FONT
    Determines the TEXT FONT to be used in generating text strings. Each display device should support at least one TEXT FONT which is text font number 0. The TEXT FONT is selected with SET TEXT FONT AND PRECISION and is used if the 'text font and precision' ASF is INDIVIDUAL.
  - . TEXT PRECISION
    Determines the accuracy with which a text string is generated: String, Character or Stroke. The TEXT PRECISION is selected with SET TEXT FONT AND PRECISION and is used if the 'text font and precision' ASF is INDIVIDUAL.

- CHARACTER EXPANSION FACTOR
  Determines the deviation of the width to height ratio of the characters from the width to height ratio indicated by the font designer. The CHARACTER EXPANSION FACTOR is defined by SET CHARACTER EXPANSION FACTOR and is used if the 'character expansion factor' ASF is INDIVIDUAL.

- TEXT REPRESENTATION
  Determines the attribute values to be loaded in the specified entry of the text bundle table. The TEXT REPRESENTATION is specified with SET TEXT REPRESENTATION and contains the attributes : TEXT INDEX, TEXT COLOUR INDEX, CHARACTER EXPANSION FACTOR, CHARACTER SPACING and TEXT FONT AND PRECISION.

HORIZONTAL and VERTICAL ALIGNMENTS both can have 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. The following list applies :

TEXT PATH	Normal HORIZONTAL and VERTICAL ALIGNMENT
Right	(Left, Base)
Left	(Right, Base)
Up	(Centre, Base)
Down	(Centre, Top)

The characters defined in a particular text font are display device dependent. Fonts are defined in a local 2D cartesian coordinate system. Fonts are either monospaced or proportionally spaced. Each character has an associated character body, a font baseline, a font half line, a capline and a centre line, (see figure 3).

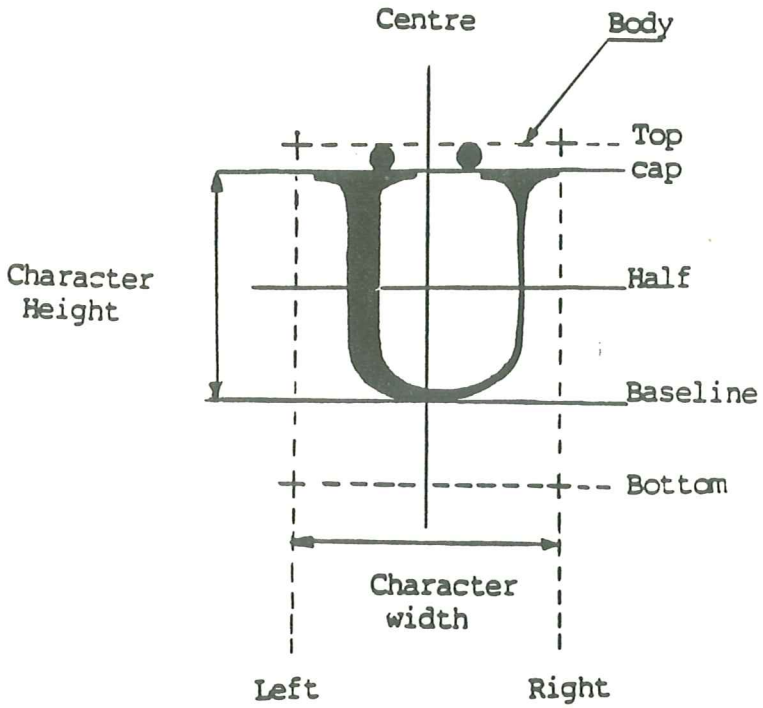


Figure 3

Font description coordinate system

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 must be parallel to the character vectors. The font baseline and the capline must be parallel to the width vector and within the vertical extent of the body. The centre line is parallel to the height vector and bisects the body.

The height of a character in the font coordinate system is given by the height from the font baseline to the capline. The width may include space on either side of the character. It is given by the width of the character body. 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 will be identical with or very close to the typographical capline or ascender line and the bottom limit to the descender line. 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 ascender/descender clashes.

Figures 4 to 8 are only inserted for clarification purposes. They show the effect of the different attributes on the display of the text "ABCD".

The characters in the text string can be specified in a 7-bit or 8-bit environment as defined by ECMA-35. The characters may be taken from the invoked G set (i.e. from columns 2 to 7) in a 7-bit environment, or from both invoked G sets (i.e. from columns 02-07 or 10-15) in an 8-bit environment.

Text strings are delimited by SOS and ST (See section 7.2.4). These characters are not considered as a part of the character string.

Besides the characters from the in-use G set, in a 7-bit environment, the shift functions contained in table 1 may be used (ESC = 1/11).

Abbreviation	Name
SI	SHIFT-IN
SO	SHIFT-OUT
LS2	LOCKING-SHIFT 2
LS3	LOCKING-SHIFT 3
SS2	SINGLE-SHIFT 2
SS3	SINGLE-SHIFT 3

Table 1

Permitted shift function in a 7-bit environment

In an 8-bit environment the shift functions contained in table 2 may be used (ESC = 01/11).

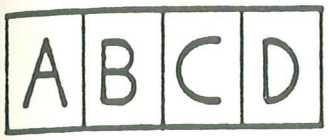
Abbreviation	Name
LS0	LOCKING-SHIFT 0
LS1	LOCKING-SHIFT 1
LS1R	LOCKING-SHIFT 1 RIGHT
LS2	LOCKING-SHIFT 2
LS2R	LOCKING-SHIFT 2 RIGHT
LS3	LOCKING SHIFT 3
LS3R	LOCKING SHIFT 3 RIGHT
SS2	SINGLE-SHIFT 2
SS3	SINGLE-SHIFT 3

Table 2

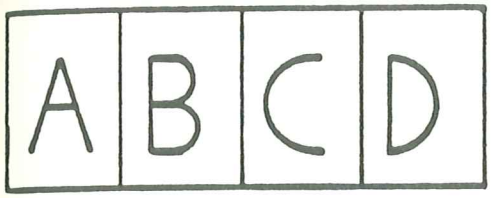
Permitted shift functions in an 8-bit environment

Characters in the text string not from the invoked G set or not from Table 1 in a 7-bit environment, will be ignored. In a 8-bit environment characters not from the invoked G sets or not from Table 2 will be ignored.

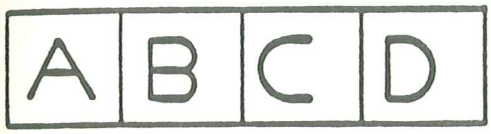
Format control characters (such as CR or LF) in a string are allowed, but have no effect.



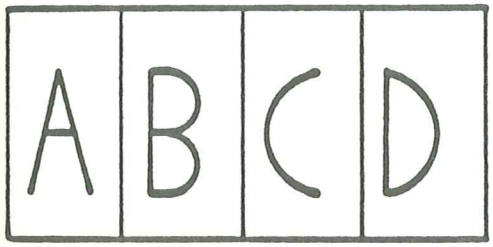
CHARACTER VECTORS = (0.0, 0.025) and
(0.025, 0.0)
CHARACTER EXPANSION FACTOR = 1.0



CHARACTER VECTORS = (0.0, 0.0375) and
(0.0375, 0.0)
CHARACTER EXPANSION FACTOR = 1.0

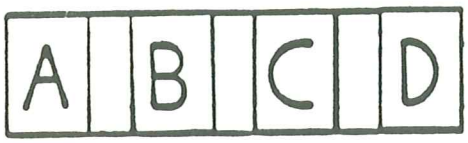


CHARACTER VECTORS = (0.0, 0.025) and
(0.025, 0.0)
CHARACTER EXPANSION FACTOR = 1.5

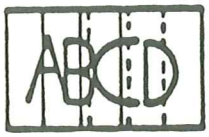


CHARACTER VECTORS = (0.0, 0.050) and
(0.050, 0.0)
CHARACTER EXPANSION FACTOR = 0.75

Figure 4 : CHARACTER VECTORS and CHARACTER EXPANSION FACTOR



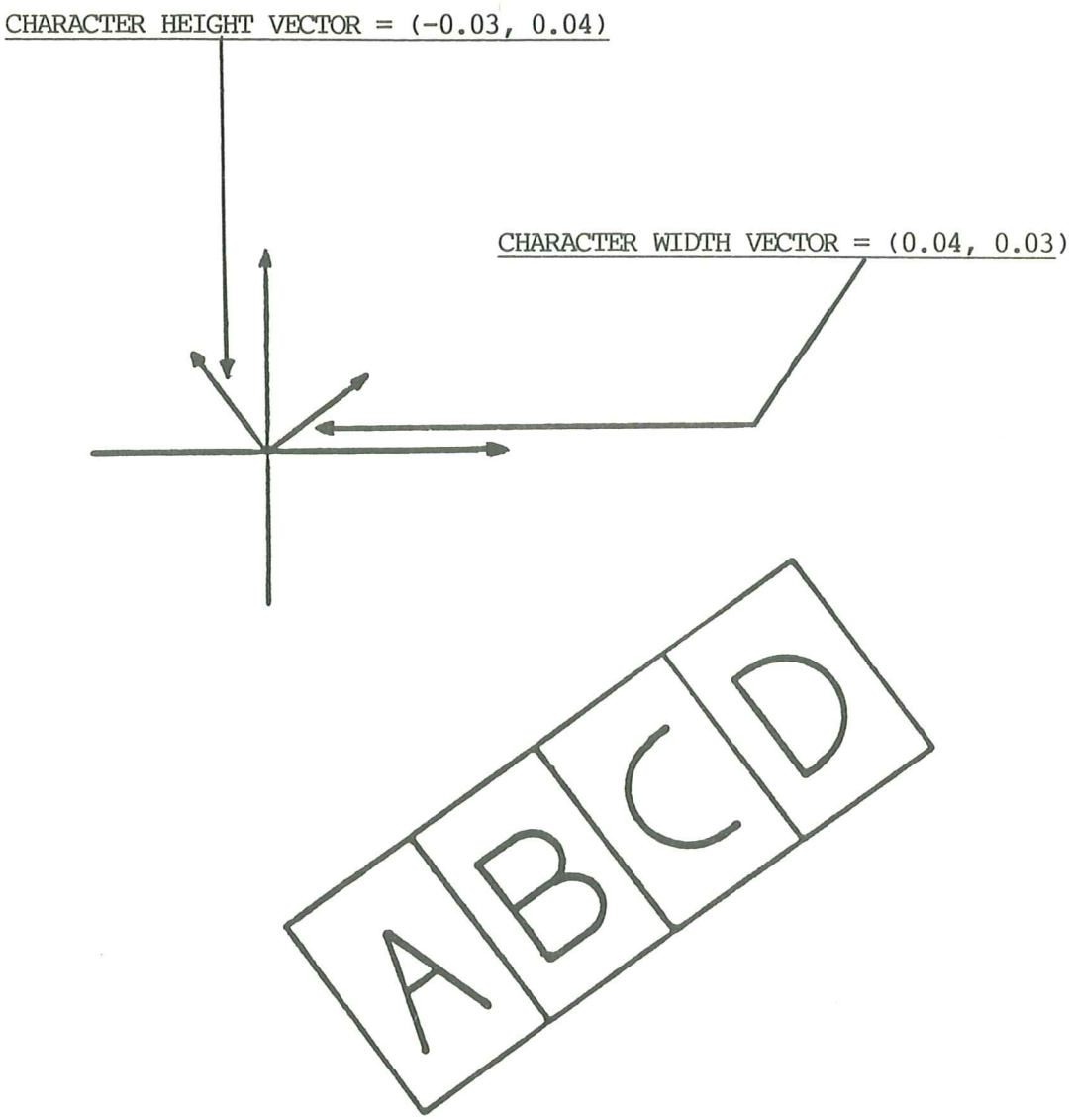
CHARACTER VECTORS =
(0.0, 0.025) and (0.025, 0.0)
CHARACTER SPACING = 0.67
TEXT PATH = right



CHARACTER VECTORS = (0.0, 0.025) and
(0.025, 0.0)
CHARACTER SPACING = -0.67
TEXT PATH = right

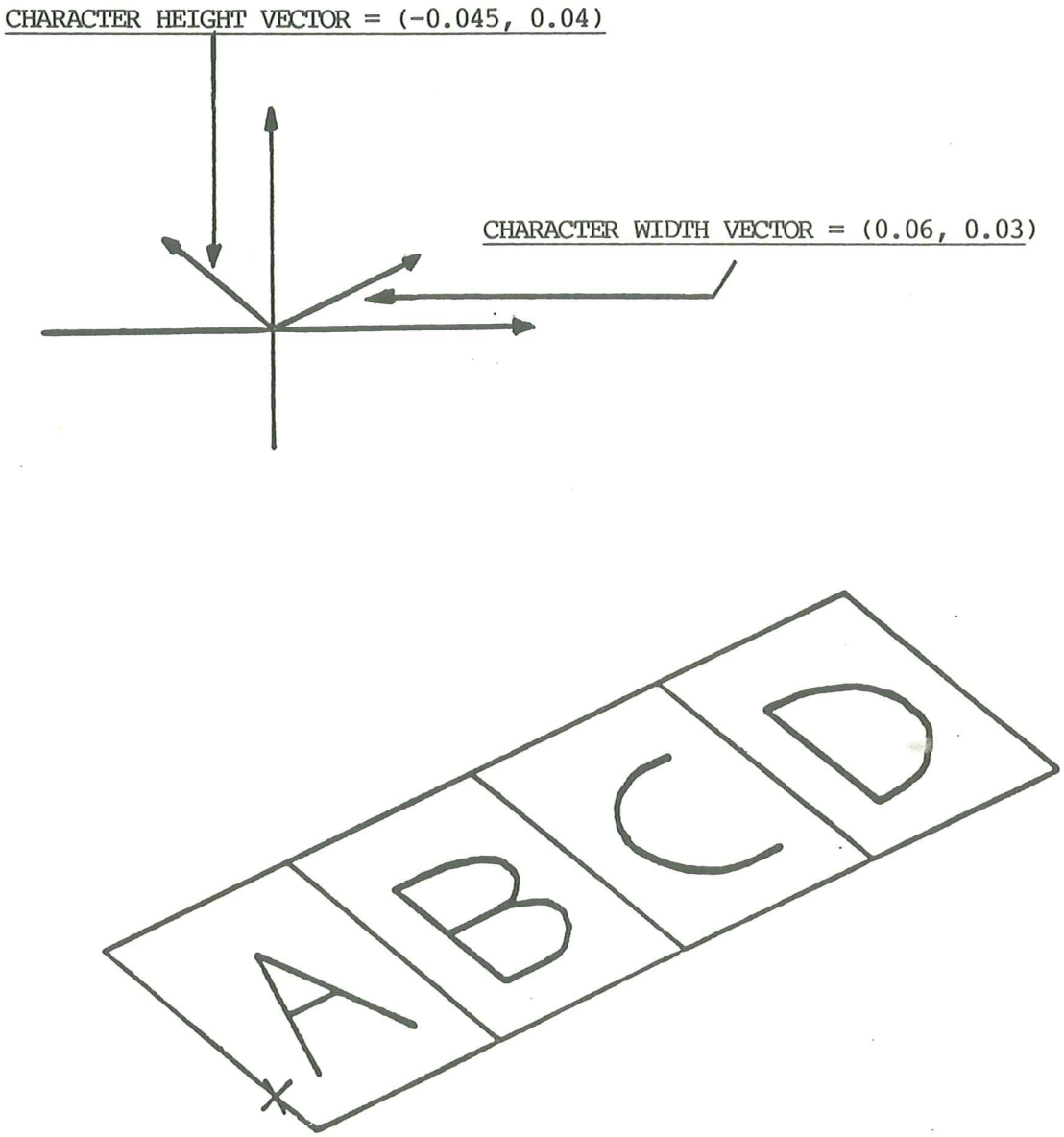
CHARACTER VECTORS = (0.0, 0.025) and
(0.025, 0.0)
CHARACTER SPACING = 2.0
TEXT PATH = down

Figure 5 : CHARACTER SPACING



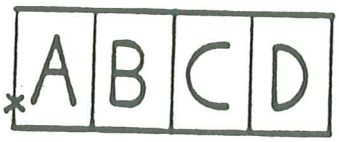
CHARACTER VECTORS = (-0.03, 0.04) and (0.04, 0.03)
TEXT PATH = right

Figure 6 : CHARACTER VECTORS

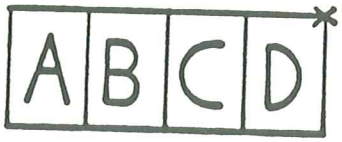


CHARACTER VECTORS = (-0.045, 0.04) and (0.06, 0.03)
TEXT PATH = right

Figure 7 : CHARACTER VECTORS after anisotropic transformation



TEXT ALIGNMENT = (left, base)
TEXT PATH = right



TEXT ALIGNMENT = (right, top)
TEXT PATH = right

TEXT ALIGNMENT = (centre, bottom)
TEXT PATH = down

TEXT ALIGNMENT = (left, half)
TEXT PATH = down

Figure 8 : TEXT ALIGNMENT and TEXT PATH

5.1.2.4 FILL AREA attributes

FILL AREA has the geometric attributes PATTERN REFERENCE POINT and PATTERN VECTORS.

- PATTERN VECTORS
  Determines the size of the parallelogram in which the pattern cells are defined. The PATTERN VECTORS are defined with SET PATTERN VECTORS and used when the selected (either BUNDLED or INDIVIDUALLY) INTERIOR STYLE is Pattern.

- PATTERN REFERENCE POINT
  Determines the position of the start of the pattern. The PATTERN REFERENCE POINT is defined with SET PATTERN REFERENCE POINT and used when the selected (either BUNDLED or INDIVIDUALLY) FILL AREA INTERIOR STYLE is Pattern.

The representation of FILL AREA at a workstation is controlled by :

- FILL AREA COLOUR INDEX
  Determines the COLOUR which is used to fill the closed boundary. It is controlled with SET FILL AREA COLOUR INDEX and used if the 'fill area colour' ASF is INDIVIDUAL

- FILL AREA INTERIOR STYLE
  Determines how the closed boundary is filled : Hollow, Solid, Pattern or Hatch. The FILL AREA INTERIOR STYLE is selected with SET FILL AREA INTERIOR STYLE and used if the 'fill area interior style' ASF is INDIVIDUAL.

- FILL AREA STYLE INDEX
  Determines for FILL AREA INTERIOR STYLE = Hatch, the hatch type to be used and for FILL AREA INTERIOR STYLE = Pattern, the entry from the pattern table to be used. The FILL AREA STYLE INDEX is selected with SET FILL AREA STYLE INDEX and used when the 'fill area style index' ASF is INDIVIDUAL.

- FILL AREA INDEX
  Determines the entry of the fill area bundle table to be used in filling areas. The FILL AREA INDEX is specified with SET FILL AREA INDEX and used for BUNDLED ASFs.

- FILL AREA REPRESENTATION
  Determines the attribute values to be loaded in the specified entry of the fill area bundle table. The FILL AREA REPRESENTATION is specified with SET FILL AREA REPRESENTATION and contains the attributes : FILL AREA INDEX, FILL AREA COLOUR INDEX, FILL AREA INTERIOR STYLE and FILL AREA STYLE INDEX.

The FILL AREA bundle contains three attributes per entry : FILL AREA COLOUR INDEX, FILL AREA INTERIOR STYLE and FILL AREA STYLE INDEX.

The pattern table contains the following attributes per entry :

- PATTERN DIMENSIONS (DX, DY) which define the number of cells in horizontal and vertical directions,

- PATTERN CELL COLOUR INDEX LIST which determines a colour index value for each of the defined cells.

An entry in the pattern table is defined by SET PATTERN REPRESENTATION.

For interior style Pattern, the pattern is defined by the pattern representation, which specifies a pattern cell colour index list, which is conceptually an array (DX* DY) of colour indices, that are pointers into the colour table. The size and position of the start of the pattern are defined by a pattern box. The pattern box, which is a paparallelogram, is defined by the PATTERN VECTORS located relative to the PATTERN REFERENCE POINT. The pattern box is conceptually divided into a grid of DX * DY equally sized 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 closed boudary by conceptually replicating the transformed pattern box in didirections parallel to its sides until the interior of the complete closed boundary is covered.

5.1.2.5 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.

5.1.2.6 GDP attributes

The GDP primitives that generate closed boundaries :

RECTANGLE
CIRCLE
CIRCULAR ARC 3 POINT CHORD
CIRCULAR ARC 3 POINT PIE
CIRCULAR ARC CENTRE CHORD
CIRCULAR ARC CENTRE PIE
ELLIPSE
ELLIPTIC ARC CHORD
ELLIPTIC ARC PIE

use the FILL AREA attributes. These are described in 5.1.2.4.
The GDP primitives that do not generate closed boundaries :
CIRCULAR ARC 3 POINT
CIRCULAR ARC CENTRE
ELLIPTIC ARC
SPLINE
use the POLYLINE attributes. These are described in 5.1.2.1.

5.1.2.7 Colour

The colour is specified as an index into a colour table in the workstation. Each workstation has 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 colour of the display surface after it has been cleared. Entry 1 is the default colour to display pictures and entries higher than 1 correspond to different colours. All entries may be redefined. Entries in the table are set by SET COLOUR REPRESENTATION which specifies the colour as combination of red, green and blue intensities. The specified colour is mapped to the nearest available by the workstation. The accuracy of the intensity of each colour component is set by SET COLOUR HEADER.
On monochrome workstations (workstations only capable of displaying colours with equal intensities of red, green and blue or displaying colours which are different intensities of the same colour), the intensity is computed from the colour values as follows :

intensity = 0,3 * red + 0,59 * green + 0,11 * blue

and this intensity is mapped to the nearest available intensity.

5.2 WORKSTATIONS

5.2.1 GRAPHICS WORKSTATIONS

This document is based on the concept of graphics workstations of GKS, which are abstractions of collections of physical devices. The concept of workstation allows to specify device independent applications that can, at the same time, take full advantage of the physical device capabilities. An abstract graphical workstation with maximum capabilities :

- has one addressable display surface of fixed resolution ;
- allows only rectangular display spaces, that cannot consist of a number of separate parts ;
- 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 ;
- supports several line types, text fonts, character sizes, etc.., in order to allow output primitives to be drawn with different attributes ;
- has one or more logical input devices for each input class and permits different input modes ;
- allows storage for short term of graphical information in segments and provides facilities for manipulating them.

5.2.2 WORKSTATION CHARACTERISTICS

Each workstation falls into one of the following categories :

- OUTPUT
  Output workstation, having a display surface for displaying output primitives (e. g. a plotter).

- INPUT
  Input workstation, having at least one input device (e. g. a digitizer, a keyboard).

- OUTIN
  Output/input workstation, having a display surface and at least one input device, also called an interactive graphical workstation.

- WISS
  Workstation independent segment storage.

A graphics configuration is comprised of one or more workstations, each of which pertains to a given category.

As an example, a graphics configuration made up of :

- a CRT and its associated keyboard,
- a printer,
- a diskette,

can be logically interfaced through the following workstations:

- an OUTIN workstation (CRT and keyboard),
- an OUTPUT workstation (printer),
- a workstation independent segment storage (diskette).

A combination of GKS and one or more physical devices might be regarded as a GKS instance, within such an instance one and only one WISS is allowed to exist.

5.2.3 Selecting a workstation

The workstations are identified by a workstation identifier. Connection to a particular workstation is established by the primitive OPEN WORKSTATION.

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 primitive ACTIVATE WORKSTATION.

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

The following sequence of primitives illustrates workstation selection :

```
OPEN WORKSTATION (N1);
OPEN WORKSTATION (N2);
ACTIVATE WORKSTATION (N1);

    Output primitives;      generated only on N1
    Attribute setting;      possible on N1, N2

ACTIVATE WORKSTATION (N2);

    Output primitives;      generated on N1, N2
```

```
DEACTIVATE WORKSTATION (N1) ;

    Output primitives;      generated only on N2
    Attribute setting;      possible on N1, N2

CLOSE WORKSTATION (N1);
DEACTIVATE WORKSTATION (N2);
CLOSE WORKSTATION (N2);
```

5.2.4 STATE VARIABLES

The set of the state variables is organized into four tables that encode the specific characteristics of a configuration and their evolution they are called GDS state list, workstation description table, workstation state list and segment state list.

The GDS state list gathers both static and dynamically updated information regarding :

a) Global configuration information (e.g. maximum number of simultaneously open workstations, etc..).

b) Current global values (e.g. set of open workstations, current line type, etc..).

c) Last error condition (e.g. error in parameter, etc...).

The GDS state list can be inquired by a non-mandatory primitive that is provided for basic debugging purposes.

The workstation description table gathers only static information concerning the workstation initial state for every single workstation that can be configured onto the physical device. The workstation description table can be inquired by a non-mandatory set of primitives.

The workstation state list is allocated when a configured workstation is effectively opened. It gathers information that change according to any graphical transaction that will be performed onto the workstation. The workstation state list can be inquired by a non-mandatory set of primitives.

The segment state list gathers global information concerning the segments stored in the workstations and can be inquired by a non-mandatory set of primitives.

5.3 COORDINATE SYSTEMS AND TRANSFORMATIONS

5.3.1 Coordinate systems

Output devices that are used for representing the visual image of the graphical elements normally require the use of a specific coordinate system. In order to maintain device independency, two coordinate systems have been defined.

- Normalized Device Coordinate (NDC)
  A coordinate specified in a device independent intermediate coordinate system, normalized to some range, including -7 to + 7, as GKS requires. All output primitives are defined in the NDC space. The coordinates used in constructing display images are expressed in this coordinate system. The actual coordinates are expressed in fractional units based on the Basic Grid Unit which is determined by the accuracy of the coordinate encoding.

- Device Coordinate (DC)

  A coordinate specified in the actual coordinate system of the workstation display space. The mapping from NDC to DC is done by the workstation itself. Every workstation may have a different device coordinate space, resulting in a different mapping.

  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.
  c) x and y increases to the right and upwards respectively.

5.3.2 WORKSTATION TRANSFORMATION

The normalized device coordinate space can be regarded as a workstation independent abstract viewing surface. Each workstation can select independently some part of the NDC space in the range [0.0, 1.0] X [0.0, 1.0] to be displayed somewhere on the workstation display surface. 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.

A workstation transformation is specified by defining the limits of an area in the normalized device coordinate system within the range [0.0, 1.0] x [0.0, 1.0] (WORKSTATION WINDOW) which is to be mapped onto a specified area of the device space (WORKSTATION VIEWPORT) defined in the device coordinate system (See Fig. 9).

Window and 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 window is displayed, the picture is clipped at the workstation window boundaries, and this clipping cannot be disabled.

If the workstation window and workstation viewport have different aspect ratios, the specified scaling would be different on each axes if the window was mapped onto the viewport in its entirety. To ensure equal scaling on each axes, the transformation maps the window onto the largest rectangle that can fit within the viewport such that (See Fig.10) :

- Aspect ratio is preserved.

- The lower left hand corner of the workstation window is mapped to the lower left hand corner of the workstation viewport.

The largest square which fits into the display area having its bottom and left sides coincident with the bottom and left sides of the rectangular display surface is seen as the default workstation viewport.

5.3.3 Clipping

Only those parts of the NDC space that lie within user definable rectangles can be shown. Cutting away parts outside such rectangles (CLIPPING RECTANGLE) is called clipping. Clipping takes place when the output primitives are displayed on the view surface of a workstation. Output primitives stored in segments will have the associated clipping rectangle stored with the primitives.

An example of clipping and workstation transformation is given in Fig. 11.

5.3.4 Coordinate specification

The coordinates may be specified in two possible modes :

- Displacement mode defining a displacement from the preceding point; For the first point of each point list, the displacement is a displacement from the origin.
- Incremental mode, defining steps (increments) from one coordinate position to another.

The GDS does not contain the concept of current position therefore, the primitives are logically independent. For each primitive that as a point list as parameter, the first coordinate point is in displacement mode, the displacement coded being relative to the origin of NDC space.

Displacement mode coordinates can be specified with a different number of bits. The parameters of the SET COORDINATE PRECISION primitive specify the number of bits of the displacement mode coordinates.

Incremental mode coordinates are based on a variable ring size and a variable number of points on the ring. The primitive SET DOMAIN RING has two parameters related to the Incremental mode :

- RING SIZE : determines the size of the ring.
- ANGULAR RESOLUTION FACTOR : determines the number of points on a ring.

A detailed description of the coordinate data encoding is given in clause 7.

5.4 SEGMENTS

5.4.1 Concept of segments

A picture is composed of output primitives. They may be grouped into parts that can be addressed and manipulated as a whole. These picture parts are called segments.

Segments are identified by a unique name called segment identifier.

A segment may be :

- transformed ;
- made visible or invisible ;
- highlighted or not ;
- assigned different priorities ;
- made detectable or undetectable ;

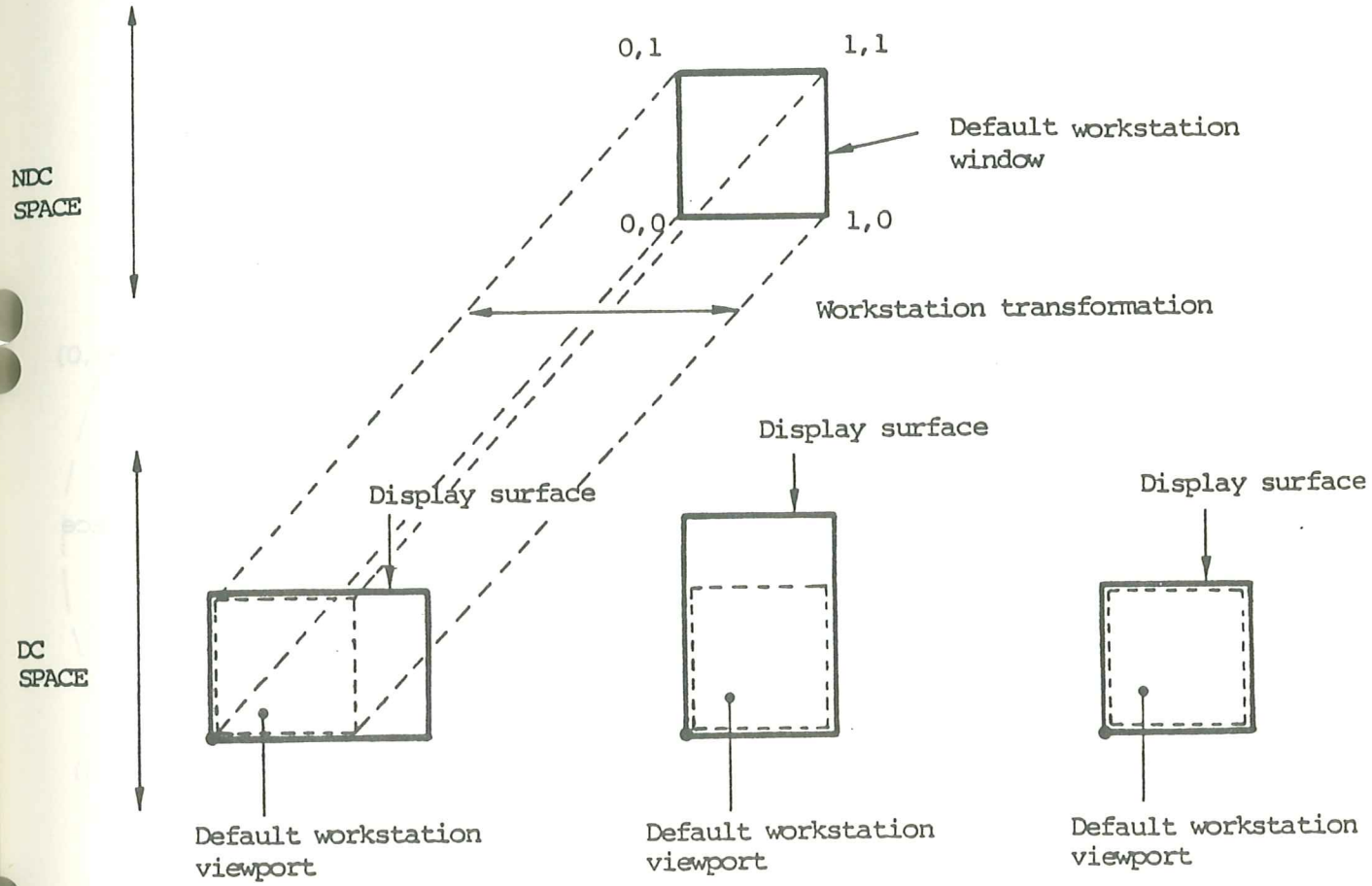


Fig. 9 - Default Workstation Transformation

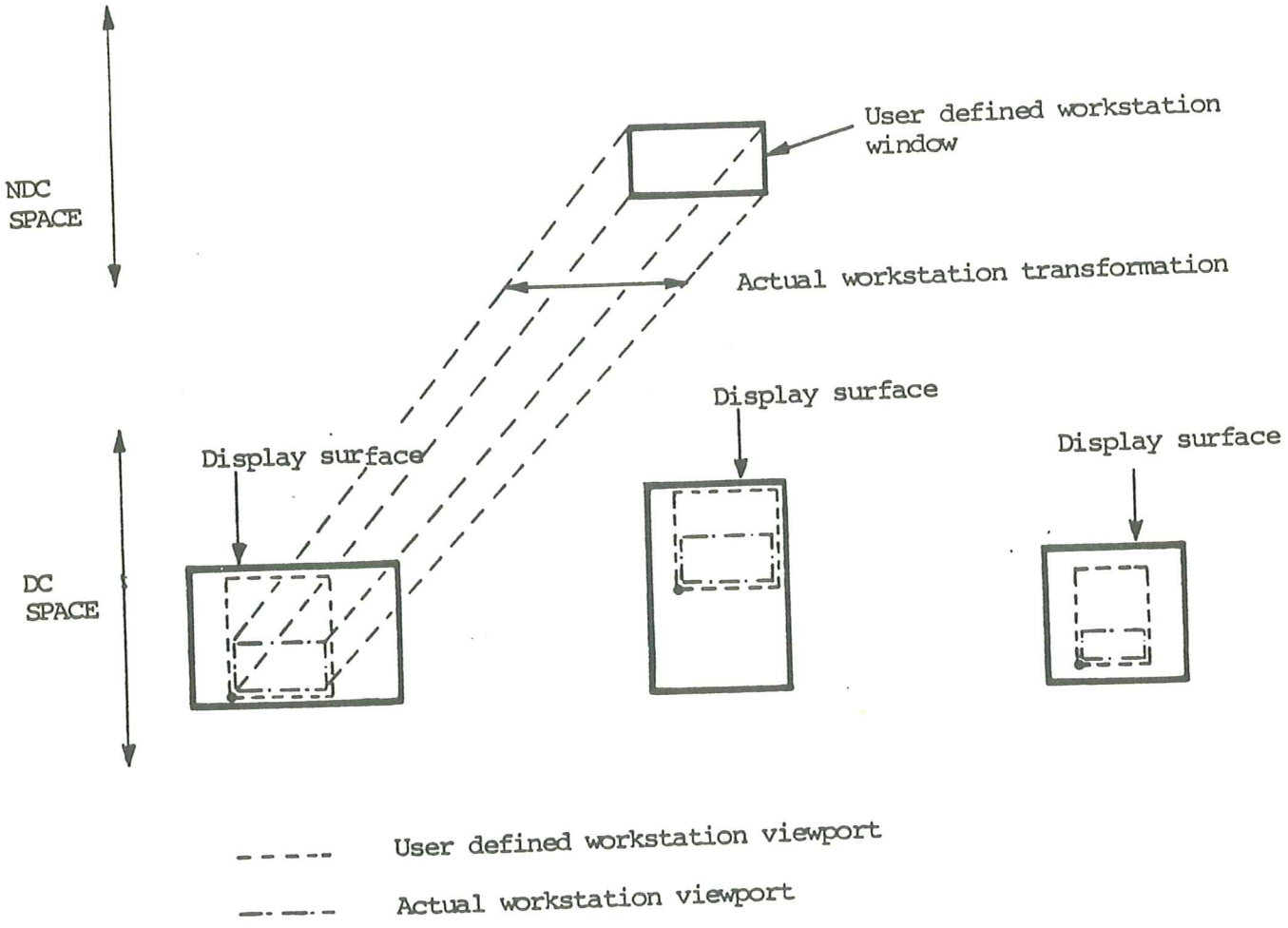


Fig. 10 Workstation transformation with anisotropic workstation window and workstation viewport.

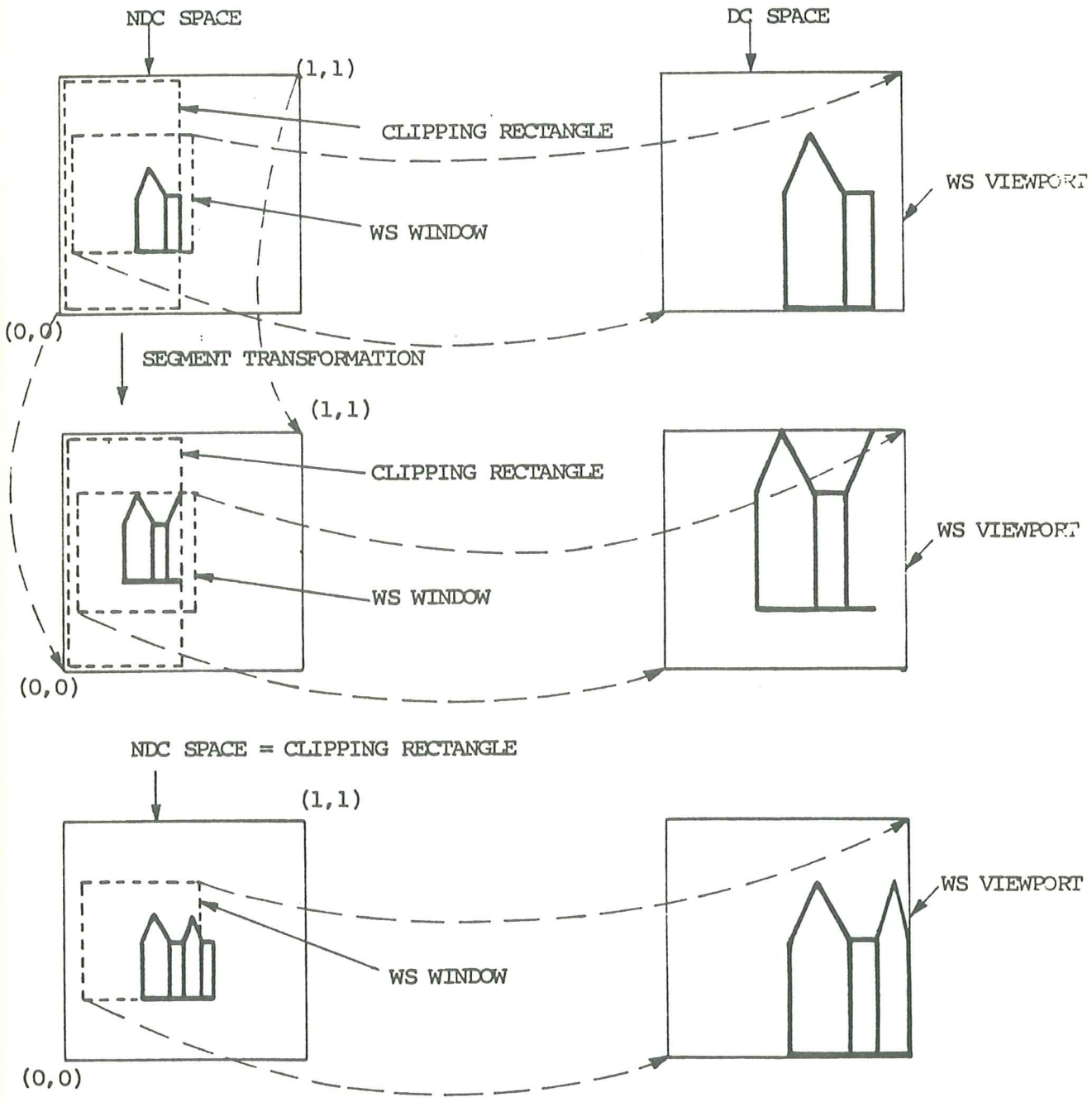


FiG. 11 Clipping and Workstation Transformation

- deleted ;
- renamed ;

- inserted into the open segment or into the stream of primitives outside segments.

Each segment is stored on all workstations active at the time the segment is created.

All output primitives are collected in a segment after it has been created and until it is closed. After a segment is closed, no primitives can be added to or deleted from the segment.

The output primitives within a segment can have an additional identification that needs not be unique, called pick identifier. It is part of the input value delivered by a pick input device when a segment is picked. The pick identifier has no meaning for workstations of category OUTPUT.

5.4.2 Segment attributes

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

- SEGMENT TRANSFORMATION
  A segment may be transformed by translation, rotation, scaling or a combination of them.

- VISIBILITY
  A segment is either displayed or not.

- HIGHLIGHTING
  A visible segment is either highlighted or not.

- SEGMENT PRIORITY
  If parts of segments (for example, FILL AREA, CELL ARRAY) overlap, the segment with the highest priority will be preferred, both when the segments are displayed and when they are picked ;

- 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 0.0, undetectable) are assigned to a segment when it is created. The segment attributes of any segment in existence, including the open segment, may be changed.

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.

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

5.4.3 Segment Transformations

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

Segment transformations are characterized by :

- segment name ;
- transformation matrix.

The transformation matrix is a 2 by 3 matrix consisting of a 2 by 2 scaling and rotation portion and a 2 by 1 translation portion.

The segment transformation takes place before any clipping.

A segment transformation, specified by the SET SEGMENT TRANSFORMATION primitive, 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 primitives 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.

Note that locator input data is not affected by any segment transformation.

5.4.4 Clipping and WDSS

Clipping takes place after the segment transformation has been applied. Each primitive is clipped against the clipping rectangle associated with the primitive when it was put into the segment.

Note that 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 coordinate axes.

5.4.5 Workstation Independent Segment Storage

A 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 primitives. None of these primitives modify the contents of the segments to which they are applied. Only one WISS is permitted, in a GKS-instance.

The ability to manipulate segments requires the storage of all segments when they are created so that they can be reused on whatever workstations are active when they are created. By contrast, primitives outside segments cannot be reused.

The treatment of the primitives COPY SEGMENT TO WORKSTATION, ASSOCIATE SEGMENT WITH WORKSTATION and INSERT SEGMENT is explained in figure 12.

5.4.6 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 the current clipping rectangle is associated with each primitive.

COPY SEGMENT TO WORKSTATION copies primitives from a segment in WISS to be output on the specified workstation. The primitive takes a copy of each primitive and its associated clipping rectangle from a segment in 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). This primitive cannot be invoked when a segment is open. By contrast with ASSOCIATE SEGMENT WITH WORKSTATION, this primitive 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 primitive 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 is 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 current clipping rectangle. Note that all primitives processed by a single invocation of INSERT SEGMENT receive the same clipping rectangle, and that 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 primitive can be used when a segment is open but the open segment cannot, itself, be inserted.

5.5 Deferring picture changes

The display of a workstation should reflect as far as possible the actual state of the picture as defined by the sending entity. However, to use efficiently the capabilities of a workstation, the requested action may be delayed for a certain period of time. During this period, the state of the display may be undefined.

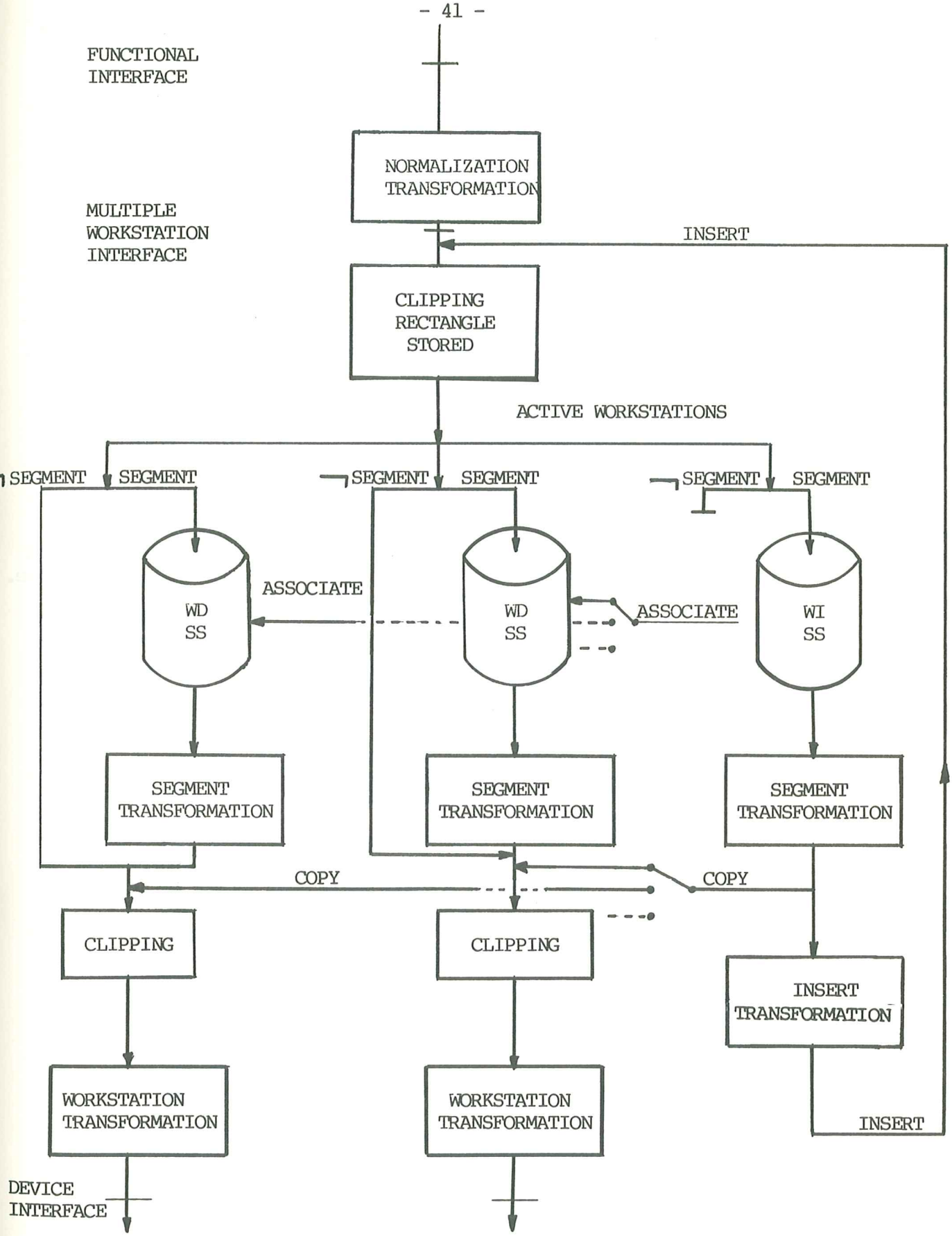


Fig. 12 The data flow chart for graphical output

A workstation state variable controlling if and how long such a delay of picture updates is allowed, is called the deferral state of the workstation. The SET DEFERRAL STATE primitive allows the sending entity to choose that deferral state which takes into account the capabilities of the workstation. Two attributes are defined for this purpose. Deferral mode controls the time at which output primitives have their visual effect. 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 the primitives. Effects on the segment or on the state of the workstation are not deferred.

Deferral mode controls the possible delaying of output primitives. The values of deferral mode (in increasing order of delay) are :

a) ASAP: The visual effect of each primitive will be achieved on the workstation As Soon As Possible (ASAP).

b) BNIG: The visual effect of each primitive 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. If an interaction on any workstation is already underway, the visual effect will be achieved as soon as possible. This value is meaningful in situations where all the workstations handled by an application through a specific occurrence of GKS are located in one graphics device.

c) BNIL : The visual effect of each primitive 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. 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 primitive will be achieved on the workstation At Some Time (ASTI).

Deferral applies to the following primitives that generate output :

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

Certain primitives can be performed immediately on some workstations, but on other workstations they 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 :

a) Lead to an implicit regeneration (IRG) ;

b) 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 primitive REDRAW ALL SEGMENTS. Its possible delay is controlled by the implicit regeneration mode. The mode can be specified as follows :

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

b) ALLOWED : implicit regeneration of the picture is allowed.

An implicit regeneration is made necessary if the primitives listed below have a visible effect on the display image of the respective workstation :

a) If the 'dynamic modification accepted' entry in the workstation description table is IRG (implicit regeneration necessary) for the specified representation :

   SET POLYLINE REPRESENTATION
   SET POLYMARKER REPRESENTATION
   SET TEXT REPRESENTATION
   SET FILL AREA REPRESENTATION
   SET PATTERN REPRESENTATION
   SET COLOUR REPRESENTATION

b) If the 'dynamic modification accepted' entry in the workstation description table is IRG for the workstation transformation :

   SET WORKSTATION WINDOW
   SET WORKSTATION VIEWPORT

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

1) If primitives are added to the open segment overlapping a segment of higher priority :

POLYLINE
POLYMARKER
TEXT
FILL AREA
CELL ARRAY
GENERALIZED DRAWING PRIMITIVES
INSERT SEGMENT

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

2) If the complete execution of one of the following primitives would be affected by segment priority :

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

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

SET SEGMENT TRANSFORMATION

e) If the 'dynamic modification accepted' entry in the workstation description table is IRG for 'visibility (visible --> invisible)':

SET VISIBILITY (INVISIBLE)

f) If the 'dynamic modification accepted' entry in the workstation description table is IRG for 'visibility (invisible --> visible)':

SET VISIBILITY (VISIBLE)

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

SET HIGHLIGHTLING

h) If the 'dynamic modification accepted' entry in the workstation description table is IRG for delete segment :

DELETE SEGMENT
DELETE SEGMENT FROM WORKSTATION

An implicit regeneration has to be done (including deletion of primitives outside segments) only if one of the primitives listed causes a visible effect on the display : for example, if an invisible segment is deleted, a regeneration does not need to 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 primitive or by an appropriate change of the deferral state.

5.6 GRAPHICAL INPUT

5.6.1 Logical input devices

The graphical information that is input from a physical device, as a result of operator actions, can be obtained by controlling the activity of one or more logical input devices that send logical input values to the peer entity.

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

The six input classes and the logical input values are :

- LOCATOR provides a point.

- STROKE provides a sequence of points.

- VALUATOR provides a real number.

- CHOICE provides a non negative integer representing a selection from a number of choices.

- PICK provides a PICK status, a segment name and a PICK identifier.

- STRING provides a character string.

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 the primitive SET LOGICAL INPUT DEVICE <input class> MODE.

The three operating modes are REQUEST, SAMPLE and EVENT :

- REQUEST A program will REQUEST an input from a logical input device and will then wait until the logical input device takes over or a break action is performed by the operator.
  The break concept is consistent with the corresponding GKS concept and, as suggested by the current state of the art is implementation-dependent.

- EVENT Any input values in the input queue containing temporally ordered event reports is returned.
An event report contains the identification of a logical input device and a logical input value from that device and is generated asynchronously by operator action.
For a specified logical input device the oldest event report can be removed from the queue and its content examined and also all report events can be flushed from the queue.

- SAMPLE The current logical input value from a specified logical input provided.

The basic difference among these modes is that in the first case the logical input device takes part in an interaction synchronously while in the others two cases it takes part asynchronously.

In fact the invocation REQUEST LOGICAL INPUT DEVICE <input class> causes the primitive to be served only when the input is entered, while the invocation AWAIT EVENT <time out> or SAMPLE LOGICAL INPUT DEVICE <input class> causes the primitive to be served even if the input data is not available (AWAIT EVENT).

5.6.2 Logical input device model

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 ans echo type.

The measure of a logical input device is a value determined by one or more physical input devices together with a "measure mapping", where a specific measure mapping can be applied to each measure.

A physical input device or a set of the represent the "trigger" of a logical input device and indicate significant moments in time for the operator.
At this moment the trigger is said to "fire".

The prompt and echo type may be selected by calling the appropriate INITIALIZE function.
Also the initial value and the data record are defined through this primitive.

The logical input device model allows a synchronous mode way for input handling corresponding to the REQUEST mode in the SET LOGICAL INPUT DEVICE <input class> primitive where the input request is served only when the trigger fires or a break occurs, and echoing, when enabled, is performed only within this interval of time.

When an input request is issued, the initial value is checked according to the input class rules ; if correct, it is used as current measure ; otherwise a logical input device dependent value is used. A prompt is then ouput to indicate that the device is ready for use.

The second way for input handling consists of an asynchonous mode (corresponding to the EVENT or SAMPLE mode in the SET LOGICAL INPUT DEVICE) <input class> primitive.

In this mode the measure can be seen as the state of an independent, active process (the measure process) whose current state corresponds exactly with a logical input value.

Whenever a device is taking part in an interaction, the measure process is in existence. Under others conditions, this process doesn't exist.

When a measure process comes into existence the initial value is checked according to the rules of that input class.
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 trigger can be seen as an independent active process that sends a message to one or more recipients when it fires. The trigger process exists if there is at least one recipient for a trigger. Under other conditions this process does not exist.

In EVENT mode when the trigger of one or more devices (with the same trigger) fires, the identifications of those devices and their measure values are passed to the input queue mechanism as separate event reports and may be obtained in sequence by invoking AWAIT EVENT, and then invoking the appropriate GET <device class> primitive.

In SAMPLE mode the logical input value is obtained by calling the appropriate SAMPLE <input class> primitive.

This value is set to the current state of the measure process without waiting for a trigger firing.

A logical Input Device can change its mode by the SET LOGICAL INPUT DEVICE <input class> MODE function.

After an invocation of this primitive with the parameter REQUEST no measure process exists.

After an invocation with the parameter EVENT or SAMPLE the measure process comes into existence but while in the first case the device's identifier is on its trigger's list of recipients, this is not true in the second case.

5.6.3 Measures of each input class

A LOCATOR measure consists of a point P, expressed in NDC coordinates lying within the workstation viewport.

A STROKE measure consists of a sequence of points Pi ($1 \leq i \leq m$) in NDC coordinates lying within the workstation viewport.

A VALUATOR measure provides real numbers. Each value lies between (possibility including) minimum and maximum values defined in the data record specified by the INTIALIZE LOGICAL INPUT DEVICE <input class> primitive and stored in the workstation state list.

A CHOICE measure provides integers in the range 0 to a device dependent maximum specified in the workstation description table. The value 0 is conventionally interpreted as "NOCHOICE".

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

- The segment exists and has VISIBILITY and DETECTABILITY on.
- The segment is present on the workstation containing the PICK device.
- The pick identifier refers to, at least, one output primitive in the segment. This is tested using the clipping parameters 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 clipping rectangle. Further, the primitive is not completely overlapped by primitives in a segment of higher priority.

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

A STRING measure consists of a character string up to a device dependent maximum length specified by the buffer size value stored in the data record contained in the workstation state list.

5.6.4 Input queue and current event report

The input queue contains zero or more event report consisting of a device identifier and a logical input value as result of trigger firings and a data specifying an ordering information.

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.

For implementation purposes it would be useful to implement a queue for each device, however from the logical point of view, only a single queue is provided for all the devices.

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 that have been occured.

If there is no room in the queue, a input queue overflow occurs.

This is notified through the corresponding response primitive at the subsequent invocation of the AWAIT EVENT <input class> primitive.

The input queue has to be emptied before any further event is generated and therefore any event report will be added.

AWAIT EVENT, if the queue is not empty, removes the first event report after copying the logical input value into the current event report.

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 has elapsed whichever occurs first.

The contents of the current event report can be obtained by calling the appropriate GET LOGICAL INPUT DEVICE <input class> primitive.

In order to remove all event reports from the input queue, FLUSH DEVICE EVENTS for a specific device and CLOSE WORKSTATION for all logical input devices on that workstation can be called.

5.6.5 Initialization of input devices

If the logical input devices is in REQUEST mode an INITIALIZE function is available for each input class.

This primitive provides the the following information stored within the wokstation state list :

a) An initial value that must be consistent with the input class rules.

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 is required for all logical input devices while further prompt and echo types appropriate to each class are defined but are not required.

c) An echo area (left, right, bottom, top) specified by two points within the display surface defining a rectangular area the side of which are parallel to the axes, expressed in device coordinates.

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 a well defined place in the data record : values mandatory to the input class appear first, followed by the values mandatory to prompt and echo type.

The data record may contain additional information.

The measure of every logical input device set to any mode takes an initial value from the workstation state list, unless this is not valid for the device.
If the measure is not valid a device dependent value is taken except for PICK for which NOPICK value is taken.

As far as data record is concerned the items mandatory for each class are : in STROKE case buffer size in number of points ; in VALUATOR case low value and high value ; in STRING case input buffer size and initial cursor.

Prompt and echo types which have mandatory values are types 2, 3, 4 and 5 for CHOICE ; the corresponding prompt and echo technique depend on the device input class.

5.7 INQUIRY

The internal state of a graphics configuration is represented by means of a set of state variables organized in four tables (see 5.2.4) which can be either static or dynamic in the sense that their value change accordingly to the graphical primitives execution.
The whole set is made available by an "inquiry" mechanism realized by a specific set of primitives that do not affect the state itself.

5.8 ERROR DETECTION

All the different situations that can originate an internal error condition are noticed by means of the ERROR DETECTED (response) primitive. This notification mechanism is completely synchronous, that conceptually means that every primitive is equivalent to the issue of the (non-existing) ERROR DETECTED (request) primitive.

The error condition state is reset by issuing the ERROR DETECTED (response) primitive in all the cases except when an overflow occurs during an input operation performed in event mode. In this case the error condition is reset when the input queue is emptied.

5.9 ERROR HANDLING

Each primitive will appear in a specified format. The possible forms of each primitive are defined in this document. The following rules will apply if a workstation detects an error in the received format :

- If a parameter value is not defined or not in the range of allowed values, the default value will be used (e.g. an accuray value of 3 will default to a value of 1).

- If a primitive is received that is not supported, the primitive is ignored.

- A primitive that is not defined will be ignored.

- A primitive with parameters not encoded according to this document will be ignored.

- If an error in a point list occurs the processing of the primitive will stop after the previously received correct points have been processed.

As a result of the encoding technique the skipping of erroneous primitives is always possible.

5.10 LEVELS

The functional capabilities are 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).

Nine levels are defined in order to allow the implementation of several categories of graphics equipment.

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

The output level axis has the three possibilities :

0 : Minimimal output ;

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.

Capabilities are expressed by primitives and by ranges of parameters.

There are three different types of capability at each level :

- An explicitly defined and required capability. Every graphics equipment in conformance with a specific level supports the capability at that level.

- An explicitly defined and non-required capability. A graphics equipment may support the capability and, if it does, it is implemented according to the explicit primitive definitions.

- A conceptually defined and non-required capability. A graphics equipment may provide the capability. Its implementation follows general rules given by the concepts (See 5.2) and functional definitions.

The set of explicitly defined and required capabilities includes :

a) predefined bundle numbers up to the required minimum ;

b) line types : solid, dashed, dotted and dash-dotted ;

c) marker types : dot, plus sign, asterisk, circle and diagonal cross ;

d) text precision STROKE (output levels 1 and 2) ;

e) interior style HOLLOW ;

f) one input device for each input class (input levels b and c) ;

g) prompt and echo type 1 (input levels b and c).

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

a) text precision STROKE (output level 0) ;

b) interior style SOLID, PATTERN, HATCH ;

c) transformable patterns ;

d) segment priority (output levels 1 and 2) ;

e) prompt and echo types (input levels b and c).

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

a) line types other than those explicitly defined ;

b) marker types other than those explicitly defined ;

c) specific generalized drawing primitives ;

d) prompt and echo types different from the defined set (input levels b and c) ;

e) specific escape primitives.

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

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

Output level 0 : minimal output

a) basic control ;

b) all primitives available at least in minimal performance ;

c) use of predefined bundles only (no modification to bundles) ;

d) colour representation modification possible ;

e) only one workstation with output capabilities available at a time ;

f) suitable basic inquiries ;

g) 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) basic segmentation (no Workstation Independent Segment Storage) ;

g) 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 primitives ;

b) REQUEST primitives on all appropriate devices ;

c) appropriate logical input device include PICK if and only if combined with output level 1 capabilities.

Input level c : full input.

a) all input level b capabilities ;

b) SAMPLE and EVENT mode input.

Table 3 gives a short overview of the functionality of each level. Each box contains only those functions added to the previous boxes of the same row and column.

Output Level	Input level a	b	c
0	No input, minimal control, only predefined bundles and all output primitives	REQUEST input, mode setting and initialize primitives for logical input devices, no PICK.	SAMPLE and EVENT input, no PICK.
1	Full output including full bundle concept, multiple workstation concept, basic segmentation (everything except Workstation Independent Segment Storage).	REQUEST PICK, mode setting and initialize for PICK.	SAMPLE and EVENT input for PICK.
2	Workstation Independent Segment Storage		

Table 3 Level Concept

Embedded in the levels summarized above are variations in the number of possibilities required in the set of explicitly defined and required capabilities. Table 4 exactly identifies the minimum support which is always provided at each level.

CAPABILITY	Level 0a	0b	0c	1a	1b	1c	2a	2b	2c
Colours (Intensity)	1	1	1	1	1	1	1	1	1
Line types	4	4	4	4	4	4	4	4	4
Line widths	1	1	1	1	1	1	1	1	1
Predefined polyline bundles (6)	5	5	5	5	5	5	5	5	5
Settable polyline bundles	-	-	-	20	20	20	20	20	20
Marker types	5	5	5	5	5	5	5	5	5
Marker sizes	1	1	1	1	1	1	1	1	1
Predefined polymarker bundles (6)	5	5	5	5	5	5	5	5	5
Settable polymarker bundles	-	-	-	20	20	20	20	20	20
Character vectors (see Note 1)	1	1	1	1	1	1	1	1	1
Character expansion factor (see Note 1)	1	1	1	1	1	1	1	1	1
String precision fonts	1	1	1	1	1	1	1	1	1
Character precision fonts	1	1	1	1	1	1	1	1	1
Stroke precision fonts	0	0	0	2	2	2	2	2	2
Predefined text bundles (6)	2	2	2	6	6	6	6	6	6
Settable text bundles	-	-	-	20	20	20	20	20	20
Predefined patterns (see Note 2,6)	1	1	1	1	1	1	1	1	1
Settable patterns (see Note 2 and 5).	-	-	-	10	10	10	10	10	10
Hatch styles (see Note 3)	3	3	3	3	3	3	3	3	3
Predefined fill area bundles (6)	5	5	5	5	5	5	5	5	5
Settable fill area bundles	-	-	-	10	10	10	10	10	10
Segment priorities (see Note 4)	-	-	-	2	2	2	2	2	2
Input classes	-	5	5	-	6	6	-	6	6
Length of input queue (see Note 5).	-	-	20	-	-	20	-	-	20
Maximum string buffer size (characters)	-	72	72	-	72	72	-	72	72
Maximum stroke buffer size (points)	-	64	64	-	64	64	-	64	64
Workstations of category OUTPUT or OUTIN	1	1	1	1	1	1	1	1	1
Workstations of category INPUT or OUTIN	-	1	1	-	1	1	-	1	1
Workstation Independent Segment Storage	-	-	-	-	-	-	1	1	1

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

Table 4 - Minimal Support Required at Each Level.

Note 1

Relevant only for character and string precision text ;

Note 2

Relevant only for workstation supporting pattern interior style ;

Note 3

Relevant only for workstation supporting hatch interior style ;

Note 4

Relevant only for workstation supporting segment priorities ;

Note 5

Since available resources are finite and entries have variable size, it may not always be possible to achieve the minimal values in a particular physical device.

Note 6

For each bundle table, the predefined entries at a given level must be distingishable from each other.

6. DESCRIPTION OF THE PRIMITIVES

6.1 INTRODUCTION

The primitives are discussed in this clause.

Each primitive is named, the parameters are described, data types are listed, and a description of implicit relationship is added.

The order in which parameters will occur in a parameter list is not to be assumed from the order in which they are mentioned in this chapter but is deferred to clause 8.

The list of data types is given below :

PARAMETER DATA TYPES		MEANING
P	Point	Two NDC normalized device coordinate values representing the x and y coordinates of a point in NDC space ;
CI	Colour Index	Index into a table of colour values ;
CL	Colour Index List	List of colour indexes encoded in different ways ;
CD	Colour direct	Colour definition with R, G, B intensities ;
E	Enumerated type	Set of standardized values. The set is defined by enumerating the identifiers that denote the values ;
I	Integer	Number with no fractionnal part other than NDC integer values ;
ID	Identifier	Name or identifier ;
IX	Index	Pointer into a table of values other than colour indices ;
M	Matrix	Segment transformation matrix ;
REC	Record	Data Record ;
S	String	Sequence of characters ;
V	Size value	Size value in NDC space ;
R	Real	Real number.

Combinations of simple types can also be used where n is an unspecified number (for example : nxP or 2xR, 2xI). Also, lists of types can be expressed (for example : I, E, R, E).

6.2 GEOMETRIC PRIMITIVES

6.2.1 WORKSTATION MANAGEMENT PRIMITIVES

6.2.1.1 OPEN WORKSTATION

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The specified workstation becomes "open". The workstation state list is allocated and initialized for the specified workstation. The workstation identifier is added to the set of open workstations in the GDS state list. Open workstation that the display surface is clear, but does not clear the surface needlessly.

Related primitives :

CLOSE WORKSTATION

Discussion :

In the following cases this primitive has no effect :
- The specified workstation is already opened ;
- The specified workstation is active ;
- The specified workstation identifier is invalid.

6.2.1.2 CLOSE WORKSTATION

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The specified workstation becomes "closed". 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 associated workstations in the segment state list of every segment containing it. If the set of associated workstations of a segment becomes empty, the segment is deleted. The input queue is flushed of all events from all devices on the workstation being closed. The display surface need not be cleared when CLOSE WORKSTATION is invoked, but it may be cleared.

Related primitives :

OPEN WORKSTATION

Discussion :

In the following cases this primitive has no effect ;
- The specified workstation is closed ;
- The specified workstation identifier is invalid ;
- The specified workstation is active.

6.2.1.3 ACTIVATE WORKSTATION

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The specified workstation is marked active in the workstation state list. The workstation identifier is added to the set of active workstations in the GDS state list.

Related primitives :

OPEN WORKSTATION
CLOSE WORKSTATION
DEACTIVATE WORKSTATION

Discussion :

In the following cases this primitive has no effect :
- The specified workstation is activated ;
- The specified workstation is closed ;
- The specified workstation identifier is invalid.

6.2.1.4 DEACTIVATE WORKSTATION

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The specified workstation is marked inactive in the workstation state list. The workstation identifier is deleted from the set of active workstations in the GDS state list.

Related primitives :

OPEN WORKSTATION
CLOSE WORKSTATION
ACTIVATE WORKSTATION

Discussion :

While a workstation is inactive, it will not process output primitives nor does it store new segments. Segments already stored on this workstation are retained.

In the following cases this primitive has no effects :
- The specified workstation is closed.
- The specified workstation identifier is invalid.

6.2.1.5 CLEAR WORKSTATION

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The effect of this primitive depends on the workstation category :

1. OUTPUT, OUTIN and WISS workstations :

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 always cleared.

c) The current workstation transformation is set to the last defined WORKSTATION WINDOW and WORKSTATION VIEWPORT, if necessary.

d) All segments stored for the specified workstation are deleted.

2. Other workstations : No effect

Related primitives : None

Discussion : None

6.2.1.6 SET DEFAULTS

Level : 0a

Parameters : None

Description :

This primitive places each workstation status or attributes to its default, as defined in clause 9.

Related primitives : None

Discussion : None

6.2.1.7 GRAPHICS ESCAPE PRIMITIVES

The graphics ESCAPE primitives allow use of device capabilities not specified by this standard, they are best suited for access to non-standardized control features of graphics devices.

Two different primitives are provided : GDS ESCAPE 2 and GDS ESCAPE 1, they use depends on whether a response from the graphics device is requested or not.

Some GDS ESCAPE functions, addressing capabilities generally found on certain type of devices (e.g. alphanumeric functions of CRT terminals) may be standardized in the future.

6.2.1.7.1 GDS ESCAPE 1

Level : 0a

Parameters :

GDS Escape identifier	(I)
GDS Escape data record	(REC)

Description :

The specific function specified by the GDS escape identifier is invoked. Non negative values of the GDS escape identifier are reserved for future standardization, negative values are available for implementation dependent use.

The GDS escape data record depends on the function being performed.

Related primitives : None

Discussion : None

6.2.1.7.2 GDS ESCAPE 2 (REQUEST)

Level : 0a

Parameters :

GDS Escape identifier	(I)
GDS Escape data record	(REC)

Description :

The specific function specified by the GDS escape identifier is invoked ; a response from the graphics device is requested.

Non negative values of the escape identifier are reserved for future standardization, negative values are available for implementation dependent use.

The GDS escape data record is dependent on the function being performed.

Related primitives :

GDS ESCAPE 2 (RESPONSE)

Discussion : None

6.2.1.7.3 GDS ESCAPE 2 (RESPONSE)

Level : 0a

Parameters :

GDS Escape identifier	(I)
GDS Escape data record	(REC)

Description :

A GDS escape data record depending on the GDS escape identifier is returned.

Related primitives :

GDS ESCAPE 2 (REQUEST)

Discussion : None

6.2.1.8 MESSAGE

Level : 1a

Parameters :

Workstation identifier	(ID)
Message	(S)

Description :

The specified workstation may display the message at an implementation dependent location on the workstation viewport or on some separate device associated with the workstation.

Related primitives : None

Discussion : None

6.2.1.9 INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (REQUEST)

Level : 0a

Parameters : None

Description :

This primitive is intended to gather the identification of the connected graphics equipment, its capabilities and the list of its workstations.

Related primitives :

INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (RESPONSE)

Discussion : None

6.2.1.10 **INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (RESPONSE)**

Level : 0a

Parameters :

Graphics configuration name : (ID)
Number of workstations (n) (I)
List of workstation identifications nx(ID, E1, E2)
where E1 specifies the category of the workstation and E2 the supported level in the specified category.

Description :

A graphics configuration identification is composed of :

- the graphics configuration name ;
- the number of available workstations ;
- a list of workstation identifications ;
  a workstation identification comprises three elements :

  . the workstation name ;

  . the category of the workstation ;
  each workstation pertains to one of the following categories :

    - Output
    - Input
    - Output/Input
    - Workstation independant segment storage

  . the level supported by that particular workstation, i.e. the set of primitives which can be processed.

This primitive is the response to the INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (REQUEST) primitive.

Related primitives :

INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (REQUEST)

Discussion :

The ID specifies the terminal configuration.
The primitive INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (REQUEST/RESPONSE) allows only the description of a static configuration of a graphics configuration.

6.2.2 OUTPUT WORKSTATION PRIMITIVES

6.2.2.1 Output Drawing Primitives

6.2.2.1.1 POLYLINE

Level : 0a

Parameters :

Point list (2..n) (nxP)

Description :

A line is drawn from the starting point to the second point,..., from the next-to-last point to the ending point.

Related primitives :

SET POLYLINE INDEX
SET POLYLINE ASPECT SOURCE FLAGS
SET LINE TYPE SCALE FACTOR
SET LINE WIDTH
SET POLYLINE COLOUR INDEX

Discussion :

The interpretation of a zero-length line segment is implementation dependent.
If only one point is specified, the primitive is ignored.

6.2.2.1.2 POLYMARKER

Level : 0a

Parameters :

Point list (nxP)

Description :

The marker corresponding to the currently selected marker type is drawn at each of the points in the point list. If the marker type is one of the pre-defined markers, it is drawn centred at each of the points. Other implementation dependent markers may have other alignments where desired. If the resulting marker is completely within the clipping area, the entire marker is drawn. If any part of the marker would have to be executed outside the clipping rectangle the result is device dependent.

Related primitives :

SET POLYMARKER INDEX
SET POLYMARKER ASPECT SOURCE FLAGS
SET MARKER TYPE
SET MARKER SIZE SCALE FACTOR
SET POLYMARKER COLOUR INDEX

Discussion : None

6.2.2.1.3 FILL AREA

Level : 0a

Parameters :

Point list (3..n) (nxP)

Description :

A boundary of a polygonal region is defined by connecting each vertex to its successor in the ordered point list and connecting the last vertex to the first. The polygonal region may be non-simple. For example, edges are allowed to cross. In this way, subareas can be created. Any given point is considered inside the polygon if a straight line from the given point to infinity intersects the polygon edges an odd number of times. If this line passes through a vertex point tangentialy, the intersection count is not changed. If a polygon is clipped and subareas are generated, the new boundaries become part of the polygon boundaries.

A non-degenerate polygon (one with three or more vertices, not all of which are colinear) is displayed with interior as specified by the SET FILL AREA INDEX, SET ASPECT SOURCE FLAGS and SET INTERIOR STYLE primitives. The appearance of the boundary is implementation dependent.

Related primitives :

SET FILL AREA INDEX
SET ASPECT SOURCE FLAGS
SET FILL AREA INTERIOR STYLE
SET FILL AREA STYLE INDEX
SET PATTERN REPRESENTATION
SET PATTERN VECTORS
SET PATTERN REFERENCE POINT
SET FILL AREA COLOUR INDEX

Discussion

If the list contains only colinear vertices a straight line is drawn through them. If all specified points coincide, a dot is displayed. If less than three points are specified, the primitive is ignored.

6.2.2.1.4 **TEXT**

Level : 0a

Parameters :

Text position	(P)
String	(S)

Description :

The character codes specified in the string are interpreted to obtain the associated symbols. Characters are displayed on the view surface as specified by the text attributes.

The characters are dimensionned according to the SET CHARACTER VECTORS, font-dependent character aspect ratio, and SET CHARACTER EXPANSION FACTOR and are oriented according to SET CHARACTER VECTORS. The direction of the character placement in the string relative to SET CHARACTER VECTORS is controlled by SET TEXT PATH.

Related primitives :

SET TEXT FONT AND PRECISION
SET TEXT INDEX
SET SOURCE FLAGS
SET CHARACTER EXPANSION FACTOR
SET CHARACTER SPACING
SET TEXT COLOUR INDEX
SET CHARACTER VECTORS
SET TEXT PATH
SET TEXT ALIGNMENT

Discussion : None

6.2.2.1.5 **CELL ARRAY**

Level : 0a

Parameters :

Parallelogram (P, Q, R)	(3xP)
Number of cells in P - R direction m	(I)
Number of cells in Q - R direction n	(I)
Cell colour index list	(CL)

Description :

The points P, Q and R define a parallelogram. P and Q are the end points of a diagonal and R is the third corner.

This area is subdivided into m x n contiguous parallelograms of dimension $|xR - xP|/m$ by $|(yQ - yR)|/n$. Colour values are mapped to a list of m x n elements identifying the colour of each of the m x n parallelograms.

The cell colour index list consists of m * n colour indices, conceptually an array of dimensions m and n representing respectivelly the column and row dimensions.

Array element (1,1) is mapped to the cell at corner P and array element (m,1) is mapped to the cell at corner R. Array element (m,n) is mapped to the cell at corner Q. Hence, the colour elements are mapped within rows running from P to R and with rows incrementing in order from R to Q. This is illustrated in Figure 13. No current colour setting is changed.

Related primitives : None

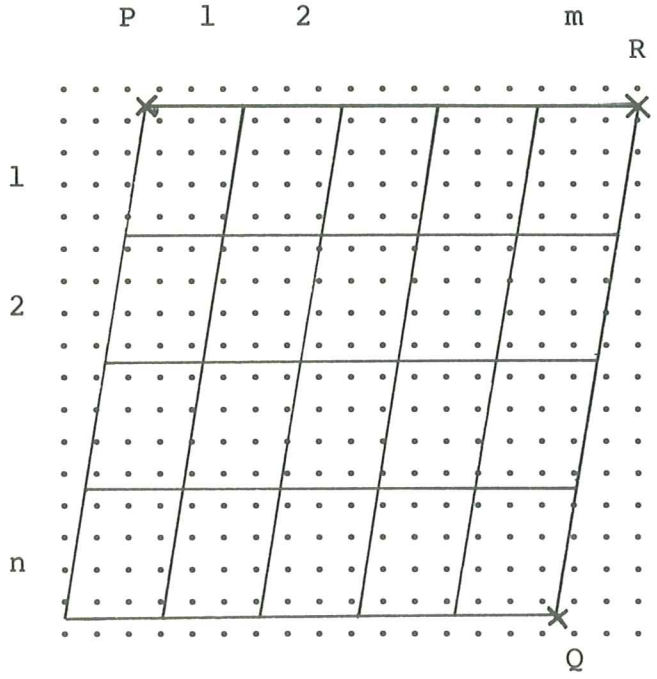


Figure 13 : m by n parallelograms mapped onto display surface. The dots indicate pixel locations.

Discussion :

The parameter and actions of this primitive supersede any setting of any other primitive. Specifically, individual or bundled colour specifiers are ignored.

If the number of indices in the colour index list is less than m x n, the last index is repeated until the end of the list. If the number of indices in the colour index list is greater than m x n, then the exceeding indexes are omitted.

If the number of cells parameter m and n are equal 0 the primitive is ignored.

6.2.2.1.6 GENERALIZED DRAWING PRIMITIVE (GDP)

Level : 0a

Parameters :

Identifier	(I)
Point list	(nxP)
Data record	(REC)

Description :

A GDP is specified by the identifier, the data of the point list and the data record. The appearance of the GDP is determined by zero or more of the attribute sets of the standardized output primitives, depending on the particular GDP. These attributes are listed in 5.1.2.

Non negative values of the identifier are reserved for standardization and negative values are available for private use.

Related primitives :

Not standardized

Discussion :

The GDP is displayed on all active workstations capable of doing so.

The points specified as parameters are transformed after the interpretation of the points 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. The resulting output of the GDP is clipped against the clipping rectangle.

6.2.2.1.6.1 GDP RECTANGLE

Level : 0a

Parameters :

GDP identifier (= RECTANGLE)	(I)
Point List	(2xP)

Description :

A rectangle is specified by a pair of points. These points are the opposite corners of the rectangle having its sides parallel to the axes.

The rectangle displayed with interior style as defined by SET FILL AREA INDEX, SET ASPECT SOURCE FLAGS and SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as for FILL AREA.

Discussion :

If the two points define a line segment parallel to one axis, a line through the two points is displayed.
If the two points are identical, a dot is displayed.

6.2.2.1.6.2 GDP CIRCLE

Level : 0a

Parameters :

GDP identifier (= CIRCLE)	(I)
Centre	(P)
Radius	(V)

Description :

A circle of the specified radius at the specified centre position is displayed with interior style as defined by SET FILL AREA INDEX, SET ASPECT SOURCE FLAGS and SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as for FILL AREA primitive.

Discussion :

If the radius parameter has the value 0, a dot is displayed at the centre position.
If the radius parameter is negative, the primitive is ignored.

6.2.2.1.6.3 GDP CIRCULAR ARC 3 POINT

Level : 0a

Parameters :

GDP identifier (= CIRCULAR ARC 3 POINT)	(I)
Starting point, intermediate point, ending point	(3xP)

Description :

A circular arc is displayed from the starting point, through the specified intermediate point, to the specified ending point.

Related primitives :

Same as for POLYLINE.

Discussion :

If the starting point and ending point are identical, a circle is displayed which passes through the specified points and has a diameter equal to the distance from the starting point to the intermediate point. The resulting circle is not considered to be a closed area, so it does not have an interior style.

If the intermediate point coincides with the starting or ending point, a straight line between the two points is displayed.

If the three points coincide a dot is displayed.

If the three points are colinear, a straight line from the starting point through the intermediate point to the ending point is drawn.

6.2.2.1.6.4 GDP CIRCULAR ARC 3 POINT CHORD

Level : 0a

Parameters :

GDP identifier (= CIRCULAR ARC 3 POINT CHORD) (I)
Starting point, intermediate point, ending point (3xP)

Description :

A circular arc 3 point chord defined by three points (starting point, intermediate point and ending point) is a boundary closed by the arc specified by the three points and the chord from the starting point to the ending point.

Such a circular arc 3 point chord is displayed with an interior style as defined by SET FILL AREA INDEX, SET ASPECT SOURCE FLAGS and SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as for FILL AREA.

Discussion :

If the starting point and ending point of an arc chord are identical, a filled area bounded by the circle which passes through the specified points and has a diameter equal to the distance from the starting point to the intermediate point is displayed.

If the intermediate point coincides with the starting or ending point, a straight line between the two points is displayed.

If three points coincide a dot is displayed.

If the three points are colinear, a straight line from the starting point through the intermediate point to the ending point is drawn.

6.2.2.1.6.5 GDP CIRCULAR ARC 3 POINT PIE

Level : 0a

Parameters :

GDP identifier (= CIRCULAR ARC 3 POINT PIE) (I)
Starting point, intermediate point, ending point (3xP)

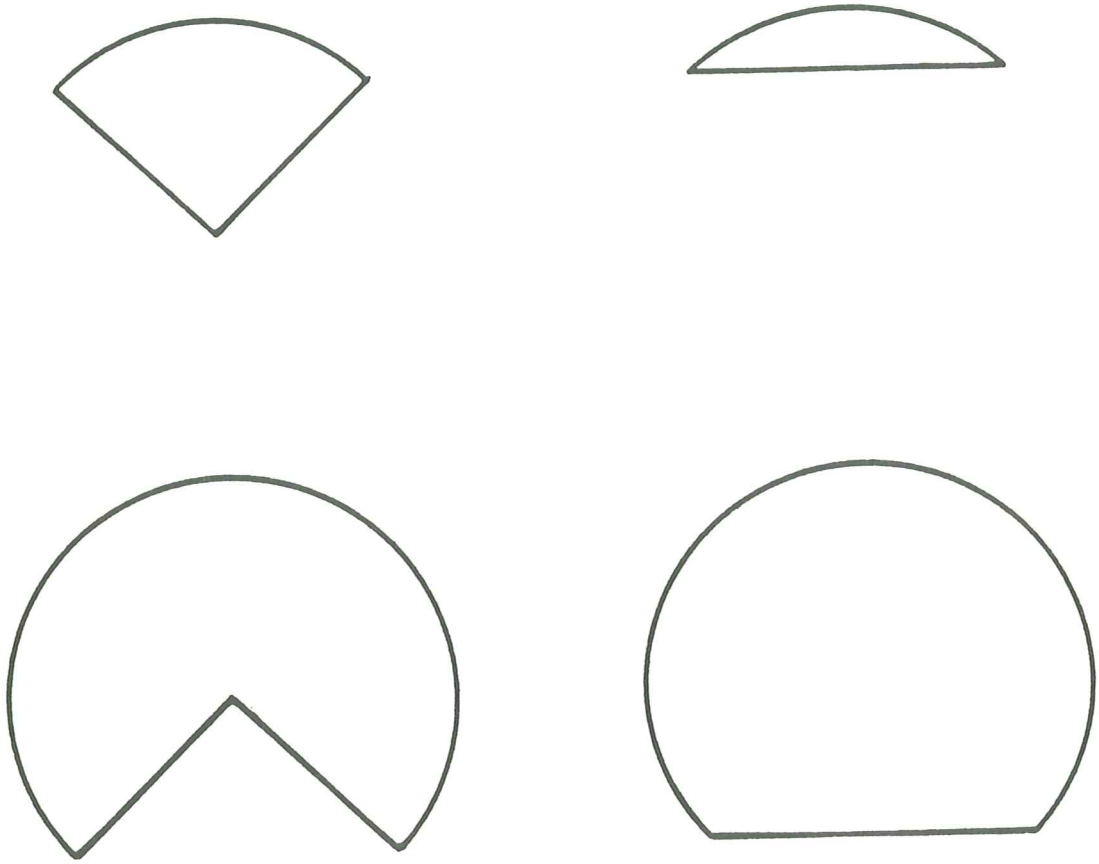


Figure 14 : ARC specifications with "pie" and "chord".

Description :

A circular arc 3 point pie arc or pie sector defined by three points (starting point, intermediate point and ending points) is a boundary closed by the arc specified by the three points and the two radii from starting point (resp. ending point) to the computed centre.

Such a circular arc 3 point pie is displayed with an interior style as defined by SET FILL AREA INDEX, SET, ASPECT SOURCE FLAGS and SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as for FILL AREA.

Discussion :

If the starting point and ending point of a circular 3 point pie are identical a filled area bounded by the circle which passes through the specified points and has a diameter equal to the distance from the starting point to the intermediate point is displayed.

If the intermediate point coincides with the starting or ending point, a straight line between the two points is displayed.

If the three points coincide, a dot is displayed.

If the three points are colinear, the primitive is ignored.

6.2.2.1.6.6 GDP CIRCULAR ARC CENTRE

Level : 0a

Parameters :

GDP identifier (= CIRCULAR ARC CENTRE)	(I)
Centre	(P)
Radius	(V)
Start vector	(P)
End vector	(P)

Description :

The radius parameter and the centre parameter together define a circle. The third parameter, together with the centre parameter, defines the start vector. The fourth parameter, together with the centre parameter, defines the end vector.

The starting (resp. ending) point of the circular arc centre is obtained by measuring a distance equal to the radius parameter along the start (resp. end) vector. The circular arc centre is drawn counterlockwise along the circle from starting point to ending point.

Related primitives :

Same as POLYLINE

Discussion :

If the radius parameter has the value 0 a dot is displayed at the centre point.

If the radius parameter is negative the primitive is ignored.

If the start vector and the vector coincide the primitive is ignored.

6.2.2.1.6.7 GDP CIRCULAR ARC CENTRE CHORD

Level : 0a

Parameters :

GDP identifier (= CIRCULAR ARC CENTRE CHORD)	(I)
Centre	(P)
Radius	(V)
Start vector	(P)
End vector	(P)

Description :

A circular arc centre chord is a boundary closed by the circular arc, specified by the parameters, and the chord from starting point of the circular arc to ending point of the circular arc (see definition of circular arc centre) for the construction of the circular arc, starting point and ending point).

Such a circular arc centre chord is displayed with an interior style as defined by SET FILL AREA INDEX, SET ASPECT SOURCE FLAGS and SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as POLYLINE

Discussion :

If the radius parameter has the value 0 a dot is displayed at the centre point.

If the radius parameter is negative the primitive is ignored.

If the start vector and the end vector coincide the primitive is ignored.

6.2.2.1.6.8 GDP CIRCULAR ARC CENTRE PIE

Level : 0a

Parameters :

GDP identifier (= CIRCULAR ARC CENTRE PIE)	(I)
Centre	(P)
Radius	(V)
Start vector	(P)
End vector	(P)

Description :

A circular arc centre pie is a boundary closed by the circular arc, specified by the parameters, and the two radii from starting point (resp. ending point) of the circular arc to the centre (see definition of circular arc centre for the construction of the circular arc, starting point and ending point).

Such a circular arc centre pie is displayed with an interior style as defined by SET FILL AREA INDEX, SET ASPECT SOURCE FLAGS and SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as POLYLINE

Discussion :

If the radius parameter has the value 0 a dot is displayed at the centre point.

If the radius parameter is negative the primitive is ignored.

If the start vector and the end vector coincide the primitive is ignored.

6.2.2.1.6.9 GDP ELLIPSE

Level : 0a

Parameters :

GDP identifier (= ELLIPSE)	(I)
Centre point	(P)
Endpoints (X1, Y1) and (X2, Y2)	(2xP)

Description :

An ellipse is specified by a Conjugate Diameter Pair (CDP). If the centre point is (XM, YM), the endpoints of an ellipse Conjugate Diameter Pair are defined by the two endpoints. The CDP vector components, relative to the centre point, are defined as follows :

```
DX1 = X1 - XM;
DY1 = Y1 - YM;
DX2 = X2 - XM;
DY2 = Y2 - YM;
```

The CDP vector components are the coefficients of the parametric equations :

```
X = XM + DX1 * cos(t) + DX2 * sin(t)
Y = YM + DY1 * cos(t) + DY2 * sin(t)
```

When t=0, (X, Y) is the endpoint of the first conjugate diameter, when t=pi/2 (X, Y) is the endpoint of the second conjugate diameter, and when t=2*pi the ellipse is closed and (X, Y) is again the endpoint of the first conjugate diameter.

The ellipse so defined is displayed with interior style as specified by SET FILL AREA INDEX, SET ASPECT SOURCE FLAGS and SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as for FILL AREA.

Discussion :

If the three points coincide a dot is displayed.

If the three points are colineair a straight line connecting the two endpoints is displayed.

If any two of three points are coinciding the primitive is ignored.

6.2.2.1.6.10 GDP ELLIPTIC ARC

Level : 0a

Parameters :

GDP identifier (= ELLIPTIC ARC)	(I)
Centre point	(P)
Endpoints (X1, Y1) and (X2, Y2)	(2xP)
T_start, T_end	(2xR)

Description :

The centre point and the two endpoints together define an ellipse (see section 6.2.2.1.6.5).

T_start and T_end correspond to two points on the ellipse. The defined elliptic arc is that which begins at T_start and goes to T_end in the direction established by the coefficients of the parametric equations (the CDP vector components DX1, DY1, DX2, DY2).

The elliptic arc is displayed with the attributes defined for POLYLINE.

Related primitives :

Same as for POLYLINE.

Discussion :

If the centre point and the endpoints coicinde a dot is displayed.

If the centre point and the endpoints are colineair a straight line connecting the two endpoints is displayed.

If any two of the centre point and the two endpoints are coinciding the primitive is ignored.

If T_start equals T_end the full ellipse is displayed.

6.2.2.1.6.11 GDP ELLIPTIC ARC CHORD

Level : 0a

Parameters :

GDP identifier (= ELLIPTIC ARC CHORD)	(I)
Centre point	(P)
Endpoints (X1, Y1) and (X2, Y2)	(2xP)
T_start, T_end	(2xR)

Description :

An elliptic arc chord is a boundary closed by the elliptic arc, defined by the parameters, and the chord from the starting to the ending point of the elliptic arc (see section 6.2.2.1.6.6).

The elliptic arc chord is displayed with interior style as as defined by SET FILL AERA INDEX, SET ASPECT SOURCE FLAGS ans SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as for FILL AREA.

Discussion :

If the centre point and the endpoints coincide a dot is displayed.

If the centre point and the endpoints are colinear a straight line connecting the two endpoints is displayed.

If any two the centre point and the two endpoints are coinciding the primitive is ignored.

If T_start equals T_end the full ellipse is displayed.

6.2.2.1.6.12 GDP ELLIPTIC ARC PIE

Level : 0a

Parameters :

GDP identifier (= ELLIPTIC ARC PIE)	(I)
Centre point	(P)
Endpoints (X1, Y1) and (X2, Y2)	(2xP)
T_start, T_end	(2xR)

Description :

An elliptic arc pie is a boundary closed by the elliptic arc, defined by the parameters and the lines from the starting and ending points of this elliptic arc to the centre point (Cf. GDP ELLIPTIC ARC).

The elliptic arc pie is displayed with an interior style as defined by SET FILL AREA INDEX, SET ASPECT SOURCE FLAGS ans SET FILL AREA INTERIOR STYLE.

Related primitives :

Same as for FILL AREA

Discussion :

If the centre point and the endpoints coincide a dot is displayed.

If the centre point and the endpoints are colinear a straight line connecting the two endpoints is displayed.

If any two of the centre point and the two endpoints are coinciding the primitive is ignored.

If T_start equals T_ends the full ellipse is displayed.

6.2.2.1.6.13 GDP SPLINE

Level : 0a

Parameters :

GDP identifier (= SPLINE)	(I)
Point list (3..n)	(nxP)

Description :

A smooth curve is drawn based on the specified points. This curve, known as a uniform quadratic B-spline, is displayed with the attributes as defined for the display of POLYLINE.

Related primitives :

Same as for POLYLINE

Discussion : None

6.2.2.2 Output primitives related to display element attributes

6.2.2.2.1 SET POLYLINE REPRESENTATION

Level : 1a

Parameters :

Workstation identifier	(ID)
Polyline index (1..n)	(IX)
Line type	(I)
Line width scale factor	(V)
Polyline colour index	(CI)

Description

In the polyline bundle table of the specified workstation the given index is associated with the specified parameters:

Line type
Line width scale factor
Polyline colour index

The polyline bundle table in the workstation has predefined entries. Any table entry may be redefined.

Which of the aspects in an entry thare are used depends upon the setting of the corresponding ASFs.

Related primitives :

POLYLINE
SET POLYLINE INDEX
GDP CIRCULAR 3 POINT
GDP CIRCULAR ARC CENTRE
GDP ELLIPTIC ARC
GDP SPLINE

Discussion :

If one or more parameters are invalid or out of range, the primitive is ignored.

6.2.2.2.2 SET POLYLINE INDEX

Level : 0a

Parameters :

Polyline index (1..n)	(IX)

Description :

The polyline index is set to the value specified by the parameter.

Related primitives :

SET ASPECT SOURCE FLAGS

Discussion :

If the polyline index parameter is out of range, the default value is set.

6.2.2.2.3 **SET LINE TYPE**

Level : 0a

Parameters :

Line type
(one of : SOLID, DASHED, DOTTED, DASHED-DOTTED, (I)
<other implementation dependent or reserved for future standardization >).

Description :

The line type is set to the value specified by the parameter.

When the line type ASF is 'INDIVIDUAL', subsequent display elements using POLYLINE attributes are displayed with this line type.

When the line type ASF is 'BUNDLED', this primitive does not affect the display of subsequent display elements using POLYLINE attributes until the ASF returns to 'INDIVIDUAL'.

Related primitives :

POLYLINE
GDP CIRCULAR ARC 3 POINT
GDP CIRCULAR ARC CENTRE
GDP ELLIPTIC ARC
GDP SPLINE

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.4 **SET LINE WIDTH SCALE FACTOR**

Level : 0a

Parameters :

Line width scale factor (R)

Description :

The line width is set to the value specified by the parameter.

When the line width ASF is 'INDIVIDUAL', subsequent display elements using POLYLINE attributes are displayed with this line width.

When the line width ASF is 'BUNDLED', this primitive does not affect the display of subsequent display elements using POLYLINE attributes until the ASF returns to 'INDIVIDUAL'.

Related primitives :

POLYLINE
GDP CIRCULAR ARC 3 POINT
GDP CIRCULAR ARC CENTER
GDP ELLIPTIC ARC
GDP SPLINE

Discussion :

If the line width scale factor parameter has the value 0 the default line width will be set.

If the line width scale factor parameter is less than 0, the primitive is ignored.

6.2.2.2.5 SET POLYLINE COLOUR INDEX

Level : 0a

Parameters :

Polyline colour index (CI)

Description :

The polyline colour index is set to the value specified by the parameter.

When the POLYLINE COLOUR ASF is 'INDIVIDUAL', subsequent display elements using POLYLINE attributes using this line colour.

When the POLYLINE COLOUR ASF is 'BUNDLED' this primitive does not affect the display of subsequent display elements using POLYLINE attributes until the ASF returns to 'INDIVIDUAL'.

Related primitives :

POLYLINE
GDP CIRCULAR ARC 3 POINT
GDP CIRCULAR CENTRE
GDP ELLIPTIC ARC
GDP SPLINE

Discussion :

If the parameter value is out of range, polyline colour index is set to the default value.

6.2.2.2.6 SET POLYMARKER REPRESENTATION

Level : 1a

Parameters :

Workstation identifier	(ID)
Polymarker index (1..n)	(IX)
Marker type	(I)
Marker size scale factor	(V)
Polymarker colour index	(CI)

Description

In the polymarker bundle table of the specified workstation the given polymarker index is associated with the specified parameters:

Marker type
Marker size scale factor
Polymarker colour index

The polymarker bundle table in the workstation has predefined entries. Any table entry (including the predefined entries) may be redefined.

Which of the aspects in the entry are used depends upon the setting of the corresponding ASFs.

Related primitives :

POLYMARKER
SET POLYMARKER INDEX

Discussion :

If one or more parameters are invalid or out of range, the primitive is ignored.

6.2.2.2.7 SET POLYMARKER INDEX

Level : 0a

Parameters :

Polymarker index (1..n)	(IX)

Description :

The polymarker index is set to the value specified by the parameter. This value is used when creating subsequent POLYMARKER display elements.

Related primitives :

SET ASPECT SOURCE FLAGS.

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.8 SET MARKER TYPE

Level : 0a

Parameters :

Marker type
(one of : DOT, PLUS, ASTERISK, CIRCLE, DIAGONAL CROSS, (I)
<other implementation dependent or reserved for future standardization>).

Description :

The marker type is set to the value specified by the parameter.

When the Polymarker type ASF is 'INDIVIDUAL' subsequent POLYMARKER elements are displayed with this marker type.

When the Polymarker type ASF is 'BUNDLED', this primitive does not affect the display of subsequent POLYMARKER elements until the ASF returns to 'INDIVIDUAL'.

The marker types produce centred symbols as indicated :

dot	.
plus	+
asterisk	*
circle	o
diagonal cross	x

Related primitives :

POLYMARKER

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.9 SET MARKER SIZE SCALE FACTOR

Level : 0a

Parameters :

Marker size scale factor (R)

Description :

The marker size is set to the value specified by the parameter.

When the marker size ASF is 'INDIVIDUAL' subsequent POLYMARKER elements are displayed with this marker size.

When the marker size ASF is 'BUNDLED', this primitive does not affect the display of subsequent POLYMARKER elements until the ASF returns to 'INDIVIDUAL'.

Related primitives :

POLYMARKER

Discussion :

If the specified marker size is zero, the default value is set.

If the specified marker size value is less than zero, the primitive is ignored.

6.2.2.2.10 **SET POLYMARKER COLOUR INDEX**

Level : 0a

Parameters :

Polymarker colour index (CI)

Description :

The polymarker colour index is set to the value specified by the parameter.

When the Polymarker colour ASF is 'INDIVIDUAL' subsequent POLYMARKER elements are displayed with this polymarker colour index.

When the Polymarker colour ASF is 'BUNDLED', this primitive does not affect the display of subsequent POLYMARKER elements until the ASF returns to 'INDIVIDUAL'.

Related primitives :

POLYMARKER

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.11 **SET FILL AREA REPRESENTATION**

Level : 1a

Parameters :

Workstation identifier	(ID)
Fill area index (1..n)	(IX)
Fill area interior style	(E)
(one of : HOLLOW, SOLID, PATTERN, HATCH)	
Fill area style index	(I)
interior style = HATCH	
or	
Fill area style index (1..n)	(IX)
interior style = PATTERN	
Fill area colour index	(CI)

Description :

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

Fill area interior style
Fill area style index (hatch or pattern)
Fill area colour index

The fill area bundle table in the workstation has predefined entries. Any table entry (including the predefined entries) may be redefined.

Which of the aspects in the entry are used depends upon the setting of the corresponding ASFs.

Related primitives :

FILL AREA
SET FILL AREA INDEX
GDP RECTANGLE
GDP CIRCLE
GDP CIRCULAR ARC 3 POINT CHORD
GDP CIRCULAR ARC 3 POINT PIE
GDP CIRCULAR ARC CENTRE CHORD
GDP CIRCULAR ARC CENTRE PIE
GDP ELLIPSE
GDP ELLIPTIC ARC PIE
GDP ELLIPTIC ARC CHORD

Discussion :

If one or more parameters are invalid or out of range, the primitive is ignored.

6.2.2.2.12 SET FILL AREA INDEX

Level : 0a

Parameters :

Fill area index (1..n)	(IX)

Description :

The fill area index is set to the value specified by the parameter. This value is used when filling subsequent display elements which define closed boundaries.

Related primitives :

SET ASPECT SOURCE FLAGS

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.13 SET FILL AREA INTERIOR STYLE

Level : 0a

Parameters :

Fill area interior style (E)
(one of : HOLLOW, SOLID, PATTERN, HATCH)

Description :

The interior style of the display elements which define closed boundaries is set to the value specified by the parameter.

When the Fill Area Interior Style ASF is ' INDIVIDUAL ', subsequent elements are displayed with this interior style.

When the Fill Area Interior Style ASF is 'BUNDLED', this element does not affect the display of these subsequent display elements until the ASF returns to 'INDIVIDUAL'.

The fill area Interior Style is used to determine in what style the area is to be filled :

'hollow' - no filling. The boundary is drawn using the fill area colour index currently selected (either bundled or individually depending on the corresponding fill area colour ASF).

'solid' - fill the interior using the fill colour area colour index currently selected (either bundled or individually, depending on the corresponding Fill Area Colour ASF).

'pattern' - fill the interior using the Fill Area style index currently selected (either bundled or individually, depending on the corresponding Fill Area Style Index ASF).

'hatch' - fill the interior using the fill area colour index and the fill style index currently selected (either bundled or individually, depending on the corresponding Fill Area Colour ASF and Fill Area style index ASF).

Related primitives :

FILL AREA
GDP RECTANGLE
GDP CIRCLE
GDP CIRCULAR ARC 3 POINT CHORD
GDP CIRCULAR ARC 3 POINT PIE
GDP CIRCULAR ARC CENTRE CHORD
GDP CIRCULAR ARC CENTRE PIE
GDP ELLIPSE
GDP ELLIPTIC ARC CHORD
GDP ELLIPTIC ARC PIE
SET FILL AREA COLOUR INDEX
SET FILL AREA STYLE INDEX

Discussion : None

6.2.2.2.14 **SET FILL AREA COLOUR INDEX**

Level : 0a

Parameters :

Fill area colour index (CI)

Description :

The fill area colour index is set to the value specified by the parameter.

When the Fill Area Colour ASF is 'INDIVIDUAL', subsequent display elements which define closed boundaries are filled using this colour index.

When the Fill Area Colour ASF is 'BUNDLED', this primitive does not affect the display of subsequent display elements which define closed boundaries until the ASF returns to 'INDIVIDUAL'.

The Fill Area Colour index is only significant if the Fill Area Interior Style is either 'SOLID', 'HATCH ' or 'HOLLOW'.

Related primitives :

FILL AREA
GDP RECTANGLE
GDP CIRCLE
GDP CIRCULAR ARC 3 POINT CHORD
GDP CIRCULAR ARC 3 POINT PIE
GDP CIRCULAR ARC CENTRE CHORD
GDP CIRCULAR ARC CENTRE PIE
GDP ELLIPSE
GDP ELLIPTIC ARC CHORD
GDP ELLIPTIC ARC PIE

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.15 **SET FILL AREA STYLE INDEX**

Level : 0a

Parameters :

Fill area style index (I)
if (interior style = HATCH)
or
Fill area style index (IX)
if (interior style = PATTERN)

Description :

The fill area style index is set to the value specified by the parameter.

When the Fill Area style index ASF is 'INDIVIDUAL', subsequent display elements which define closed boundaries are displayed using this Fill Area style index.

When the Fill Area style index ASF is 'BUNDLED', this primitive element does not affect the display of subsequent display elements which define closed boundaries until the ASF returns to 'INDIVIDUAL'.

Related primitives :

FILL AREA
GDP RECTANGLE
GDP CIRCLE
GDP CIRCULAR ARC 3 POINT CHORD
GDP CIRCULAR ARC 3 POINT PIE
GDP CIRCULAR ARC CENTRE CHORD
GDP CIRCULAR ARC CENTRE PIE
GDP ELLIPSE
GDP ELLIPTIC ARC CHORD
GDP ELLIPTIC ARC PIE

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.16 **SET PATTERN REPRESENTATION**

Level : 1a

Parameters :

Workstation identifier	(ID)
Pattern index (1..n)	(IX)
Delta X (DX)	(I)
Delta Y (DY)	(I)
Pattern cell colour index list	(CL)

Description :

The DX and DY values define a horizontal by vertical DX by DY array into which colour values are mapped. The array is loaded sequentially by rows starting with the upper-left (maximum y, minimum x) corner.

Related primitives :

FILL AREA
GDP RECTANGLE
GDP CIRCLE
GDP CIRCULAR ARC 3 POINT CHORD
GDP CIRCULAR ARC 3 POINT PIE
GDP CIRCULAR ARC CENTRE CHORD
GDP CIRCULAR ARC CENTER PIE
GDP ELLIPSE
GDP ELLIPTIC ARC CHORD
GDP ELLIPTIC ARC PIE
SET FILL AREA INTERIOR STYLE

Discussion :

If one or more parameters are invalid or out of range, the primitive is ignored.
If the number of indices in the colour index list is less than DX x DY, then the last index is repeated until the end of the list.
If the number of indices in the colour index list is greater than DX x DY, then the exceeding indices are omitted.
If delta X or delta Y are less than or equal to 0 the primitive is ignored.

6.2.2.2.17 **SET PATTERN REFERENCE POINT**

Level : 0a

Parameters :

Reference point	(P)

Description :

The pattern reference point is set to the value specified by the parameter.

When the currently selected interior style is 'PATTERN', this value is used in conjunction with the pattern vectors for displaying filled areas.

When the currently selected interior style is 'HATCH', it is implementation dependent if the pattern reference point is used for displaying filled areas. The position of the start of the pattern (or hatch) is defined by the pattern reference point. The pattern is mapped onto the filled area by conceptually replicating it in directions parallel to the sides of the pattern box until the interior of the fill area is completely covered.

Related primitives :

FILL AREA
GDP RECTANGLE
GDP CIRCLE
GDP CIRCULAR ARC 3 POINT CHORD
GDP CIRCULAR ARC 3 POINT PIE
GDP CIRCULAR ARC CENTRE CHORD
GDP CIRCULAR ARC CENTER PIE
GDP ELLIPSE
GDP ELLIPTIC ARC CHORD
GDP ELLIPTIC ARC PIE
SET FILL AREA INTERIOR STYLE
SET PATTERN REPRESENTATION
SET PATTERN VECTORS

Discussion : None

6.2.2.2.18 **SET PATTERN VECTORS**

Level : 0a

Parameters :

Height vector	(P)
Width vector	(P)

Description :

The origin of the NDC Space and the first point define the pattern height vector. The origin of the NDC Space and the second point define the pattern width vector.

When interior style is set to 'PATTERN', subsequent display elements which define closed boundaries are displayed with this pattern vectors.

The height and width vector define a paralleogram. This area is subdivided into DX x DY contiguous parallelograms of dimension <length of height vector> / DX by <length of width vector> / DY, using the dimensions DX and DY of the pattern table entry selected by the current fill area style index. The fill area style index can be selected either BUNDLED or INDIVIDUALLY.

Colours are mapped into the resulting DX by DY array, defining an area texture. Conceptually, the lower left corner of the parallelogram is placed at the pattern reference point and replicated as necessary, horizontally and vertically over the display surface. The coincidence of this imposed pattern and the interior to which it is to be applied defines the interior style for the filled area display element being created.

Related primitives :

FILL AREA
GDP RECTANGLE
GDP CIRCLE
GDP CIRCULAR ARC 3 POINT CHORD
GDP CIRCULAR ARC 3 POINT PIE
GDP CIRCULAR ARC CENTRE CHORD
GDP CIRCULAR ARC CENTRE PIE
GDP ELLIPSE
GDP ELLIPTIC ARC CHORD
GDP ELLIPTIC ARC PIE
SET FILL AREA INTERIOR STYLE
SET PATTERN REPRESENTATION
SET PATTERN REFERENCE POINT

Discussion :

If the two vectors are coinciding or at 180 degrees, the primitive is ignored.

6.2.2.2.19 **SET TEXT REPRESENTATION**

Level : 1a

Parameters :

Workstation identifier	(ID)
Text index (1..n)	(IX)
Text font	(I)
Text precision	(E)
Character expansion factor	(R)
Character spacing	(R)
Text colour index	(CI)

Description :

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

Text font and precision
Character expansion factor
Character spacing
Text colour index

Text fonts greater than 0 are for future standardization, text font less than 0 are implementation dependent.

The text bundle table in the workstation has predefined entries. Any entry (including the predefined entries) may be redefined.

Which of the aspects in the entry are used depends upon the setting of the corresponding ASFs.

Related primitives :

TEXT
SET TEXT INDEX

Discussion :

If one or more parameters are invalid or out of range, the primitive is ignored.

6.2.2.2.20 **SET TEXT INDEX**

Level : 0a

Parameters :

Text Index (1..n) (IX)

Description :

The text index is set to the value specified by the parameter. This value is used when processing subsequent TEXT display elements while the ASF is set to 'BUNDLED'.

Related primitives :

SET ASPECT SOURCE FLAGS.
TEXT

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.21 **SET TEXT FONT AND PRECISION**

Level : 0a

Parameters :

Text font (I)
Text precision (one of : STRING, CHARACTER, STROKE) (E)

Description :

The Text Font and Precision are set to the values specified by the parameters.

Text fonts greater than 0 are reserved for future standardization, text fonts less than 0 are implementation dependent.

When the Text Font and Precision ASF is 'INDIVIDUAL', subsequent TEXT display elements are displayed with this text precision and using this text font.

When the Text Font and Precision ASF is 'BUNDLED', this primitive does not affect the display of subsequent TEXT elements until the ASF returns to 'INDIVIDUAL'.

The accuracy of execution of text attributes can be controlled by one of three values for text precision.

If 'STRING' precision is specified, the text string is generated in the requested text font and is positioned by aligning the text string at the given text position. The length of the character vectors and the character expansion factor are evaluated as closely as reasonable, given the capabilities of the workstation. The direction of the character vectors, the text path, the text alignement and the character spacing need not be used. Clipping is done in an implementation dependent way.

If 'CHARACTER' precision is specified, the text string is generated in the requested text font. For the representation of each individual character, both the length and the direction of the character vectors and the 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 text string is determined by the text alignment and the text position. Clipping is performed at least on a character by character basis.

If 'STROKE' precision is specified, the text string is displayed at the text position by applying all text aspects. The next string is clipped exaclty at the clipping rectangle.

Related primitives :

TEXT

Discussion :

If one or more parameters values are out of range, the default values are set.

6.2.2.2.22 SET CHARACTER EXPANSION FACTOR

Level : 0a

Parameters :

Character expansion factor (R)

Description :

The character expansion factor is set to the value specified by the parameter.

When the Character Expansion Factor ASF is 'INDIVIDUAL', subsequent TEXT elements are displayed with this character expansion factor.

When the Character Expansion Factor ASF is 'BUNDLED', this primitive does not affect the display of subsequent TEXT elements until the ASF returns to 'INDIVIDUAL '.

The character expansion factor specifies the deviation of the width/height ratio of the characters from the ratio indicated by the font designer.

The character expansion factor is a nondimensional scalar. The desired resulting character width is given by the length of the character width vector multiplied by the font width to height ratio for the character and by the character expansion factor.

Related primitives :

TEXT
SET CHARACTER VECTORS

Discussion :

If the character expansion factor has the value 0, default character expansion factor will be set.

6.2.2.2.23 **SET CHARACTER SPACING**

Level : 0a

Parameters :

Character Spacing (R)

Description :

The character spacing is set to the value specified by the parameter.

When the Character Spacing ASF is 'INDIVIDUAL', subsequent TEXT display elements are displayed with this character spacing.

When the Character Spacing ASF is 'BUNDLED', this primitive does not affect the display of subsequent TEXT display elements until the ASF returns to 'INDIVIDUAL'.

The parameter represents the desired space to be added between characters of a text string. It is specified as a fraction of the current Character Height Vector attribute. The space is added along the character path. A negative value implies that characters may overlap.

Related primitives :

TEXT
SET CHARACTER VECTORS

Discussion :

The parameter is compared to the dimensions of available intercharacter space in the device. The available value next smaller than or equal to the specified value is selected. If no such value is available, the next larger value may be selected.

6.2.2.2.24 SET TEXT COLOUR INDEX

Level : 0a

Parameters :

Text colour index (CI)

Description :

The text colour index is set as specified by the parameter.

When the Text colour ASF is 'INDIVIDUAL', subsequent TEXT display elements are displayed using this text colour index. When the text colour ASF is 'BUNDLED' this primitive does not affect the display of subsequent TEXT display elements until the ASF returns to 'INDIVIDUAL'.

Related primitives :

TEXT

Discussion :

If the parameter value is out of range, the default value is set.

6.2.2.2.25 SET TEXT PATH

Level : 0a

Parameters :

Text path (one of : RIGHT, LEFT, UP, DOWN) (E)

Description :

The text path is set to the value specified by the parameter. Subsequent TEXT display elements are displayed with this text path.

This primitive sets the value of the text path attribute, specifying the writing direction of a text string relative to the character height vector and character width vector.

'Right' means in the direction of the character width vector.
'Left' means 180 degrees from the character width vector.
'Up' means in the direction of the character height vector.
'Down' means 180 degrees from the character height vector.

Related primitives :

TEXT
SET CHARACTER VECTORS

Discussion : None

6.2.2.2.26 SET CHARACTER VECTORS

Level : 0a

Parameters :

Height vector (P)
Width vector (P)

Description :

The origin of the NDC space and the first point define the character height vector. The origin of the NDC space and the second point define the character width vector.

The direction of the two vectors defines both the orientation and the skew of the character body and the sense of the text path in subsequent TEXT display elements.

The length of the character height vector defines the height of the character within the character body and is also used as a scaling factor when calculating the desired space to be added between the characters. The length of the width vector is used as a factor when calculating the width of the character within the character body.

The character expansion factor is a non dimensional scalar. The desired resulting character width is the product of the length of the character width vector times the width/height ratio for the character times the character expansion factor.

Related primitives :

TEXT
SET TEXT PATH
SET CHARACTER SPACING
SET CHARACTER EXPANSION FACTOR
SET TEXT ALIGNMENT

Discussion :

If the two vectors are coincident or at 180-degrees, the primitive is discarded.

When the primitive is processed, it is transformed by whatever transformation the workstation is using to map NDC coordinates to device coordinates, and the resulting value is compared to the set of available character heights in the device.
The available value next smaller than or equal to the transformed value is selected. If no such value is available, the next larger value may be selected.

6.2.2.2.27 **SET TEXT ALIGNMENT**

Level : 0a

Parameters :

Horizontal alignment (one of : NORMAL, LEFT, CENTRE, RIGHT) (E)
Vertical alignment (one of : NORMAL, TOP, CAP, HALF, BASE, BOTTOM)(E)

Description :

The text alignment is set to the value specified by the parameters. Subsequent text strings are displayed with this text alignment.

Related primitives :

TEXT
SET CHARACTER VECTORS

Discussion :

The "NORMAL" parameters value are dependent on the text path at the time of the elaboration of the TEXT display element.

PATH	NORMAL HORIZONTAL	NORMAL VERTICAL
RIGHT	LEFT	BASE
LEFT	RIGHT	BASE
UP	CENTRE	BASE
DOWN	CENTRE	TOP

6.2.2.2.28 **SET COLOUR REPRESENTATION**

Level : 0a

Parameters :

Workstation identifier (ID)
Starting entry in the colour table (CI)
Colour data list (nxCD)

Description :

This primitive is used to load colour table. The first parameter specifies the workstation ; the second parameter defines the first table entry to be loaded.

The number of bits used to define an intensity is defined by the 'unit resolution' parameter of the last set COLOUR HEADER primitive processed.

The end of the colour data list parameter (and therefore the number of entries n in the colour data list) is implicitly indicated by the receipt of the next primitive to be processed.

Related primitives :

SET COLOUR HEADER

Discussion :

If the first of second parameters are invalid or out of range the primitive is ignored. If the last colour data in the colour data list is incomplete this truncated colour data is ignored.

6.2.2.2.29 SET ASPECT SOURCE FLAGS

Level : 0a

Parameters :

Line type ASF	(E)
Line width scale factor ASF	(E)
Polyline colour ASF	(E)
Marker type ASF	(E)
Marker size scale factor ASF	(E)
Polymarker colour ASF	(E)
Text font and precision ASF	(E)
Character expansion factor ASF	(E)
Character spacing ASF	(E)
Text colour ASF	(E)
Fill area interior style ASF	(E)
Fill area style index ASF	(E)
Fill area colour ASF	(E)

Description :

The aspect source flags (ASFs) are set to the values specified by the parameters.

Related primitives : None

Discussion : None

6.2.2.2.30 SET PICK IDENTIFIER

Level : 1a

Parameters :

Pick identifier	(ID)

Description :

The current pick identifier is set to the value specified by the parameter.

Related primitives :

REQUEST LOGICAL INPUT DEVICE
SAMPLE LOGICAL INPUT DEVICE
GET LOGICAL INPUT DEVICE
INITIALIZE LOGICAL INPUT DEVICE

Discussion :

The pick identifier has meaning for workstation of category OUTIN with WDSS capabilities or for a graphics device composed of a workstation of category OUTPUT, a workstation of category INPUT and a workstation of category WISS.

6.2.2.3. Transformation primitives

6.2.2.3.1 SET WORKSTATION WINDOW

Level : 0a

Parameters :

Workstation identifier	(ID)
Window limits	(2xP)

Description :

The parameters specify a rectangle in the unit square of the NDC space used for the display of images.

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.

Related primitives :

SET WORKSTATION VIEWPORT

Discusion :

If the points are outside the unit square, the default workstation window is set.

6.2.2.3.2 SET WORKSTATION VIEWPORT

Level : 0a

Parameters :

Workstation identifier	(ID)
Viewport limits	(4xR)
(xmin < xmax and ymin < ymax)	

Description :

The parameters specify a rectangle in the DC space.

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.

Related primitives :

SET WORKSTATION WINDOW

Discussion :

If the points are outside the display space, the default workstation viewport is set.

6.2.2.4. Clipping primitives

6.2.2.4.1 SET CLIPPING RECTANGLE

Level : 0a

Parameters :

Clipping rectangle limits	(2xP)

Description :

The parameters specify a rectangle in the unit square of the NDC space, which defines the clipping rectangle.

Related primitives : None

Discussion :

This primitive is necessary to provide local clipping areas within the NDC space. For example, high-quality text could be sent to the device as text string and clipping rectangle so that the interpreter can clip character strokes.
If the points are outside the unit square, the default clipping rectangle is set.

6.2.2.5. Control primitives :

6.2.2.5.1 UPDATE WORKSTATION

Level : 0a

Parameters :

Workstation identifier	(ID)
Update regeneration flag (one of : PERFORM, POSTPONE)	(E)

Description :

All deferred actions for the specified workstation are executed (without intermediate clearing of the display surface). If the 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' entry of the workstation state list) are redisplayed. This action typically causes the 'display surface empty' entry of the workstation state list to be set to NOTEMPTY.

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

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

```
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 regeneration flag is POSTPONE, UPDATE WORKSTATION merely initiates the transmission of blocked data. If the value of the 'new frame action necessary at update' entry 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 5.5) :

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).

Related primitives :

SET DEFERRAL STATE

Discussion : None

6.2.2.5.2 **SET DEFERRAL STATE**

Level : 1a

Parameters :

Workstation identifier	(ID)
Deferral mode (one of : ASAP, BNIL, BNIG, ASTI)	(E)
Implicit regeneration flag (one of : SUPPRESSED, ALLOWED)	(E)

Description :

The deferral mode and implicit regeneration flag of the specified workstation are set to the values specified by the parameters. Depending of the new value of deferral mode, deferred output may be unblocked. If the new value of implicit regeneration flag is ALLOWED and 'newframe action necessary at update' entry in the workstation state list is YES, an action equivalent to REDRAW ALL SEGMENTS ON WORKSTATION is performed.

Related primitives :

UPDATE WORKSTATION

Discussion : None

6.2.2.5.3 **EMERGENCY CLOSE**

Level : 0a

Parameters : None

Description :

The physical device is emergency closed. The following actions are performed (if possible) :

a) CLOSE SEGMENT (if open) ;

b) UPDATE WORKSTATION with update regeneration flag PERFORM for all open workstations ;

c) DEACTIVATE WORKSTATION for all active workstations ;

d) CLOSE WORKSTATION for all open workstations ;

Related primitives : None

Discussion : None

6.2.2.5.4 ERROR DETECTED (RESPONSE)

Level : 0a

Parameters : None

Description :

In synchronous interactions, this primitive is issued by the physical device after it has detected an error or an inquire primitive which it is not able to process.

Related primitives :

INQUIRE primitives

Discussion : None

6.2.3 SEGMENT RELATED PRIMITIVES

6.2.3.1 WDSS related primitives

6.2.3.1.1 CREATE SEGMENT

Level : 1a

Parameters :

Segment Name (ID)

Description :

A segment is opened with the name < segment name >. All subsequent output primitives belong to this segment until the first CLOSE SEGMENT primitive.

Related primitives :

CLOSE SEGMENT

Discussion : None

6.2.3.1.2 **CLOSE SEGMENT**

Level : 1a

Parameters : None

Description :

No further primitives will be added to the current open segment. If no segment was open, this primitive will have no effect.

Related primitives :

CREATE SEGMENT

Discussion : None

6.2.3.1.3 **RENAME SEGMENT**

Level : 1a

Parameters :

Old segment name	(ID)
New segment name	(ID)

Description :

Each occurrence of old segment name is replaced by new segment name.

If old segment name is the name of the open segment, the name of the open segment is set to the new segment name.

Related primitives : None

Discussion :

If old segment name doesn't exist in a workstation, the primitive has no effect on that workstation.

6.2.3.1.4 **DELETE SEGMENT FROM WORKSTATION**

Level : 1a

Parameters :

Workstation identifier	(ID)
Segment name	(ID)

Description :

The named segment is deleted from the specified workstation and the <segment name> is removed from the set of segment names in use for this workstation.

Related primitives : None

Discussion :

If the specified segment is not in use for this workstation, this primitive has no effect.

6.2.3.1.5 DELETE SEGMENT

Level : 1a

Parameters :

Segment Name (ID)

Description :

The named segment and the <Segment Name> are deleted from all workstation segment storages.

Related primitives : None

Discussion :

If the named segment does not exist, this primitive has no effect.

6.2.3.1.6 REDRAW ALL SEGMENTS ON WORKSTATION

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

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

c) The current workstation transformation is set to the last defined WORKSTATION WINDOW and WORKSTATION VIEWPORT, if the 'workstation transformation update state' entry on the workstation state list is PENDING. The entry is set to NOTPENDING.

d) All visible segments stored of this workstation are redisplayed.

Related primitives : None

Discussion : None

6.2.3.1.7 **SET HIGHLIGHTING**

Level : 1a

Parameters :

Segment Name (ID)
Highlighting flag (one of : NOTHIGHLIGHTED, HIGHLIGHTED) (E)

Description :

The highlighting attribute of the named segment is set to the value specified by the parameter.

Related primitives : None

Discussion : None

6.2.3.1.8 **SET VISIBILITY**

Level : 1a

Parameters :

Segment Name (ID)
Visibility flag (one of : VISIBLE, INVISIBLE) (E)

Description :

The visibility attribute of the named segment is set to the value specified by the parameter.

Related primitives : None

Discussion : None

6.2.3.1.9 **SET SEGMENT TRANSFORMATION**

Level : 1a

Parameters :

Segment Name (ID)
Transformation matrix (M)

Description :

The segment transformation matrix is set to the value specified by the parameter. When a segment is displayed, the coordinates of its display elements 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 VECTORS), the values M_{13} and M_{23} are ignored.

The function can be used to transform a stored segment.

The segment transformation (conceptually) takes place in NDC space.
This transformation is not cumulative, i.e. it always applies to the segment as originally created.

Related primitives :

INSERT SEGMENT
REDRAW ALL SEGMENTS ON WORKSTATION

Discussion : None

6.2.3.1.10 **SET SEGMENT PRIORITY**

Level : 1a

Parameters :

Segment Name	(ID)
Segment Priority	(R)

Description :

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 precedence is given to segments with higher priority. If segments with the same priority overlap, the result is implementation dependent.

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. If 'number of segment priorities surpported' 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.

Related primitives :

REDRAW ALL SEGMENTS ON WORKSTATION

Discussion : None

6.2.3.1.11 SET DETECTABILITY

Level : 1a

Parameters :

Segment Name	(ID)
Detectability flag (one of : DETECTABLE, UNDETECTABLE)	(E)

Description :

The 'dectectabiliy' flag for 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.

Related primitives : None

Discussion : None

6.2.3.2 WISS related primitives

6.2.3.2.1 ASSOCIATE SEGMENT WITH WORKSTATION

Level : 2a

Parameters :

Workstation identifier	(ID)
Segment name	(ID)

Description :

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.

Related primitives : None

Discussion :

If the segment is not present in the WISS or if it is already associated with the specified workstation, the primitive has no effect.

6.2.3.2.2 COPY SEGMENT TO WORKSTATION

Level : 2a

Parameters :

Workstation identifier	(ID)
Segment name	(ID)

Description :

The primitives in the specified segment are sent to the indicated workstation after segment transformation and clipping at the clipping rectangle stored with each primitive.

The primitives are not stored in a segment.

All display elements keep the values of the display element attributes, that where assigned to them when they were created, for their whole lifetime.

In particular, when segments are copied, the values of the primitive attributes within the copied segments are unchanged.

Related primitives : None

Discussion

If the segment is not present in the WISS or if the specified workstation is the WISS, this primitive has no effect.

6.2.3.2.3 INSERT SEGMENT

Level : 2a

Parameters :

Segment name	(ID)
Transformation matrix	(M)

Description :

Having been transformed as defined below, the primitives contained in the segment are drawn on the display surface and, eventually, stored in the open segment.

The coordinates of the primitives contained in the inserted segment are transformed firstly by any segment transformation specified for it, and secondly by applying the following matrix multiplication to them :

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

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 (example CHARACTER VECTORS) the values M_{13} and M_{23} are ignored.

Other than the segment transformation, attributes of the inserted segment are ignored.

All clipping rectangle in the inserted segment are ignored each primitive processed is assigned a new clipping rectangle which is the current CLIPPING RECTANGLE.

All primitives processed by a single invocation of INSERT SEGMENT receive the same clipping rectangle. All primitives keep the values of the primitive attributes, that were assigned to them when they were created.

In particular, where segments are inserted, the values of the primitive attributes within the inserted segments are unchanged. The values of primitive attributes used in the creation of subsequent primitives within the segment into which the insertion takes place are unaffected by that insertion.

Related primitives :

SET SEGMENT TRANSFORMATION

Discussion : None

6.2.4 INPUT PRIMITIVES

6.2.4.1 INITIALIZE LOGICAL INPUT DEVICE

Level : 0b

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number	(I)
Initial value	
(Depends on the logical input device class)	
Prompt and echo type	(I)
Echo area (Xmin < Xmax, Ymin < Ymax)	(4xR)
Data record	
(Depends on the logical input device class).	

Description :

The initial value and the data record vary according to the logical input class as follows :

LOCATOR device :
The initial value is the initial position (P).
The initial locator position, prompt and echo types, echo area are stored into the workstation state list entry for the specified LOCATOR device.
For some LOCATOR prompt and echo types, two positions are required. The first, which remains fixed during the input operations, is the initial LOCATOR position. The second is the current LOCATOR position and can vary dynamically.

The prompt and echo types are :

< 0 prompting and echoing is LOCATOR device dependent,
1 implementation-dependent,
2 crosshair : The current position is indicated by a vertical and a horizontal line spanning over the display surface or the workstation viewport intersecting at the current position,
3 tracking cross : The current position is indicated by a tracking cross,
4 rubber band : The current position is indicated by a rubber band line connecting the initial position and the current position,
5 Rectangle : The current position is indicated by a rectangle whose diagonal is the line connecting the initial position and the current position,
6 Digital representation of the current position within the echo area for the LOCATOR device dependent coordinate,
≥ 7 Reserved for future standardization.

STROKE DEVICE :
The initial values are :
- The number of points in the initial stroke (n) (I)
- The sequence of points in the initial stroke (nxP)

The data record consists of :
- Input buffer size (expressed in number of points) (I)
- Initial buffer editing position (I)

The initial stroke, prompt and echo types, echo area and stroke data record are loaded into 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 replaced in the stored data record.

If the initial stroke is longer than the buffer size, the primitive is ignored. When a STROKE measure is expected, 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.

Prompt and echo types are :
< 0 prompting and echoing is STROKE device dependent,
1 implementation-dependent,
2 a digital representation of the current stroke position within the echo area,
3 a marker at each point of the current stroke,
4 a line joining successive points of the current stroke,
≥ 5 reserved for future standardization.

VALUATOR device :
The initial value is the initial VALUATOR value (R)
The data record consists of :
LOW VALUE < HIGH VALUE (2xR)

The initial value, prompt and echo types, echo area and valuator data record are loaded into 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, the low value and the 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 are :
< 0 prompting and echoing is VALUATOR device dependent,
1 implementation-dependent,
2 a graphical representation of the current VALUATOR value within the echo area, (e.g. a dial or a pointer),
3 a digital representation of the current VALUATOR value within the echo area,
≥ 4 reserved for future standardization.

CHOICE device :
The initial value is the initial CHOICE number (I)
The data record consists of one of the following possiblities according to prompt and echo type :

- prompt and echo type 2 :
  number of alternatives (n) (I)
  prompt array (one of : ON, OFF) (nxE)
- prompt and echo type 3, 4 :
  number of choice strings (n) (I)
  list of strings (nxS)
- prompt and echo type 5 :
  segment name (ID)

The initial CHOICE number, prompt and echo types, echo area and CHOICE data record are stored into the workstation state list entry for the specified CHOICE device.

Prompt and echo types are :
< 0 prompting and echoing is CHOICE device dependent,
1 implementation-dependent,
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, the primitive is ignored. The second entry in the choice data record is the "prompt array".
3 A choice number, indicating one of a set of CHOICE strings, can be selected using an appropriate technique. 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, the primitive is ignored. The second entry in the choice data record is the 'array of choice strings'.
4 A choice number, one of a set of CHOICE strings, can be selected via an alphanumeric keyboard. 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 string 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, the primitive is ignored. 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 INITIALIZE 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 area. The pick identifier 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 entry in the choice data record is the segment name.

$\geq$ 6 Reserved for future standardization.

PICK DEVICE :
The initial values are :

a) the initial status (one of : PICK, NOPICK)	(E)
b) the initial segment name	(ID)
c) the initial pick identifier	(ID)

The prompt and echo types, echo area, initial status, initial segment, initial pick identifier are loaded into the workstation state list entry for the specified PICK device.

Prompt and echo types are :

< 0 prompting and echoing is PICK device dependent,

1 Implementation dependent,

2 The echo of the contigous group of primitives within the segment with the same pick identifier as the "picked" primitive or all the primitives of the segment with the same pick ident as the "picked" primitive,

3 The echo of the whole segment containing the "picked" primitive,

$\geq$ 4 Reserved for future standardization.

STRING DEVICE :

The initial value is the initial string	(S)
The data record consists of :	
a) input buffer size (expressed in number of bytes)	(I)
b) initial cursor position	(I)

The initial string, prompt and echo types, echo area and string data record are stored into the workstation state list entry for the specified STRING device.

For all prompt and echo types, the first entry in 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. If the requested buffer size is greater, the "maximum input buffer size" is replaced in the stored data record. If the initial string is longer than the buffer size, the primitive is ignored.
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 length of initial string plus 1.

When a STRING measure is expected, a buffer of the current input buffer size is provided. 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 byte by byte. If the operator enters more characters than the current input buffer size, the additional characters are lost.

Prompt and echo types are :
< 0 prompting and echoing is STRING device dependent,
1 The current STRING value within the echo area displayed in an implementation dependent manner,
≥ 2 Reserved for future standardization.

Related primitives : None

Discussion : None

6.2.4.2 **SET LOGICAL INPUT DEVICE MODE**

Level : 0b

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number	(I)
Operating mode (one of : REQUEST, SAMPLE, EVENT)	(E)
Echo switch (one of : ECHO, NOECHO)	(E)

Description :

The given input device is set to the specified operating mode (REQUEST, SAMPLE or EVENT) 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 not.
If the operating mode is set to event, the time stamp is initialized to 0.

Related primitives : None

Discussion : None

6.2.4.3 **REQUEST LOGICAL INPUT DEVICE (REQUEST)**

Level : 0b

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number.	(I)

Description :

The primitive is used by the sender to request a measure from the given logical input device.
The receiver responds with a REQUEST LOGICAL INPUT DEVICE (RESPONSE).

Related primitives :

REQUEST LOGICAL INPUT DEVICE (RESPONSE).

Discussion : None

6.2.4.4 **REQUEST LOGICAL INPUT DEVICE (RESPONSE)**

Level : 0b

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number	(I)
Status	(E)
(one of Logical Input Device Class dependent alternatives)	
Logical input device values	
(depending on the logical input device class)	

Description :

The status and the value may vary according to the logical input class as follows :

LOCATOR device :
Status :
If the break facility is invoked by the operator, the status is returned NONE ; otherwise OK is returned together with the logical input value.

Value :
Locator position if status is OK (P)

STROKE device :
Status :
If the break facility is invoked by the operator, the status is returned NONE ; otherwise OK is returned together with the logical input value.

Value :
Number of points (n) (I)
Point list (nxP)

VALUATOR device :
Status :
If the break facility is invoked by the operator, the status is returned NONE ; otherwise OK is returned together with the logical input value.

Value :
Valuator value (R)

CHOICE device :
Status :
If the break facility is invoked by the operator, the status is returned NONE. If the measure of the CHOICE device indicates no choice, status NOCHOICE is returned otherwise OK is returned together with the logical input value.

Value :
Choice value (I)

PICK device :
Status :
If the break facility is invoked by the operator, the status is returned NONE ; if the measure of the PICK device indicates no pick status is returned NOPICK ; otherwise OK is returned together with a segment name and a pick identifier which are set according to current measure of the PICK device.

Value :
Segment name (ID)
Pick identifier (ID)

STRING DEVICE
Status :
If the break facility is invoked by the operator, the status is returned NONE ; otherwise OK is returned together with the logical input value which is the current measure of the specified STRING device.

Value :
String value (S)

Related primitives:

REQUEST LOGICAL INPUT DEVICE (REQUEST)

Discussion : None

6.2.4.5 **SAMPLE LOGICAL INPUT DEVICE (REQUEST)**

Level : 0c

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number	(I)

Description :

The primitive is issued by the sender to acquire the current measure from the given logical input device.
The receiver responds with a SAMPLE LOGICAL INPUT DEVICE (RESPONSE).

Related primitives :

SAMPLE LOGICAL INPUT DEVICE (RESPONSE)

Discussion : None

6.2.4.6 **SAMPLE LOGICAL INPUT DEVICE (RESPONSE)**

Levels : 0c

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number	(I)
Logical input device values	
(depending on the logical input device class)	

Description :

The value may vary according to the logical input class as follows :

LOCATOR device

Value:
Locator position if status is OK (P)

STROKE device

Value :

Number of points	(I)
List of points	(nxP)

VALUATOR device

Value:
Valuator value (R)

CHOICE device

Status :
If the current measure of the specific CHOICE device is indicating no choice, status NOCHOICE is returned ; otherwise OK is returned together with a choice number which is according to the current measure of the CHOICE device.

Value :
Choice value (I)

PICK device

Value:
Segment name (ID)
Pick identifier (ID)

STRING device

Value :
String value (S)

Related primitives :

SAMPLE LOGICAL INPUT DEVICE (RESPONSE)

Discussion : None

6.2.4.7 AWAIT EVENT (REQUEST)

Level : 0c

Parameters :

Timeout (R)

Description :

If the input queue in the receiver is empty, the primitive is not served until an input event is written into the queue or the time specified in the timeout parameter has elapsed whichever occurs first. If the input queue in the receiver in not empty or after the expiration of the timeout, the receiver responds with a AWAIT EVENT (RESPONSE).

The timeout is expressed in seconds.

Related primitives :

AWAIT EVENT (RESPONSE)

Discussion : None

6.2.4.8 AWAIT EVENT (RESPONSE)

Level : 0c

Parameters :

Status (one of : NONE, NOTEMPTY, OVERFLOW) (E)
If status is one of (NOTEMPTY, OVERFLOW) then
Workstation identifier (ID)
Logical input device class (E)
(One of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
Logical input device number (I)

Description :

This primitive is used by the physical device to inform its peer entity that either the timeout has elapsed and the input queue is still empty (Status NONE) or there is at least one entry in the input queue Status NOTEMPTY, OVERFLOW).
In the latter case if Status is OVERFLOW, the identification of the logical input device that caused the overflow is returned while if Status is NOTEMPTY, the content of the oldest entry is removed from the queue and made available in the current event report.

Related primitives :

AWAIT EVENT (REQUEST)

Discussion :

A timeout of zero in AWAIT EVENT (REQUEST) causes an immediate inspection of the queue and an immediate response.

6.2.4.9 GET LOGICAL INPUT DEVICE (REQUEST)

Level : 0c

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATION, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number	(I)

Description :

The primitive is used by the sender to request the logical input value, which is the current measure of the specified device.

Related primitives :

GET LOGICAL INPUT DEVICE (RESPONSE)

Discussion : None

6.2.4.10 GET LOGICAL INPUT DEVICE (RESPONSE)

Level : 0c

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number	(I)
Logical input device values	
(depend on the logical input device class)	
Time stamp	(R)

Description :

The value contained in the current event report can vary according to the logical input class parameter as follows :

LOCATOR device :
Value : (P)
Locator position

STROKE device :
Value :
Number of points (I)
List of points (nxP)

VALUATOR device :
Value :
Valuator value : (R)

CHOICE device :
Value :
Choice value (I)

PICK device :
Value :
Status (E)
Segment name (ID)
Pick identifier (ID)

If the current measure of the specified PICK device is indicating no pick, status is returned NOPICK ; otherwise OK will be returned together with the segment name and the pick identifier associated with the protocol element within the segment, that was picked.

STRING device
Value :
String value (S)

Related primitives :

GET LOGICAL INPUT DEVICE (REQUEST)

Discussion : None

6.2.4.11 FLUSH DEVICE EVENTS

Levels : 0c

Parameters :

Workstation identifier (ID)
Logical input device class (E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
Logical input device number (I)

Description :

All the entries in the input queue from the specified logical input device are removed.

Related primitives : None

Discussion : None

6.2.5 INQUIRE PRIMITIVES

6.2.5.1 INQUIRE PRIMITIVES FOR WORKSTATION STATE LIST

6.2.5.1.1 INQUIRE WORKSTATION STATE (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The state of the selected workstation is requested.

Related primitives :

INQUIRE WORKSTATION STATE (RESPONSE)

Discussion : None

6.2.5.1.2 INQUIRE WORKSTATION STATE (RESPONSE)

Level : 0a

Parameters :

Workstation state (one of : OPEN, CLOSED, ACTIVE) (E)

Description :

The workstation state is returned.

Related primitives :

INQUIRE WORKSTATION STATE (REQUEST)

Discussion : None

6.2.5.1.3 INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The selected workstation deferral and update states are requested.

Related primitives :

INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (RESPONSE)

Discussion : None

6.2.5.1.4 **INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (RESPONSE)**

Level : 0a

Parameters :

Deferral mode (one of : ASAP, BNIG, BNIL, ASTI)	(E)
Implicit regeneration flag (one of : SUPPRESSED, ALLOWED)	(E)
Display surface empty (one of : EMPTY, NOTEMPTY)	(E)
New frame action necessary at update (one of : NO, YES)	(E)

Description :

The deferral mode, the implicit regeneration flag, the display surface empty condition and the new frame action condition are returned.

Related primitives :

INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (REQUEST)

Discussion : None

6.2.5.1.5 **INQUIRE POLYLINE INDICES (REQUEST)**

Level : 1a

Parameters :

Workstation identifier	(ID)

Description :

The number of indices in the selected workstation polyline bundle table is requested.

Related primitives :

INQUIRE POLYLINE INDICES (RESPONSE)

Discussion : None

6.2.5.1.6 **INQUIRE POLYLINE INDICES (RESPONSE)**

Level : 1a

Parameters :

Number of polyline bundle table entries	(I)
Number of defined indices	(I)

Description :

The number of polyline bundle table entries and the number of defined indices are returned.

Related primitives :

INQUIRE POLYLINE INDICES (REQUEST)

Discussion : None

6.2.5.1.7 **INQUIRE POLYLINE REPRESENTATION (REQUEST)**

Level : 1a

Parameters :

Workstation identifier	(ID)
Polyline Index (1..n)	(IX)

Description :

The polyline representation according to the polyline index is requested for the selected workstation.

Related primitives :

INQUIRE POLYLINE REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.1.8 **INQUIRE POLYLINE REPRESENTATION (RESPONSE)**

Level : 1a

Parameters :

Line type	(I)
Line width scale factor	(R)
Polyline colour index	(CI)

Description :

The polyline representation consisting of the effectively realized line type, line width scale factor and polyline colour index is returned.

Related primitives :

INQUIRE POLYLINE REPRESENTATION (REQUEST)

Discussion : None

6.2.5.1.9 INQUIRE POLYMARKER INDICES (REQUEST)

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

The number of indices in the selected workstation polymarker bundle table is requested.

Related primitives :

INQUIRE POLYMARKER INDICES (RESPONSE)

Discussion : None

6.2.5.1.10 INQUIRE POLYMARKER INDICES (RESPONSE)

Level : 1a

Parameters :

Number of polymarker bundle table entries (I)
Number of defined indices (I)

Description :

The number of polymarker bundle table entries and the number of defined indices are returned.

Related primitives :

INQUIRE POLYMARKER INDICES (REQUEST)

Discussion : None

6.2.5.1.11 INQUIRE POLYMARKER REPRESENTATION (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)
Polymarker index (1..n) (IX)

Description :

The polymarker representation according to the polymarker index for the selected workstation is requested.

Related primitives :

INQUIRE POLYMARKER REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.1.12 INQUIRE POLYMARKER REPRESENTATION (RESPONSE)

Level : 1a

Parameters :

Marker type	(I)
Marker size scale factor	(R)
Polymarker colour index	(CI)

Description :

The polymarker representation consisting of the effectively realized marker type, marker size scale factor and polymarker colour index is returned.

Related primitives :

INQUIRE POLYMARKER REPRESENTATION (REQUEST)

Discussion : None

6.2.5.1.13 INQUIRE TEXT INDICES (REQUEST)

Level : 1a

Parameters :

Workstation identifier	(ID)

Description :

The number of indices in the selected workstation text bundle table is requested.

Related primitives :

INQUIRE TEXT INDICES (RESPONSE)

Discussion : None

6.2.5.1.14 INQUIRE TEXT INDICES (RESPONSE)

Level : 1a

Parameters :

Number of text bundle table entries	(I)
Number of defined indices	(I)

Description :

The number of text bundle table entries and the number of defined entries are returned.

Related primitives :

INQUIRE TEXT INDICES (REQUEST)

Discussion : None

6.2.5.1.15 **TEXT REPRESENTATION (REQUEST)**

Level : 1a

Parameters :

Workstation identifier	(ID)
Text index (1..n)	(IX)

Description :

The text representation according to the text index for the selected workstation is requested.

Related primitives :

INQUIRE TEXT REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.1.16 **INQUIRE TEXT REPRESENTATION (RESPONSE)**

Level : 1a

Parameters :

Text font	(I)
Text precision (one of : STRING, CHARACTER, STROKE)	(E)
Character expansion factor	(R)
Character spacing	(R)
Text colour index	(CI)

Description :

The text representation consisting of the effectively realized text font, text precision, character expansion factor, character spacing and text colour index is returned.

Related primitives :

INQUIRE TEXT REPRESENTATION (REQUEST)

Discussion : None

6.2.5.1.17 **INQUIRE TEXT EXTENT (REQUEST)**

Level : 0a

Parameters :

Workstation identifier	(ID)
Text position	(P)
Character string	(S)

Description :

The extent of the specified character string with the specified text position is requested.

Related primitives :

INQUIRE TEXT EXTENT (RESPONSE)

Discussion :

The currently selected text font, text precision character expansion factor and character spacing and the current text attributes (character height, character up vector, text path, text alignment) together with the text position parameter are used for computing the text extent parallelogram for the specified string.

6.2.5.1.18 **INQUIRE TEXT EXTENT (RESPONSE)**

Level : 0a

Parameters :

Text extent parallelogram	(3xP)

Description :

The text extent parallelogram is returned. The parallelogram is defined in the same way as the CELL ARRAY parallelogram.

Related primitives :

INQUIRE TEXT EXTENT (REQUEST)

Discussion : None

6.2.5.1.19 **INQUIRE FILL AREA INDICES (REQUEST)**

Level : 1a

Parameters :

Workstation identifier	(ID)

Description :

The number of indices in the fill area bundle table of the selected workstation is requested.

Related primitives :

INQUIRE FILL AREA INDICES (RESPONSE)

Discussion : None

6.2.5.1.20 **INQUIRE FILL AREA INDICES (RESPONSE)**

Level : 1a

Parameters :

Number of fill area bundle table entries	(I)
Number of defined indices	(I)

Description :

The number of fill area bundle table entries and the number of defined indices are returned.

Related primitives :

INQUIRE FILL AREA INDICES (REQUEST)

Discussion : None

6.2.5.1.21 **INQUIRE FILL AREA REPRESENTATION (REQUEST)**

Level : 1a

Parameters :

Workstation identifier	(ID)
Fill area index (1..n)	(IX)

Description :

The fill area representation according to the fill area index for the selected workstation is requested.

Related primitives :

INQUIRE FILL AREA REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.1.22 **INQUIRE FILL AREA REPRESENTATION (RESPONSE)**

Level : 1a

Parameters :

Fill area interior style (one of: HOLLOW, SOLID, PATTERN, HATCH) (E)
Hatch type or pattern index (I or IX)
Fill area colour index (CI)

Description :

The fill area representation consisting of the effectively realized fill area interior style, fill area style index, fill area colour index is returned.

Related primitives :

INQUIRE FILL AREA REPRESENTATION (REQUEST)

Discussion : None

6.2.5.1.23 **INQUIRE PATTERN INDICES (REQUEST)**

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

The number of indices in the pattern table of the selected workstation is requested.

Related primitives :

INQUIRE PATTERN INDICES (RESPONSE)

Discussion : None

6.2.5.1.24 **INQUIRE PATTERN INDICES (RESPONSE)**

Level : 1a

Parameters :

Number of pattern table entries (I)
Number of defined indices (I)

Description :

The number of pattern table entries and the number of defined indices are returned.

Related primitives :

INQUIRE PATTERN INDICES (REQUEST)

Discussion : None

6.2.5.1.25 INQUIRE PATTERN REPRESENTATION (REQUEST)

Level : 1a

Parameters :

Workstation identifier	(ID)
Pattern index (1..n)	(IX)

Description :

The pattern representation according to the pattern index for the selecter workstation is requested.

Related primitives :

INQUIRE PATTERN REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.1.26 INQUIRE PATTERN REPRESENTATION (RESPONSE)

Level : 1a

Parameters :

Delta X (DX)	(I)
Delta Y (DY)	(I)
Pattern cell colour index list	(CL)

Description :

The pattern representation consisting of the effectively realized pattern array is returned.

Related primitives :

INQUIRE PATTERN REPRESENTATION (REQUEST)

Discussion : None

6.2.5.1.27 INQUIRE COLOUR INDICES (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)

Description :

The number of indices in the colour table of the selected workstation is requested.

Related primitives :

INQUIRE COLOUR INDICES (RESPONSE)

Discussion : None

6.2.5.1.28 INQUIRE COLOUR INDICES (RESPONSE)

Level : 0a

Parameters :

Number of colour table entries	(I)
Number of defined indices	(I)

Description :

The number of colour table entries and the number of defined indices are returned.

Related primitives :

INQUIRE COLOUR INDICES (REQUEST)

Discussion : None

6.2.5.1.29 INQUIRE COLOUR REPRESENTATION (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)
Colour table index	(CI)

Description :

The colour representation according to the colour index for the selected workstation is requested.

Related primitives :

INQUIRE COLOUR REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.1.30 INQUIRE COLOUR REPRESENTATION (RESPONSE)

Level : 0a

Parameters :

Colour (red, green, blue intensities)	(CD)

Description :

The colour representation consisting of the effectively realized colour is returned.

Related primitives :

INQUIRE COLOUR REPRESENTATION (REQUEST)

Discussion : None

6.2.5.1.31 INQUIRE WORKSTATION TRANSFORMATION (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The workstation transformation update state, the current and requested workstation windows and viewports are requested.

Related primitives :

INQUIRE WORKSTATION TRANSFORMATION (RESPONSE)

Discussion : None

6.2.5.1.32 INQUIRE WORKSTATION TRANSFORMATION (RESPONSE)

Level : 0a

Parameters :

Workstation transformation update state (one of : NOTPENDING, PENDING)	(E)
If update state is PENDING :	
Requested workstation window limits	(2xP)
Requested workstation viewport limits	(4xR)
Current workstation window limits	(2xP)
Current workstation viewport limits	(4xR)
If update state is NOTPENDING :	
Current workstation window limits	(2xP)
Current workstation viewport limits	(4xR)

Description :

The workstation transformation update state, the requested and/or the current workstation window and workstation viewport are returned.

Related primitives :

INQUIRE WORKSTATION TRANSFORMATION (REQUEST)

Discussion : None

6.2.5.1.33 INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (REQUEST)

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

The number and the sequence of segment names associated to the specified workstation are requested.

Related primitives :

INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (RESPONSE)

Discussion : None

6.2.5.1.34 INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (RESPONSE)

Level : 1a

Parameters :

Number of segment names (n) (I)
Name of stored segments (nxID)

Description :

The number and the sequence of segments within a given workstation are returned.

Related primitives :

INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (REQUEST)

Discussion : None

6.2.5.1.35 INQUIRE LOGICAL INPUT DEVICE STATE (REQUEST)

Level : 0b

Parameters :

Workstation identifier (ID)
Logical device class (E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)
Logical device number (I)

Description :

The logical input device state of the given class and number, associated to the selected workstation is requested.

Related primitives :

INQUIRE LOGICAL INPUT DEVICE STATE (RESPONSE)

Discussion : None

6.2.5.1.36 INQUIRE LOGICAL INPUT DEVICE STATE (RESPONSE)

Level : 0b

Parameters :

Operating mode (one of : REQUEST, SAMPLE, EVENT) (E)
Echo switch : (one of : ECHO, NOECHO) (E)
Prompt and echo type : (I)
Echo area (lower left, upper right corners) (4xR)

If the logical input device is LOCATOR then
Initial value (initial position) (P)

If the logical input device is STROKE then

Initial value
- initial number of points (n) (I)
- initial points (nxP)

Data record
- buffer size (I)
- buffer editing position (I)

If the logical input device is VALUATOR then initial value
Initial valuator (R)

Data record
- low value (R)
- high value (R)

If the logical input device is CHOICE then initial value
Initial choice number (I)

If the prompt and echo type = 2 then

Data record
- number of choice alternatives (I)
- prompt array (one of : ON, OFF) (E)

If the prompt and echo type = 3, 4 then

Data record { number of choice strings (n) (I)
sequence of strings (nxS)

If the prompt and echo type = 5 then
Data record (segment name) (ID)

If the logical input device is PICK then

Initial value { initial status (one of : PICK, NOPICK) (E)
initial segment name (ID)
initial pick identifier (ID)

If the logical input device is STRING then

Initial value (initial string) (S)

Data record { initial buffer size (I)
initial cursor position (I)

Description :

The logical input device state consisting of the operating mode, the echo switch, the prompt and echo type, the echo area, the initial value and the data record is returned.

Related primitive :

INQUIRE LOGICAL INPUT DEVICE STATE (REQUEST)

Discussion :

The initial value and the data record vary according to the logical input device class. Different data syntaxes are provided in order to optimize the number of information the protocol must handle.

6.2.5.2 INQUIRE PRIMITIVES FOR WORKSTATION DESCRIPTION TABLE

6.2.5.2.1 INQUIRE WORKSTATION CATEGORY (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The workstation category of the specified workstation is requested.

Related primitives :

INQUIRE WORKSTATION CATEGORY (RESPONSE)

Discussion : None

6.2.5.2.2 **INQUIRE WORKSTATION CATEGORY (RESPONSE)**

Level : 0a

Parameters :

Workstation category (one of : OUTPUT, INPUT, OUTIN, WISS) (E)

Description :

The workstation category to the selected workstation is returned.

Related primitives :

INQUIRE WORKSTATION CATEGORY (REQUEST)

Discussion : None

6.2.5.2.3 **INQUIRE WORKSTATION CLASSIFICATION (REQUEST)**

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The workstation classification of the specified workstation is requested.

Related primitives :

INQUIRE WORKSTATION CLASSIFICATION (RESPONSE)

Discussion : None

6.2.5.2.4 **INQUIRE WORKSTATION CLASSIFICATION (RESPONSE)**

Level : 0a

Parameters :

Workstation classification (I)

Description :

The workstation classification is returned.

Related primitives :

INQUIRE WORKSTATION CLASSIFICATION (REQUEST)

Discussion :

The following types are defined :

Integer = 0 : vector workstation
Integer = 1 : raster workstation
Integer > 1 : reserved for future standardization.

6.2.5.2.5 INQUIRE MAXIMUM DISPLAY SURFACE SIZE (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The maximum display surface size of the specified workstation is requested.

Related primitives :

INQUIRE MAXIMUM DISPLAY SURFACE SIZE (RESPONSE)

Discussion : None

6.2.5.2.6 INQUIRE MAXIMUM DISPLAY SURFACE SIZE (RESPONSE)

Level : 0a

Parameters :

Maximum display surface size in meters (horizontal, vertical) (2xR)
Maximum display surface size in raster units (horizontal, vertical) (2xI)

Description :

The maximum display surface size expressed in meters and raster units is returned.

Related primitives :

INQUIRE MAXIMUM DISPLAY SURFACE SIZE (REQUEST)

Discussion :

If the display is not a raster display, then the maximum display surface size in raster units parameters will be zero.

6.2.5.2.7 INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (REQUEST)

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

The dynamic modification capability for the workstation attributes of the specified workstation is requested.

Related primitives :

INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (RESPONSE)

Discussion : None

6.2.5.2.8 INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (RESPONSE)

Level : 1a

Parameters :

Polyline bundle representation changeable (one of : IRG, IMM)	(E)
Polymarker bundle representation changeable (one of : IRG,IMM)	(E)
Text bundle representation changeable (one of : IRG, IMM)	(E)
Fill area bundle representation changeable (one of : IRG, IMM)	(E)
Pattern representation changeable (one of : IRG, IMM)	(E)
Colour representation changeable (one of : IRG, IMM)	(E)
Workstation transformation changeable (one of : IRG, IMM)	(E)

Description :

The dynamic modification capability for polyline, polymarker, text, fill area bundle representation, pattern and colour representation and workstation transformation are returned.

Related primitives :

INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (REQUEST)

Discussion :

IRG means that the implicit regeneration mode is set ; IMM means that the immediate mode is set ; in the latter case the action is performed immediately.

6.2.5.2.9 INQUIRE DEFAULT DEFERRAL STATE VALUES (REQUEST)

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

The default deferral state values for the specified workstation is requested.

Related primitives :

INQUIRE DEFAULT DEFERRAL STATE VALUES (RESPONSE)

Discussion : None

6.2.5.2.10 INQUIRE DEFAULT DEFERRAL STATE VALUES (RESPONSE)

Level : 1a

Parameters :

Default value for deferral mode
(one of : ASAP, BNIG, BNIL, ASTI) (E)
Default value for implicit regeneration flag (E)
(one of : SUPPRESSED, ALLOWED)

Description :

The default values for deferral mode and implicit regeneration flag are returned.

Related primitives :

INQUIRE DEFAULT DEFERRAL STATE VALUES (REQUEST)

Discussion : None

6.2.5.2.11 INQUIRE POLYLINE FACILITIES (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The polyline facilities for the specified workstation are requested.

Related primitives :

INQUIRE POLYLINE FACILITIES (RESPONSE)

Discussion : None

6.2.5.2.12 INQUIRE POLYLINE FACILITIES (RESPONSE)

Level : 0a

Parameters :

Number of available line types (n)	(I)
Sequence of available line types	(nxI)
Number of available line widths	(I)
Nominal line width	(R)
Range of line widths (minimum, maximum)	(2xR)
Number of predefined polyline indices	(I)

Description :

The polyline facilities consisting of the number of available line types and line widths, the number of predefined polyline indices, the sequence of available line types and the range of line widths are returned.

Related primitives :

INQUIRE POLYLINE FACILITIES (REQUEST)

Discussion :

If the number of available line widths is returned as 0, the workstation supports a continuous range of line widths.

6.2.5.2.13 INQUIRE PREDEFINED POLYLINE REPRESENTATION (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)
Predefined polyline index (1..n)	(IX)

Description :

The polyline representation of the specified index for the specified workstation is requested.

Related primitives :

INQUIRE PREDEFINE POLYLINE REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.2.14 INQUIRE PREDEFINED POLYLINE REPRESENTATION (RESPONSE)

Level : 0a

Parameters :

Line type	(I)
Line width scale factor	(R)
Polyline colour index	(CI)

Description :

The selected predefined polyline representation consisting of the line type, the line width and the polyline colour index, is returned.

Related primitives :

INQUIRE PREDEFINED POLYLINE REPRESENTATION (REQUEST)

Discussion : None

6.2.5.2.15 INQUIRE POLYMARKER FACILITIES (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)

Description :

The polymarker facilities for the specified workstation are requested.

Related primitives :

INQUIRE POLYMARKER FACILITIES (RESPONSE)

Discussion : None

6.2.5.2.16 INQUIRE POLYMARKER FACILITIES (RESPONSE)

Level : 0a

Parameters :

Number of available marker types	(I)
Sequence of available marker types	(nxI)
Number of available marker sizes	(I)
Nominal marker size	(R)
Range of marker sizes (minimum, maximum)	(2xR)
Number of predefined polymarker indices	(I)

Description :

The number of available marker types and sizes, the sequence of available marker types, the nominal marker size, the range of marker sizes and the number of predefined polymarker indices are returned.

Related primitives :

INQUIRE POLYMARKER FACILITIES (REQUEST)

Discussion :

If the number of available marker sizes is returned as 0, the workstation supports a continuous range of marker sizes.

6.2.5.2.17 INQUIRE PREDEFINED POLYMARKER REPRESENTATION (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)
Predefined polymarker index (1..n)	(IX)

Description :

The polymarker representation of the specified index for the specified workstation is requested.

Related primitives :

INQUIRE POLYMARKER REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.2.18 INQUIRE PREDEFINED POLYMARKER REPRESENTATION (RESPONSE)

Level : 0a

Parameters :

Marker type	(I)
Marker size scale factor	(R)
Polymarker colour index	(CI)

Description :

The selected predefined polymarker representation, consisting of the marker type, the marker size and the polymarker colour index, is returned.

Related primitives :

INQUIRE PREDEFINED POLYMARKER REPRESENTATION (REQUEST)

Discussion : None

6.2.5.2.19 INQUIRE TEXT FACILITIES (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The text facilities for the specified workstation are requested.

Related primitives :

INQUIRE TEXT FACILITIES (RESPONSE)

Discussion : None

6.2.5.2.20 INQUIRE TEXT FACILITIES (RESPONSE)

Level : 0a

Parameters :

Number of available text font and precision pairs	(I)
Sequence of available text font and precision pairs	[nx(I,E)]
Number of available character heights	(I)
Minimum and maximum characters heights	(2xV)
Number of available character expansion factors	(I)
Minimum and maximum character expansion factors	(2xR)
Number of predefined text indices	(I)

Description :

The text facilities consisting of number and sequence of available text font and precision pairs, number of available character heights and expansion factors (specifying the minimum and the maximum), and number of predefined text indices are returned.

If the available character heigths and character expansion factors vary between fonts, the character heights and character expansion factors returned are for font 0.

Related primitives :

INQUIRE TEXT FACILITIES (REQUEST)

Discussion :

For every available font the number of available text font and precision pairs parameter must explicitely include the available precisions.

6.2.5.2.21 INQUIRE PREDEFINED TEXT REPRESENTATION (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)
Predefined text index (1..n)	(IX)

Description :

The text representation of the specified index for the specified workstation is requested.

Related primitives :

INQUIRE PREDEFINED TEXT REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.2.22 INQUIRE PREDEFINED TEXT REPRESENTATION (RESPONSE)

Level : 0a

Parameters :

Text font	(I)
Text precision (one of : STRING, CHARACTER, STROKE)	(E)
Character expansion factor	(R)
Character spacing	(R)
Text colour index	(CI)

Description :

The selected predefined text representation, consisting of the text font and precision, the character spacing and expansion factor, and the text colour index, is returned.

Related primitives :

INQUIRE PREDEFINED TEXT REPRESENTATION (REQUEST)

Discussion : None

6.2.5.2.23 INQUIRE FILL AREA FACILITIES (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)

Description :

The fill area facilities for the specified workstation are requested.

Related primitives :

INQUIRE FILL AREA FACILITIES (RESPONSE)

Discussion : None

6.2.5.2.24 INQUIRE FILL AREA FACILITIES (RESPONSE)

Level : 0a

Parameters :

Number of available fill area interior styles	(I)
Sequence of available fill area interior styles	(nxE)
Number of available hatch types	(I)
Sequence of available hatch types	(nxI)
Number of predefined fill area indices	(I)

Description :

The number and the sequence of available fill area interior styles and hatch types and the number of predefined fill area indices are returned.

Related primitives :

INQUIRE FILL AREA FACILITIES (REQUEST)

Discussion : None

6.2.5.2.25 INQUIRE PREDEFINED FILL AREA REPRESENTATION (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)
Predefined fill area index (1..n)	(IX)

Description :

The fill area representation of the specified index for the specified workstation are requested.

Related primitives :

INQUIRE PREDEFINED FILL AREA REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.2.26 INQUIRE PREDEFINED FILL AREA REPRESENTATION (RESPONSE)

Level : 0a

Parameters :

Fill area interior style (one of : HOLLOW, SOLID, PATTERN, HATCH)	(E)
Hatch type or pattern index	(I or IX)
Fill area colour index	(CI)

Description :

The selected predefined fill area representation, consisting of the fill area interior style, the fill area style index and the fill area colour index, is returned.

Related primitives :

INQUIRE PREDEFINED FILL AREA REPRESENTATION (REQUEST)

Discussion : None

6.2.5.2.27 **INQUIRE PATTERN FACILITIES (REQUEST)**

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The pattern facilities for the specified workstation are requested.

Related primitives :

INQUIRE PATTERN FACILITIES (RESPONSE)

Discussion : None

6.2.5.2.28 **INQUIRE PATTERN FACILITIES (RESPONSE)**

Level : 0a

Parameters :

Number of predefined pattern indices (I)

Description :

The number of predefined pattern indices is returned.

Related primitives :

INQUIRE PATTERN FACILITIES (REQUEST)

Discussion : None

6.2.5.2.29 **INQUIRE PREDEFINED PATTERN REPRESENTATION (REQUEST)**

Level : 0a

Parameters :

Workstation identifier (ID)
Predefined pattern index (I)

Description :

The pattern representation of the specified index for the specified workstation is requested.

Related primitives :

INQUIRE PREDEFINED PATTERN REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.2.30 INQUIRE PREDEFINED PATTERN REPRESENTATION (RESPONSE)

Level : 0a

Parameters :

Delta X (DX)	(I)
Delta Y (DY)	(I)
Pattern cell colour	(CL)

Description :

The selected predefined pattern representation, consisting of the patern array dimensions and the pattern array itself, is returned.

Related primitives :

INQUIRE PREDEFINED PATTERN REPRESENTATION (REQUEST)

Discussion : None

6.2.5.2.31 INQUIRE COLOUR FACILITIES (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)

Description :

The colour facilities for the specified workstation are requested.

Related primitives :

INQUIRE COLOUR FACILITIES (RESPONSE)

Discussion : None

6.2.5.2.32 INQUIRE COLOUR FACILITIES (RESPONSE)

Level : 0a

Parameters :

Number of available colours	(I)
Colour available (one of : COLOUR, MONOCHROME)	(E)
Number of predefined colour indices	(I)

Description :

The colour facilities, number of available colours, availability of colour and number of predefined colour indices are returned.

Related primitives :

INQUIRE COLOUR FACILITIES (REQUEST)

Discussion : None

6.2.5.2.33 **INQUIRE PREDEFINED COLOUR REPRESENTATION (REQUEST)**

Level : 0a

Parameters :

Workstation identifier	(ID)
Colour table index	(CI)

Description :

The colour representation of the specified index for the specified workstation is requested.

Related primitives :

INQUIRE PREDEFINED COLOUR REPRESENTATION (RESPONSE)

Discussion : None

6.2.5.2.34 **INQUIRE PREDEFINED COLOUR REPRESENTATION (RESPONSE)**

Level : 0a

Parameters :

Colour (RED, GREEN, BLUE intensities)	(CD)

Description :

The selected predefined colour representation is returned.

Related primitives :

INQUIRE PREDEFINED COLOUR REPRESENTATION (REQUEST)

Discussion : None

6.2.5.2.35 INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)

Description :

The list of the generalized drawing primitives available on the specified workstation is requested.

Related primitives :

INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (RESPONSE)

Discussion : None

6.2.5.2.36 INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (RESPONSE)

Level : 0a

Parameters :

Number of available generalized drawing primitives (I)
List of GDPs (nxI)

Description :

The number of available GDPs and their list are returned.

Related primitives :

INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (REQUEST)

Discussion :

A non-negative integer is conventionally associated to each GDP as follows :

< 0	Private GDP	6	CIRCULAR ARC CENTRE CHORD
0	RECTANGLE	7	CIRCULAR ARC CENTRE PIE
1	CIRCLE	8	ELLIPSE
2	CIRCULAR ARC 3 POINT	9	ELLIPTIC ARC
3	CIRCULAR ARC 3 POINT CHORD	10	ELLIPTIC ARC CHORD
4	CIRCULAR ARC 3 POINT PIE	11	ELLIPTIC ARC PIE
5	CIRCULAR ARC CENTRE	12	SPLINE
		>12	RESERVED

6.2.5.2.37 INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (REQUEST)

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

The maximum length of the workstation state tables for the specified workstation is requested.

Related primitives :

INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (RESPONSE)

Discussion : None

6.2.5.2.38 INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (RESPONSE)

Level : 1a

Parameters :

Maximum number of polyline bundle table entries	(I)
Maximum number of polymarker bundle table entries	(I)
Maximum number of text bundle table entries	(I)
Maximum number of fill area bundle table entries	(I)
Maximum number of pattern table entries	(I)
Maximum number of colour table entries	(I)

Description :

The maximum number of polyline, polymarker, text and fill area bundle table entries and of pattern and colour table entries are returned.

Related primitives :

INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (REQUEST)

Discussion : None

6.2.5.2.39 INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES (REQUEST)

Level : 0b

Parameters :

Workstation identifier	(ID)

Description :

The number of logical input devices available for the specified workstation is requested.

Related primitives :

INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES (RESPONSE)

Discussion : None

6.2.5.2.40 INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES (RESPONSE)

Level : 0b

Parameters :

Number of locator devices	(I)
Number of stroke devices	(I)
Number of valuator devices	(I)
Number of choice devices	(I)
Number of pick devices	(I)
Number of string devices	(I)

Description :

The number of logical input devices for each class is returned.

Related primitives :

INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES (REQUEST)

Discussion : None

6.2.5.2.41 INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (REQUEST)

Level : 0b

Parameters :

Workstation identifier	(ID)
Logical input device class	(E)
(one of : LOCATOR, STROKE, VALUATOR, CHOICE, PICK, STRING)	
Logical input device number	(I)

Description :

The default device data for the specified logical input device number of the specified workstation is requested.

Related primitives :

INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (RESPONSE)

Discussion : None

6.2.5.2.42 INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (RESPONSE)

Level : 0a

Parameters :

Number of available prompt and echotypes	(I)
Sequence of available prompt and echo types	(nxI)
Default echo area	(4xR)

If logical input device is LOCATOR then :
Default initial value (initial locator position) (P)
If logical input device is STROKE then :

Default initial value (maximum input buffer size) (I)

Default data record { maximum buffer size (I)
first buffer editing position (I)

If logical input device is VALUATOR then :
Default initial value (default valuator initial value) (R)
Default data record { Default low value (R)
Default high value (R)

If logical input device is CHOICE then :
Default prompt and echo type (default choice prompt and echo type) (I)
Default initial value choice number (I)

If the default prompt and echo type = 2 then :
Default choice data record { maximum choice number (I)
default prompt array (E)
(one of : ON, OFF)

If the default prompt and echo type = 3,4 then :
The default choice data record { maximum number of choice strings (I)
default sequence of strings (nxS)

If the default prompt and echo type = 5 then :
The default choice data record (default segment name) (ID)

If logical input device is PICK then :
Default initial value { default status (E)
(one of : PICK, NOPICK)
default segment name (ID)
default pick identifier (ID)

If logical input device is STRING then :
Default data record { maximum buffer size (I)
first cursor position (I)

Description :

The default logical input device data consisting of the number and sequence of available prompt and echo types, echo area, initial value and data record are returned.

Discussion :

Most of the default logical input device data are implementation dependent.

6.2.5.3 INQUIRE PRIMITIVES FOR SEGMENT STATE LIST

6.2.5.3.1 INQUIRE SET OF ASSOCIATED WORKSTATION (REQUEST)

Level : 1a

Parameters :

Segment name (ID)

Description :

The number and the set of workstation names associated to the specified segment are requested.

Related primitives :

INQUIRE SET OF ASSOCIATED WORKSTATIONS (RESPONSE)

Discussion : None

6.2.5.3.2 INQUIRE SET OF ASSOCIATED WORKSTATION (RESPONSE)

Level : 1a

Parameters :

Number of associated workstations (I)
Set of associated workstation identifiers (nxID)

Description :

The number and the set of workstation identifiers are returned.

Related primitives :

INQUIRE SET OF ASSOCIATED WORKSTATIONS (REQUEST)

Discussion : None

6.2.5.3.3 INQUIRE SEGMENT ATTRIBUTES (REQUEST)

Level : 1a

Parameters :

Segment name (ID)

Description :

The segment attributes consisting of transformation matrix, visibility, highlighting, priority and detectability are requested for the specified segment.

Related primitives :

INQUIRE SEGMENT ATTRIBUTES (RESPONSE)

Discussion : None

6.2.5.3.4 INQUIRE SEGMENT ATTRIBUTES (RESPONSE)

Level : 1a

Parameters :

Segment transformation matrix	(M)
Visibility (one of : VISIBLE, INVISIBLE)	(E)
Highlighting (one of : NOTHIGHLIGHTED, HIGHLIGHTED)	(E)
Segment priority [0,1]	(R)
Detectability (one of : UNDETECTABLE, DETECTABLE)	(E)

Description :

The segment attributes consisting of transformation matrix, visibility, highlighting, priority and detectability are returned.

Related primitives :

INQUIRE SEGMENT ATTRIBUTES (REQUEST)

Discussion : None

6.2.5.3.5 INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTE (REQUEST)

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

The dynamic modification of segments attributes capability for the specified workstation is requested.

Related primitives :

INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTE (RESPONSE)

Discussion : None

6.2.5.3.6 INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTE (RESPONSE)

Level : 1a

Parameters :

Segment transformation changeable (one of IRG, IMM)	(E)
Visibility changeable from 'visible' to 'invisible' (one of : IRG, IMM)	(E)
Visibility changeable from 'invisible' to 'visible' (one of : IRG, IMM)	(E)
Highlighting changeable (one of : IRG, IMM)	(E)
Segment priority changeable (one of : IRG, IMM)	(E)
Adding primitives to the open segment (one of : IRG, IMM)	(E)
Segment deletion immediately visible (one of : IRG, IMM)	(E)

Description :

The dynamic modification capability for segment transformation, visibility, highlighting priority, insertion of new primitives and immediately visible deletion is returned.

Related primitives :

INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTE (REQUEST)

Discussion : None

6.2.5.3.7 INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (REQUEST)

Level : 1a

Parameters :

Workstation identifier (ID)

Description :

The number of segment priorities supported by the specified workstation is requested.

Related primitives :

INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (RESPONSE)

Discussion : None

6.2.5.3.8 INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (RESPONSE)

Level : 1a

Parameters :

Number of segment priorities supported (I)

Description :

The number of segment priorities supported is returned.

Related primitives :

INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (REQUEST)

Discussion : None

6.2.5.4 INQUIRE PRIMITIVES FOR PIXELS

6.2.5.4.1 INQUIRE PIXEL (REQUEST)

Level : 0a

Parameters :

Workstation identifier (ID)
Pixel point P (P)

Description :

The pixel colour index of the pixel located at the point P is requested.

Related primitives :

INQUIRE PIXEL (RESPONSE)

Discussion :

This request has sense for workstation having pixel store readback capability. Workstations not able to serve this request will respond with an error response primitive.

6.2.5.4.2 INQUIRE PIXEL (RESPONSE)

Level : 0a

Parameters :

Pixel colour index (CI)

Description :

The primitive is the response to the INQUIRE PIXEL (REQUEST). The pixel colour index is the colour index of the pixel located at the point P specified in the request.

Related primitives :

INQUIRE PIXEL (REQUEST)

Discussion : None

6.2.5.4.3 INQUIRE PIXEL ARRAY (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)
Reference point P	(P)
Dimensions of pixel array (DX,DY)	(2xI)

Description :

This primitive is used by the sending entity to request the colour indexes of the rectangular array of pixels, whose upper left corner is the pixel corresponding to the point P and whose horizontal and vertical dimensions are respectively DX and DY.

Related primitives :

INQUIRE PIXEL ARRAY (RESPONSE)

Discussion :

Same as for INQUIRE PIXEL ARRAY DIMENSIONS (REQUEST).

6.2.5.4.4 INQUIRE PIXEL ARRAY (RESPONSE)

Level : 0a

Parameters :

Pixel colour index list	(CL)

Description :

This primitive is the response to the INQUIRE PIXEL ARRAY (REQUEST) primitive for the workstation able to provide this information.

Related primitives :

INQUIRE PIXEL ARRAY (REQUEST)

Discussion :

If the colour index corresponding to a particular pixel cannot be ascertained (for example, the point P is not on the display surface), an error response primitive will be issued.

6.2.5.4.5 INQUIRE PIXEL ARRAY DIMENSIONS (REQUEST)

Level : 0a

Parameters :

Workstation identifier	(ID)
Point List (P,Q)	(2xP)

Description :

This primitive in intended to obtain the dimensions in pixels of the rectangle defined by the points P and Q, which are respectively the upper left and lower right corners.

Related primitives :

INQUIRE PIXEL ARRAY DIMENSIONS (RESPONSE)

Discussion :

The request has sense for the workstations having pixel store read-back capability.
For workstations which are not able to handle this request, the response will be an error response primitive.

6.2.5.4.6 INQUIRE PIXEL ARRAY DIMENSIONS (RESPONSE)

Level : 0a

Parameters :

Dimensions of pixel array (DX,DY) (2xI)

Description :

This primitive is the response to the INQUIRE PIXEL ARRAY DIMENSIONS (REQUEST) primitive.
DX is the number of pixels along the horizontal side of the rectangle defined by the points P and Q in the corresponding request, DY being the number of pixels along the vertical side.

Related primitives :

INQUIRE PIXEL ARRAY DIMENSIONS (REQUEST)

Discussion :

For the calculation of the number of pixels DX and DY, the rectangle defined by P and Q is clipped neither against the clipping rectangle nor the workstation.

6.2.6 PROTOCOL DESCRIPTOR PRIMITIVES

6.2.6.1 SET DOMAIN RING

Level : 0a

Parameters :

Angular resolution factor

(one of : RESOLUTION-0, RESOLUTION-1, RESOLUTION-2, RESOLUTION-3)(E)

Ring Size (I)

Description :

This primitive specifies the precision of the coordinate encoding, when using incremental mode.

Related primitives : None

Discussion :

If the ring size parameter value is zero, the basic ring size according to the current granularity code (as set with the SET COORDINATE PRECISION PRIMITIVE) is set.

If the ring size parameter is negative the primitive is ignored.

6.2.6.2 SET COLOUR HEADER

Level : 0a

Parameters :

Unit resolution (1,2....9)	(I)
Coding method (only RGB-CODING)	(E)

Description :

The parameters have the following meaning :

- The first parameter specifies the number of bits used to encode the intensity of each colour component in the RGB colour definition.

- The last parameter specifies the coding method used.

Related primitives :

SET COLOUR REPRESENTATION

Discussion :

If the selected unit resolution is out of range, the default unit resolution is selected.

6.2.6.3 SET COORDINATE PRECISION

Level : 0a

Parameters :

Magnitude code	(I)
Granularity code	(I)
Default exponent	(I)
Explicit exponent allowed (one of : ALLOWED, FORBIDDEN)	(E)

Description :

The magnitude code parameter specifices the largest possible magnitude of positive or negative NDC coordinates and size values which can be encoded (the number of bits which may occur to the left of the binary radix point in the representation of a coordinate or size va-value as a binary fraction plus one bit for the sign bit).

The granularity code parameter specifies the smallest nonzero value that can be expressed with this precision. This value is called the Basic Grid Unit (BGU). It does this by specifying the smallest exponent that is permitted in the coded representation of a coordinate or a size value.

The default exponent parameter specifies the exponent value, which is used when encoding a point, in case :

- the exponent is omitted in the encoding and

- the encoded point is a single point or the first point of a point list.

It also specifies the exponent value, which is used when encoding a size value, in case the exponent is omitted in the encoding.

The explicit exponent allowed parameter specifies whether or not an exponent part is allowed in the coded representation of a coordinate or a size value.

For example, if the magnitude code = + 11 and the granularity code = -16 the coordinates in the range from
- 1111111111.1111111111111111 (binary) to
+ 1111111111.1111111111111111 (binary) can be encoded, with each encoded coordinate being a multiple of 2**-16 (binary 0.0000000000000001), the BGU.

Related primitives :

SET DOMAIN RING

Discussion :

The magnitude code must be greater than the granularity code. The magnitude must be greater than 0. the default exponent must be greater than or equal to the granularity code. If either of these conditions are not met the primitive is ignored.

This primitive applies only to coordinates and size values.

6.2.6.4 SET REAL PRECISION

Level : 0a

Parameters :

Magnitude code	(I)
Granularity code	(I)
Default exponent	(I)
Explicit exponent allowed (one of : ALLOWED, FORBIDDEN)	(E)

Description :

The magnitude code parameter specifies the largest possible magnitude of positive or negative real numbers which can be encoded (the number of bits which may occur to the left of the binary radix point in the representation of a real number as a binary fraction plus one bit for the sign bit).

The granularity code parameter spcecifies the smallest nonzero real number that can be expressed with this precision. It does this by specifying the smallest exponent that is permitted in the coded representation of a real number.

The default exponent parameter specifies the exponent, which is used when encoding a real number, in case the exponent is omitted in the encoding.

The explicit exponent allowed parameter specifies wether or not an exponent part is allowed in the coded representation of a real number.

For example, if the magnitude code = +11 and the granularity code = -16, the real numbers in the range from
-1111111111.1111111111111111 (binary) to
+1111111111.1111111111111111 (binary) can be encoded with real numbers encoded being a multiple of 2**-16 (binary 0.0000000000000001).

Related primitives :

None.

Discussion :

The magnitude code must be greater than the granularity code. The magnitude code must be greater than 0. The default exponent must be greater than or equal to the granularity code. If either of these conditions are not met the primitive is ignored.

This primitive applies only to real numbers.

6.2.6.5 SET COLOUR INDEX PRECISION

Level : 0a

Parameters :

Number of bits (I)

Description :

The number of bits parameter specifies the number of bits necessary to encode the index values of a colour index list.

Related primitives :

CELL ARRAY
SET PATTERN REPRESENTATION
INQUIRE PATTERN REPRESENTATION
INQUIRE PREDEFINED PATTERN REPRESENTATION
INQUIRE PIXEL ARRAY

Discussion :

None.

7. ENCODING PRINCIPLES

The primitives defined in this standard can be used together with other data elements as described in Appendix D. This document deals with graphics primitives only.

The encoding of primitives is defined in terms of a 7-bit code. When used in an 8-bit environment, bit 8 of each octet shall be zero (except within the datatype 'string').

This clause deals with the detailed encoding principles of :

- the opcodes of the primitives;
- the operands of the primitives.

Each primitive is coded according to the following rules :

- a primitive is composed of one opcode and operands as required;
- the opcodes are encoded in column 2 or 3 of the 7-bit Code Table;
- operands are encoded in columns 4 up to 7. (However, the coded representation of a 'string' operand may include bit combinations from other columns of the Code Table - see the description of string operands in 7.2.5.)

The start of a character string is indicated by START OF STRING (SOS) represented by ESC 5/8 (in a 7-bit environment, ESC = 1/11) or 9/8 (in a 8-bit environment).

A character string is terminated by the delimiter STRING TERMINATOR (ST) represented by the escape sequence ESC 5/12 (in a 7-bit environment, ESC = 1/11) or 9/12 (in a 8-bit environment).
Only the datatypes string and record use character strings.
The SOS code of a string which defines a character substitution has to be omitted.

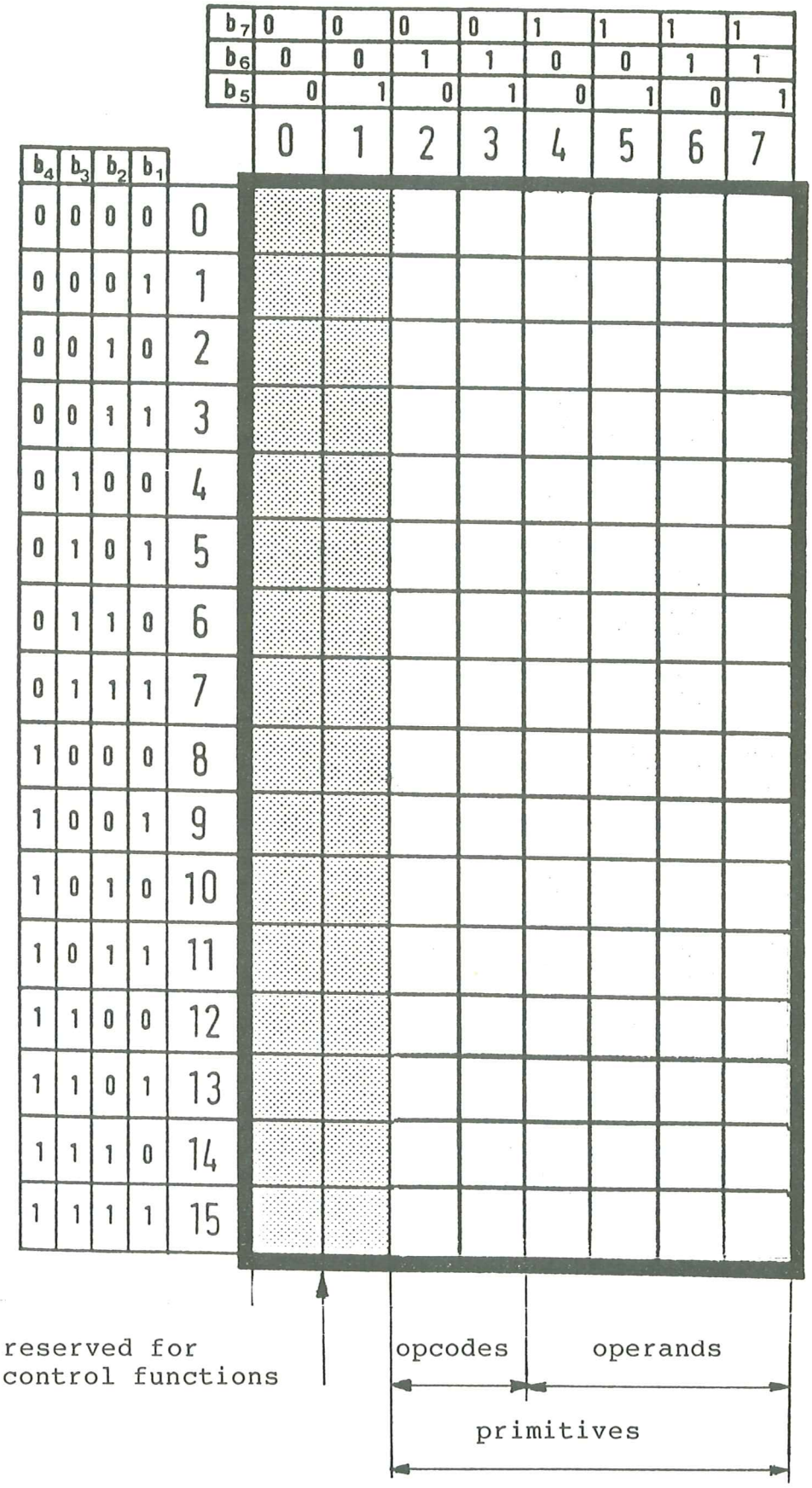


Table 5 7-bit Code Table as used for GDS coding

7.1. ENCODING PRINCIPLES OF THE OPCODE

The encoding technique used when encoding opcodes supplies :

- the basic opcode set;
- extension opcode sets.

The description of this encoding technique is therefore divided into two parts :

- description of the encoding technique of the basic opcode set;
- description of the extension mechanism.

7.1.1. ENCODING TECHNIQUE OF THE BASIC OPCODE SET

The basic opcode set consists of single-byte and double-byte opcodes. The general structure of an opcode is shown in fig. 15.

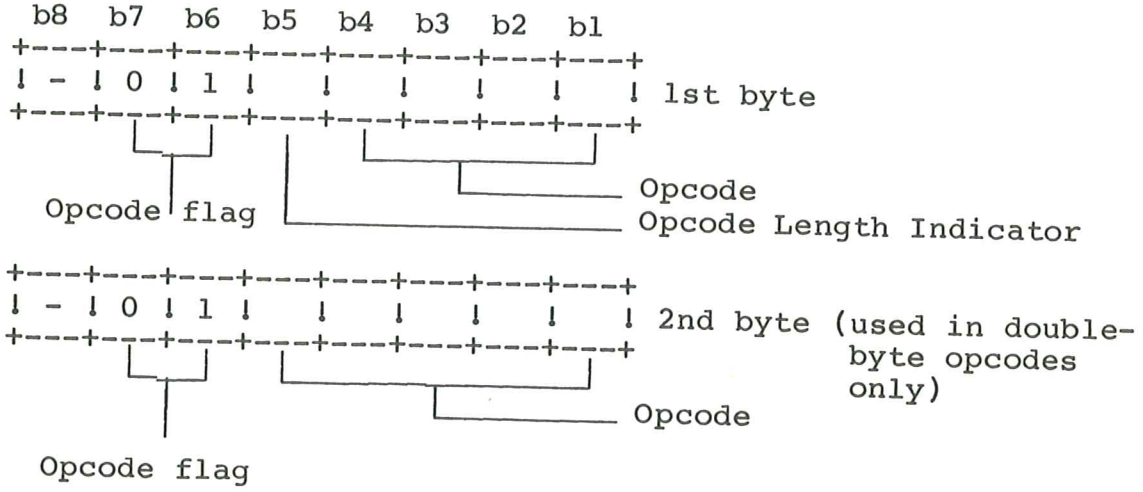


Fig. 15 : Opcode encoding structure

For single-byte opcodes the opcode length indicator, bit b5, is ZERO (opcodes of column 2).
Bits b4 to b1 are used to encode the opcode.

For double byte opcodes the opcode length indicator, bit b5, of the first byte is ONE. Bits b4 to b1 of the first byte and bits b5 to b1 of the second byte are used to encode the opcode (first byte of the opcode is from column 3, the second byte being from column 2 or 3).

The code EXTEND OPCODE SPACE (EOS, 3/15) is used in a different sense (see section 7.1.2).

The basic opcode set supplied by this encoding technique consists of 496 opcodes, being :

- 16 single-byte opcodes (from column 2);
- 15 x 32=480 double-byte opcodes (first byte from column 3 except code 3/15, second byte from column 2 or column 3).

7.1.2. EXTENSION MECHANISM

The basic opcode set can be extended with an unlimited number of extension opcode sets by means of the EXTEND OPCODE SPACE code (EOS, 3/15).

The N-th extension opcode set consists of opcodes of the basic opcode set, prefixed with n times the code EOS . The three possible formats of an opcode from the N-th extension opcode set are :

Opcode format	Extension codes	Basic opcode set codes
1	⟨EOS⟩...⟨EOS⟩ (n times)	⟨2/x⟩
2	⟨EOS⟩...⟨EOS⟩ (n times)	⟨3/y⟩ ⟨2/z⟩
3	⟨EOS⟩...⟨EOS⟩ (n times)	⟨3/y⟩ ⟨3/z⟩

EOS = 3/15
x = 0, 1,....., 15
y = 0, 1,....., 14
z = 0, 1,....., 15
n = 0, 1,.....

n = 0 determines the basic opcode set
n = 1 determines the 1st extension opcode set
n = N determines the N-th extension opcode set

The number of opcodes supplied by this encoding technique (basic opcode set plus extension opcode sets) is :

- 16 single-byte opcodes from the basic opcode set
  (opcode format 1, n = 0)

- 480 double-byte opcodes from the basic opcode set
  (opcode format 2 and 3, n = 0)
  16 double-byte opcodes from the 1st extension opcode set
  (opcode format 1, n = 1)

- 480 N-byte opcodes from extension opcode set N-2
  (opcode format 2 and 3, n = N - 2)
  16 N-byte opcodes from extension opcode set N-1
  (opcode format 1, n = N-1)

7.2. ENCODING PRINCIPLES OF THE OPERANDS

The operand part of a primitive may contain one or more operands, each operand consisting of one or more bytes. The general format of an operand byte is given in fig. 16.

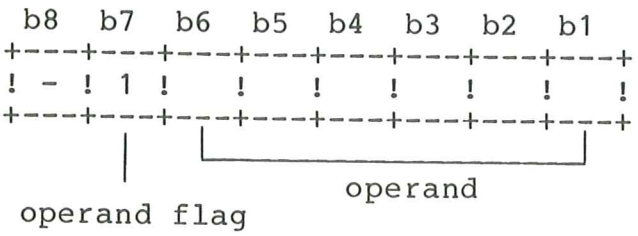


Fig. 16 : Operand encoding structure

The encoding of the operands may make use of the following DATATYPES :

a. Identifier (ID)
b. Enumerated type (E)
c. String (S)
d. Record (REC)
e. Point (P)
f. Colour index (CI)
g. Index (IX)
h. Size value (V)
i. Integer number (I)
j. Real Number (R)
k. Matrix (M)
l. Colour direct (CD)
m. Colour list (CL)

To encode the datatypes five STRUCTURES are available :

1. Basic format
2. Real format
3. Bitstream format
4. String format
5. Record format

These five structures will be explained in the following sections.

7.2.1. BASIC FORMAT

Each basic format operand is coded as a sequence of one or more bytes, which structure is shown in figure 17 :

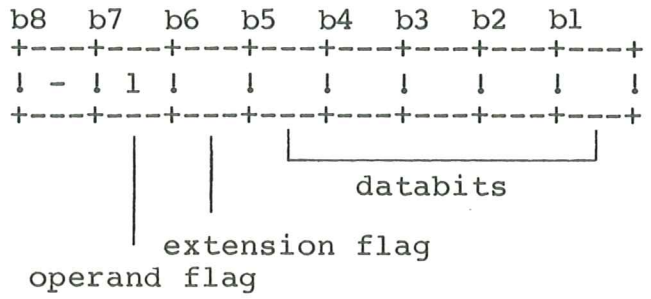


Fig. 17 : Basic format structure

Bit b6 of each byte is the extension flag.
For single-byte operands, the extension flag is ZERO. In multiple-byte operands, the extension flag is ONE in all bytes except the last byte, where it is ZERO.

The Basic format is used to encode :
- Identifiers (ID)
- Enumerated types (E)
- Colour indices (CI)
- Indices other than colour indices (IX)
- Integer numbers (I)

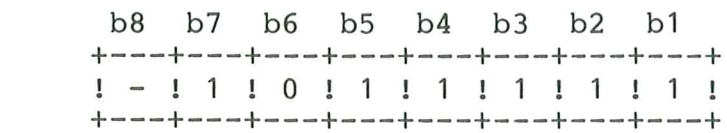
The most significant part of the operand is coded in the first byte. The least significant part of the operand is coded in the last byte.

The range of integer numbers is subdivided into a non-negative range and a negative range, using bit b5 as a sign bit.
If bit b5 is set to ZERO, the integer is non-negative; if bit b5 is set to ONE, the integer is negative.
PLUS ZERO is considered to be non-negative.
MINUS ZERO is used in incremental mode encoding (see section 7.2.2.3.3.).

In some situations the range of integers is used to indicate two categories of parameters, e.g. private and standardized use. Non-negative values indicate standardized use and negative values indicate private use.

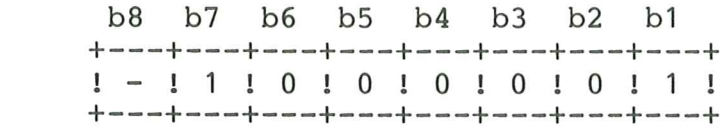
The enumerated data type is encoded in the same manner as a integer data type.
The data types Identifier, Colour index and Index are encoded in the Basic format without a sign bit.

In the next figures some examples are given.



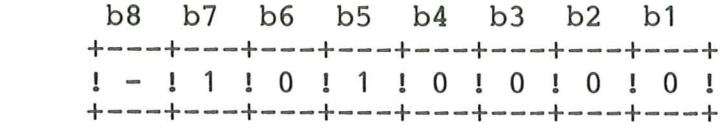
Parameter : segment name
Datatype : identifier (ID)
Operand value : 31

Fig. 18 : Identifier encoding



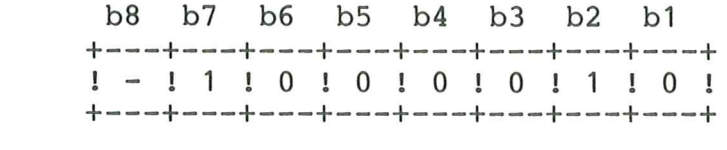
Parameter : polyline colour ASF
Datatype : enumerated type (E)
Operand value : BUNDLED

Fig. 19 : Enumerated type encoding



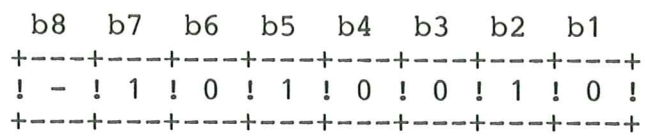
Parameter : polyline colour index
Datatype : colour index (CI)
Operand value : 16

Fig. 20 : Colour index encoding



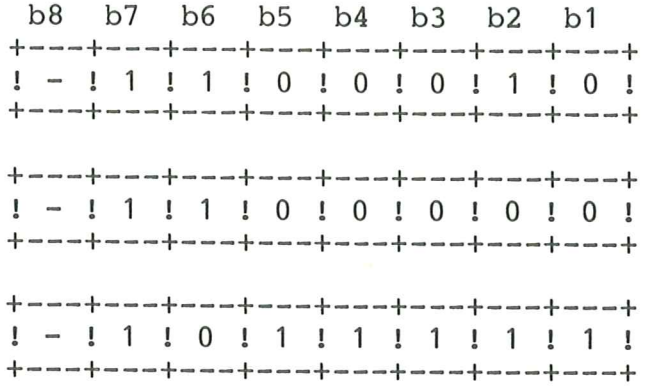
Parameter : line type
Datatype : integer (I)
Operand value : +2 (standardized line type 2)

Fig. 21 : Integer encoding for
standardized use



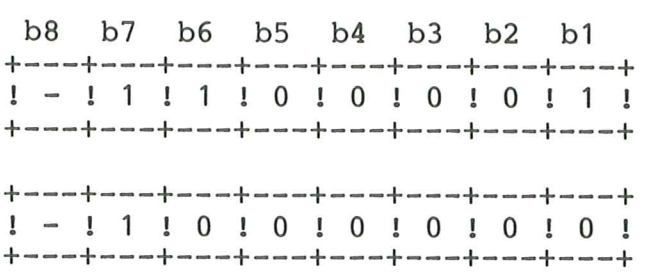
Parameter : line type
Datatype : integer (I)
Operand value : -2 (private line type 2)

Fig. 22 : Integer encoding for private use



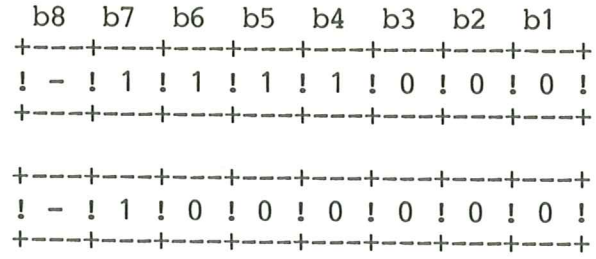
Parameter : segment name
Datatype : identifier (ID)
Operand value : 2079

Fig. 23 : Identifier encoding (more than one byte)



Parameter : bundle index
Datatype : index (IX)
Operand value: 32

Fig. 24 : Index encoding (more than one byte)



Parameter : marker type
Datatype : integer (I)
Operand value : -256 (private marker type 256)

Fig. 25 : Integer encoding for private use (more than one byte)

7.2.2. REAL FORMAT

Each Real format consists of a mantissa part and an optional exponent part both coded in Basic format. The exponent is the power of two by which the mantissa is to be multiplied. The exponent may be implicitly defined as a default exponent, which is then omitted in the Real format the or exponent may be coded explicitly as the second part of the Real format.

<real format> = <mantissa part> [<exponent part>]

Whether or not the exponent part is explicitly coded as a part of the Real format depends on :

- the current "explicit exponent allowed" value, which can be set by the "explicit exponent allowed" parameter of the SET COORDINATE PRECISION and SET REAL PRECISION primitives;

- the value of the "exponent follows" flag in the mantissa part of a Real format.

The Real format is used to encode :

- Real numbers (R)
- Size values (V)
- Points (P)
- Matrices (M)

7.2.2.1. MANTISSA

The mantissa is an integer which is coded in the Basic format. In the first byte of the mantissa, bit b5 is used as the sign bit : ZERO for a non-negative mantissa, ONE for a negative mantissa. The MINUS ZERO code for a mantissa is not permitted and reserved for future use.

If the current "explicit exponent allowed" value is ALLOWED, bit b4 of the first byte of a mantissa is used as the "exponent follows" bit : ONE if an explicit exponent follows the mantissa, ZERO as no exponent follows the mantissa.

If the current "explicit exponent allowed" value is FORBIDDEN, bit b4 of the first byte of a mantissa is used as databit.

Figures 26 and 27 show the general format of a mantissa. Figure 28 shows an example of a multiple-byte mantissa.

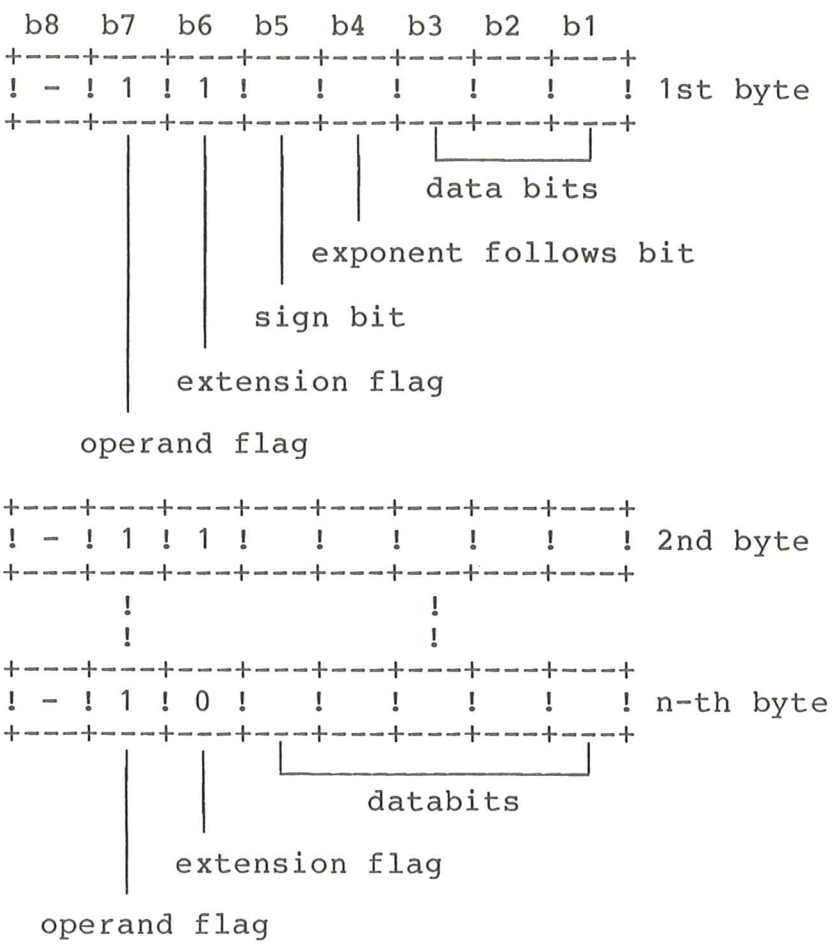


Fig. 26 Encoding of a mantissa
using the Basic format

Explicit exponent allowed = ALLOWED

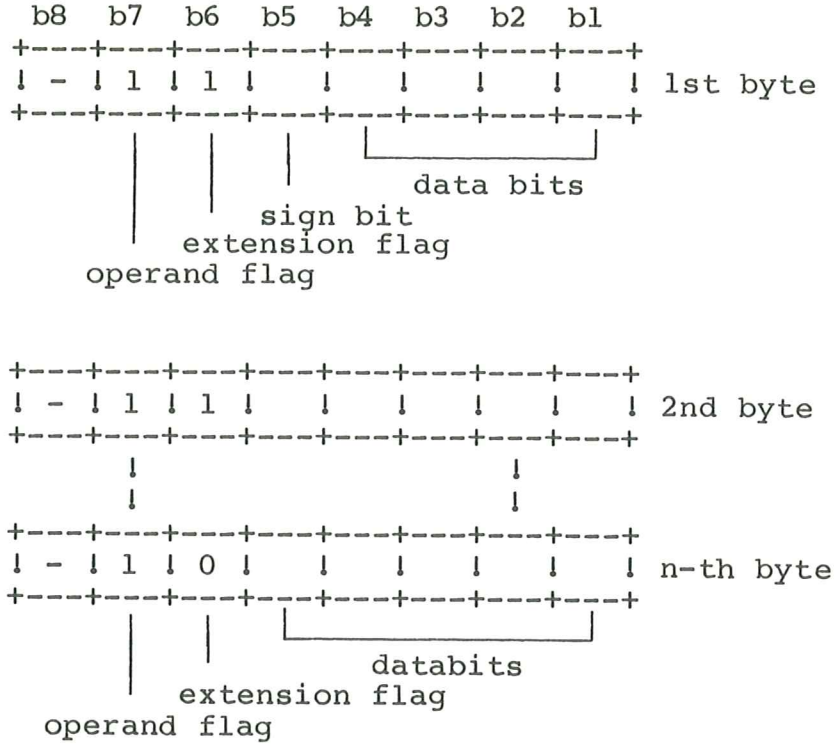


Fig. 27 : Encoding of a mantissa
using the Basic format

Explicit exponent allowed = FORBIDDEN

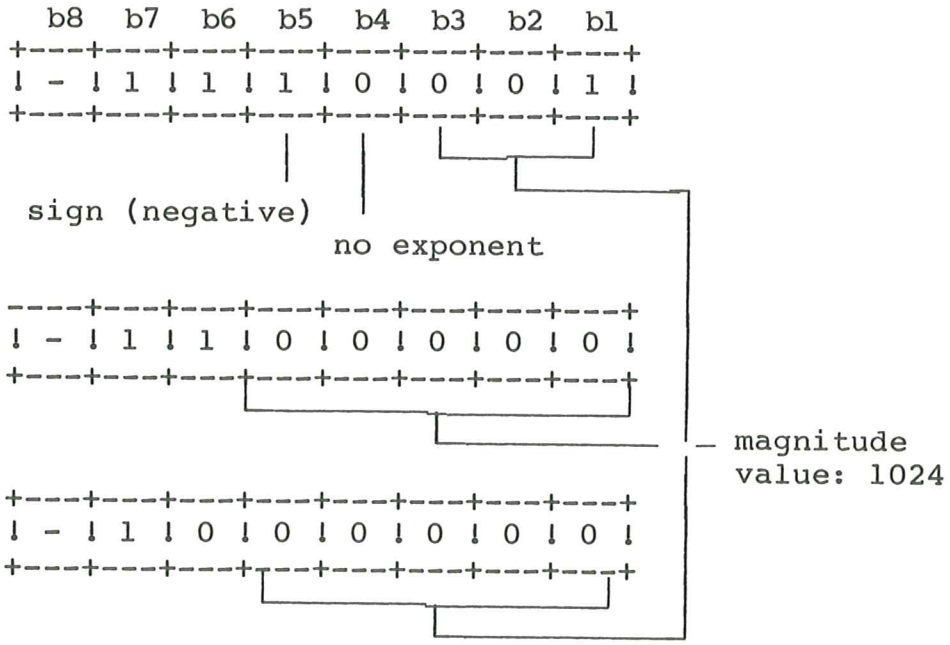


Fig. 28 : A multiple-byte mantissa

Mantissa value = -1024

7.2.2.2 EXPONENT

If the current "explicit allowed" value is ALLOWED and the "exponent follows" bit in a mantissa in ONE, then an explicit exponent follows in the mantissa.
The exponent is coded as an integer in Basic format.

If the current "explicit exponent allowed" value is FORBIDDEN or if the current "explicit exponent allowed" value is ALLOWED, together with the "exponent follows" bit in the mantissa part having the value ZERO, then the exponent is omitted from the coding and an implicit default exponent value is assumed.
The default value for an exponent depends on the data type being encoded.

If an exponent is omitted in a real number, a default value determined by SET REAL PRECISION is assumed.

If an exponent is omitted in a size value, a default value determined by SET COORDINATE PRECISION is assumed.

If an exponent is omitted in the x-component of a point, the exponent used for the x-component of the preceding point is assumed. Likewise, the exponent in a y-coordinate defaults to the value of the exponent in the preceding y-coordinate. For single points, (points not part of point lists) and for the first point of a point list, an omitted exponent assumes a default value determined by SET COORDINATE PRECISION.

If an exponent is omitted in one of the elements of a matrix, the default value, which is assumed for the exponent, depends on the matrix element which is concerned (see section 7.2.2.4.).

7.2.2.3. POINTS AND POINT LISTS

Point lists (data type nP) may be coded with either one of two coding structures, or a mixture of the two. These structures are called 'Displacement mode' and 'Incremental mode'. Individual points (data type P) and the first point of each pointlist, however, must always be coded in displacement mode.

7.2.2.3.1. DISPLACEMENT MODE

In Displacement mode, each point is coded as a sequence of two size value parameters. The first size value gives the x-component of the point's displacement from the preceding point, while the second size value specifies the y-component of that displacement.

$$\langle P \rangle = \langle V: \Delta x \rangle \langle V: \Delta y \rangle$$

For individual points (datatype P), and for the first point of each point list, the displacement values $\Delta x, \Delta y$ are displacements measured from the origin. For points after the first point list, each displacement is measured from the preceding point of the point list. (this is true regardless whether the preceding point was coded in Displacement mode or in Incremental mode).

7.2.2.3.2 INCREMENTAL MODE

The Incremental mode is defined as a so called Differential Chain Code (DCC). The data in this mode does not reflect actual coordinates, but identifies points on a Ring. A Ring is a set of points on a square which centre is the previously identified point. The first centre point is encoded in Displacement mode.
A Ring is characterized by its Radius (R in Basic Grid Units), its Angular resolution (by a factor p) and its Direction (D). The maximum number of points on a Ring is 8R. The actual number of points on a Ring with a given Angular resolution factor p follows from:

$$N = \frac{8R}{2^p} \quad \text{with } p = 0,1,2,3$$

N must be even. If N is odd, the encoded operand (the point list) must be discarded. If N is even for the first part of the operand and N is odd for the remaining part, the remaining part (with N being odd) is discarded.

The points on the Ring are numbered, starting at the Direction point, from 0 to M-1 for the upper part of the Ring and from -1 to -M for the lower part of the Ring, with M=N/2.

Figure 29 shows a Ring with Radius R=3 and Angular resolution factor p=0 respectively 1.

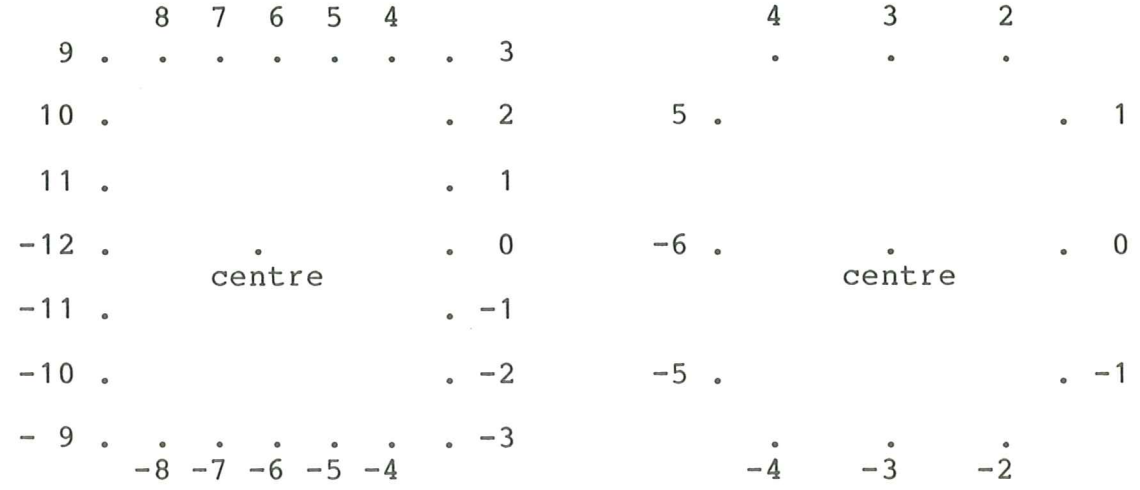


Fig. 29

Some example Rings with point numbering

The Direction of a Ring is identified by the position of the point with number ZERO. The initial position of this point is on the positive x-axis, while the Cartesian axes are drawn through the centre point of the Ring. The Direction of the Rings following the initial one is dependent on the direction of the increments. This Direction is determined in the following way:

> If P1 is the previous centre point and the current centre point is P2 (P2 is a point on the Ring with the centre in P1). The position of the point with number ZERO on the Ring, with P2 as centre point, is opposite to point P1, this is the Direction of the Ring. So the Direction of the Ring is dependent of the writing direction as indicated by the last increment. The position of the increment on the new Ring (centre P2) is described as the difference between the position of point P2 on the previous Ring and the position of the new point P3 on the current Ring.

In the DCC only the differences between points on the consecutive Rings are coded. Or to state it in another way the Direction of the Ring is dependent on the direction of the line to be displayed. As shown in figure 30, the position of point P3 is defined by the difference: P3 - P2 = -1. P3 and P2 being point numbers on the two Rings, numbered as given in figure 29. The Direction (position of the point with number ZERO) is identified by D.

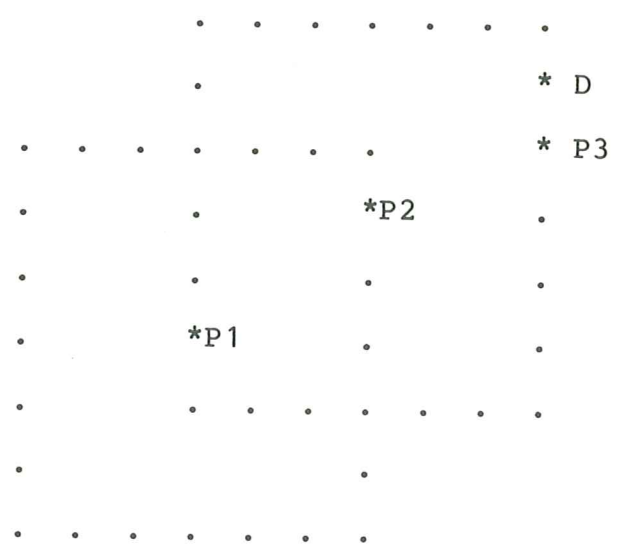


Fig. 30 : Change of direction with R=3

The basic Radius of the Ring, as used in the Incremental mode, is dependent on the granularity code as set with the SET COORDINATE PRECISION-primitive. The basic Radius follows from :

$$\text{basic Radius} = 2^{(-8 - \text{granularity code})}$$

The basic Radius is expressed in Basic Grid Units (BGUs).
The basic Radius must be equal to or greater than ONE. If the basic Radius is less than ONE, the value ONE is assumed for the basic Radius.

The basic Radius as derived from the granularity code may be changed to any value with the SET DOMAIN RING-primitive. With this primitive one can also change the Angular resolution factor p. The default value for p=0 and p can only be 0, 1, 2 or 3.

The encoding used in Incremental mode makes use of the DCC property by using variable length code-words (Huffman Code). The encoding also allows changing of the Radius and the Angular resolution factor. The Radius can have a value of R, 2R, 4R or 8R, where R is the defined Radius. The Angular resolution factor p can be 0, 1, 2, or 3.

The Huffman Code table used in the Incremental mode is a fixed length table. To allow the encoding of more points on a Ring two Escape codes are defined. With these Escape codes the points outside the Huffman Code table can be addressed. The end of the incremental mode data is indicated by an End of Block value in the Huffman Code table.

The fixed Huffman Code is given in table 6.

Length	Code-word	Point number
2	00	0
2	10	1
2	01	-1
4	1100	2
4	1101	-2
6	111000	3
6	111001	-3
6	111010	4
6	111011	-4
8	11110000	5
8	11110001	-5
8	11110010	6
8	11110011	-6
8	11110100	7
8	11110101	-7
8	11110110	8
8	11110111	-8
10	1111100000	9
10	1111100001	-9
10	1111100010	10
10	1111100011	-10
10	1111100100	11
10	1111100101	-11
10	1111100110	12
10	1111100111	-12
10	1111101000	13
10	1111101001	-13
10	1111101010	14
10	1111101011	-14
10	1111101100	15
10	1111101101	-15
10	1111101110	16
10	1111101111	-16
10	1111110000	17
10	1111110001	-17
10	1111110010	18
10	1111110011	-18
10	1111110100	19
10	1111110101	-19
10	1111110110	C1
10	1111110111	-20
10	1111111000	C2
10	1111111001	C3
10	1111111010	C4
10	1111111011	C5
10	1111111100	C6
10	1111111101	IM-ESC 1
10	1111111110	IM-ESC 2
10	1111111111	End of block

Table 6
Huffman Code table
for Incremental mode

The ⟨End of Block⟩ code from the Huffman Code table identifies the end of the Incremental mode data. Remaining bits in the last Incremental mode data byte have no meaning; they will be ignored.

The Incremental Mode escape codes ⟨IM-ESC 1⟩ and ⟨IM-ESC 2⟩ are used to extend the addressable number of points, e.g. points outside the range -20 to 19. The code ⟨IM-ESC 1⟩ adds +20 or -20 the following code depending on the sign of that following point. The code ⟨IM-ESC 2⟩ adds +40 or -40 to the following code, depending on the sign. The escape codes can follow each other in any desired order. The following examples demonstrate some possible combinations, [n] is a point number.

⟨IM-ESC 1⟩ [1] = point number 21
⟨IM-ESC 1⟩ [-1] = point number -21
⟨IM-ESC 2⟩ [14] = point number 54
⟨IM-ESC 2⟩ [-12] = point number -52
⟨IM-ESC 1⟩⟨IM-ESC 2⟩ [6] = point number (20+40+6) = 66
⟨IM-ESC 2⟩⟨IM-ESC 1⟩ [-18] = point number (-40-20-18) = -78

The codes C1 up to C6 are used to change the parameters that define the Ring to be used. The values of R are taken from the range R_0, $2R_0$, $4R_0$ and $8R_0$, where R_0 is the value of the Ring Radius before entering the Incremental Mode. The values of p are taken from the range 0, 1, 2 and 3. The function of these codes is as follows:

a. C1
Change the Ring parameters, R and p, to the next higher value e.g. if the Radius is R_0, the next higher is $2R_0$, if p=0 the next higher value is 1. R cannot become greater than $8R_0$ and p cannot become greater than 3. For example if the current Ring Radius is $8R_0$ and the current p=3, the code ⟨C1⟩ has no effect.

b. C2
Change the Ring parameters, R and p, to the next lower value. The effect of the code ⟨C2⟩ is the inverse of code (C1). R cannot become smaller than R and p cannot become smaller than 0. For example if the current Radius is R and the current p=0, the code ⟨C2⟩ has no effect.

c. C3
Change the Ring Radius R to the next higher value. The code ⟨C3⟩ has no effect if the current Radius = $8R_0$.

d. C4
Change the Angular resolution factor p to the next higher value. The code ⟨C4⟩ has no effect if the current p=3.

e. C5
Change the Ring Radius R to the next lower value. The code ⟨C5⟩ has no effect if the current Radius = R_0.

f. C6
Change the Angular resolution factor p to the next lower value. The code ⟨C6⟩ has no effect if the current p=0.

In addition, those codes (C1 to C6) set the position of the point with number ZERO on the positive x-axis, while the Cartesian axes are drawn through the center point of the Ring.

7.2.2.3.3. INCREMENTAL MODE ENCODING

The structure of the Incremental mode format is given in fig. 31.

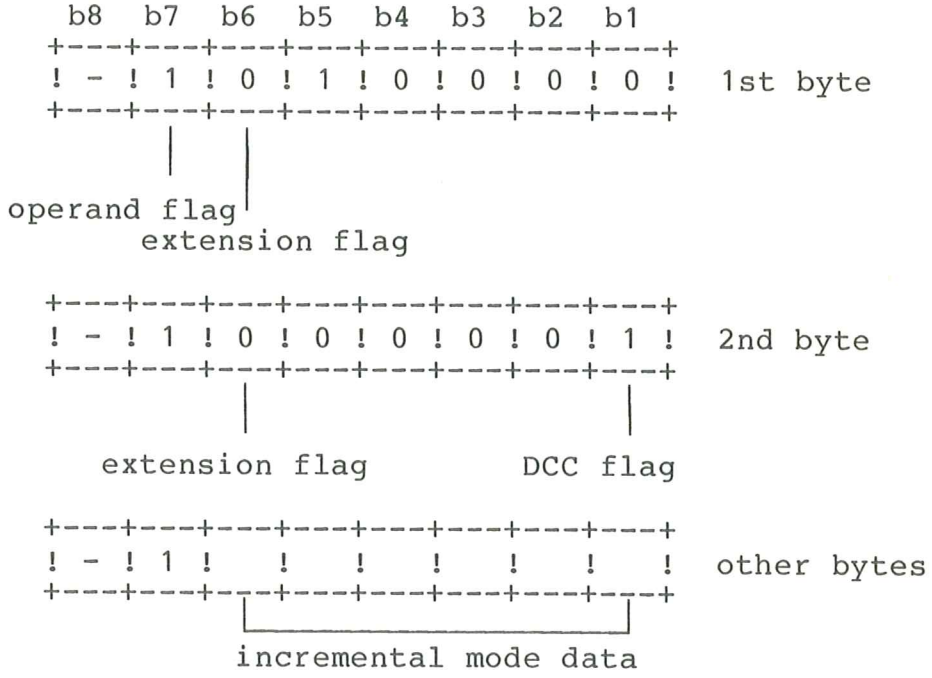


Fig. 31 : Incremental mode structure

The first byte is coded according to the Basic format structure indicating an integer value MINUS ZERO.The second byte is structured according to the Basic format structure. Bit b1 is now set to ONE to identify the use of Direct Chain Coding (DCC).
The bits b5 to b2 are reserved for future use and now set to ZERO.

The Incremental mode uses variable length code-words. This implies that the code-words do not fit in the Incremental mode data bits (bit b6 to bit b1 of the other bytes). The code-words are packed in consecutive bits of the Incremental mode bytes, starting from high numbered bits to lower numbered bits. If the code-word does not fit in one byte, the most significant part is packed in the first byte, the remaining part is packed in the second byte and so on.

The end of Incremental mode data is identified by the End of Block code. Remaining bits in the last Incremental mode data byte have no meaning, they will be ignored.

7.2.2.4. MATRICES

A matrix is a structured datatype consisting of six matrix elements. The six matrix elements are ordered in three columns, each column containing two rows. The matrix elements are numbered Mxy, x indicating the row (1..2) and y indicating the column (1..3). Element M23 denotes the element in the third column of row 2.

The elements of column 3 (M13 and M23) are in NDC coordinates, the other elements are real numbers.

The elements of column 3 (M13 and M23) are encoded in the same way as single points, using the Real format-displacement mode. The displacement values are measured from the origin.

The other elements are encoded as real numbers, using the Real format.

A matrix is encoded column by column. This means that M11 is the first element to be encoded. The next element to be encoded is M21, then M12 etc.

7.2.3. BITSTREAM FORMAT

Each bitstream format operand is encoded as a sequence of one or more bytes, which structure is shown in fig. 32.

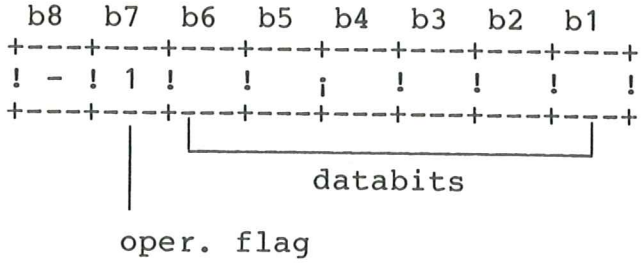


Fig. 32 : Bitstream format structure

The Bitstream format is used to encode :

- Incremental mode coordinates (see section 7.2.2.3.3.)
- Colour direct (see section 7.2.3.1.)
- Colour index lists (see section 7.2.4.1.1.)

Bitstream data are packed in consecutive databits starting from high numbered bits to lower numbered bits of the first byte for the most significant part of the bitstream data.

The end of a Bitstream format operand can not be derived from the Bitstream format itself (the format is not self-delimiting).

- Encoding Incremental mode coordinates, the end of the data (which identifies the end of the Bitstream format operand) is identified by the <End of Block> code.

- Encoding Colour index lists, the number of bits needed to encode the Colour index list (which identifies the end of the Bitstream format operand) is set by the SET COLOUR INDEX PRECISION primitive.

- Encoding Colour direct data, the number of bits needed to encode this data (which identifies the end of the Bitstream format operand) is set by the SET COLOUR HEADER primitive.

7.2.3.1. COLOUR DIRECT ENCODING

In encoding colour data which must be loaded into a colour table (Colour direct) the following applies : The bitstream format is composed of a number of bytes as derived from the unit resolution parameter of the SET COLOUR HEADER-primitive.

Each byte contains two bits for each component (R, G, B) (see fig. 33).
The first byte contains the most significant bits. In the last byte bit, b3, b2 and b1 may be left unused.

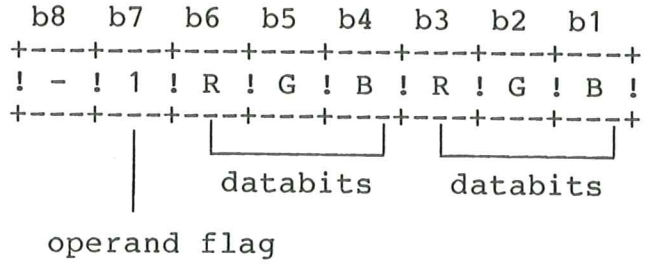


Fig. 33 : Colour direct encoding

Fig 34 gives an example of the colour loading with a unit resolution of 4.

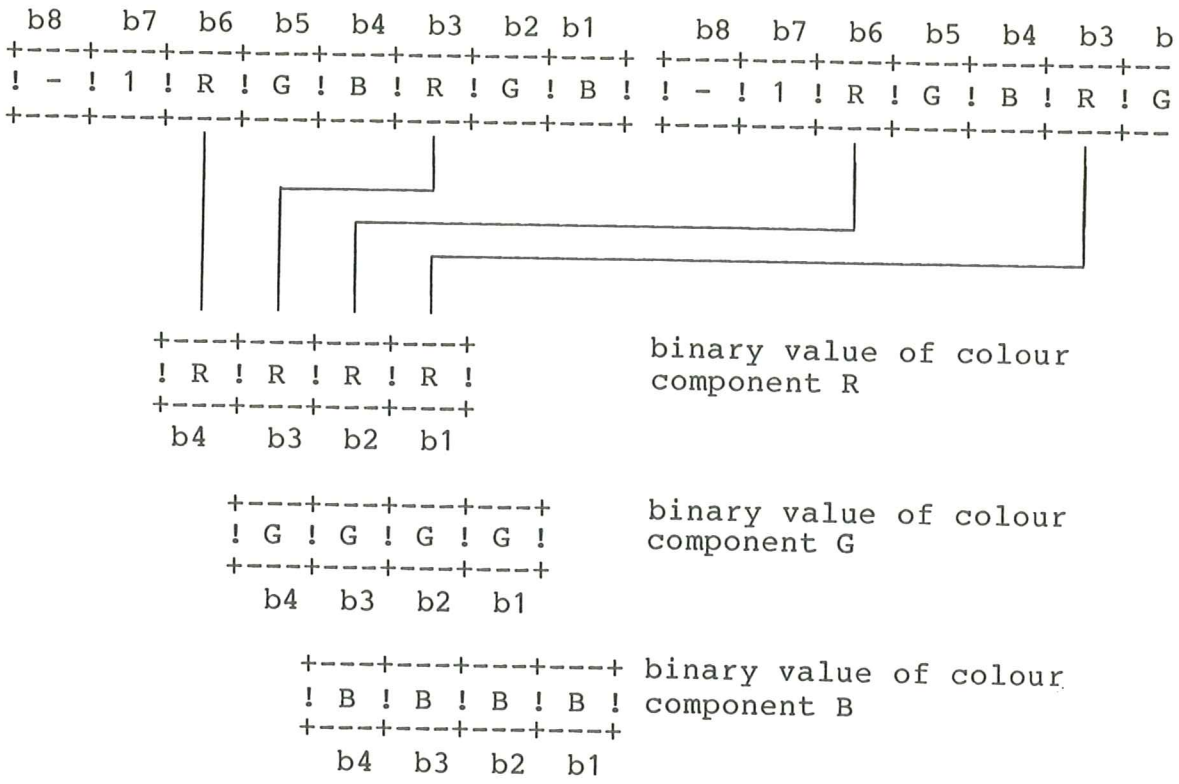
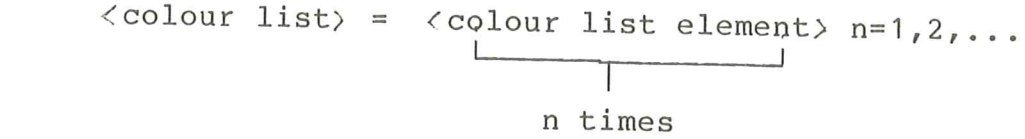


Fig. 34 : Colour loading with a unit resolution of 4

7.2.4. COLOUR LISTS

A colour list is a structured datatype consisting of a sequence of colour list elements, each element specifying a colour value. The general format of a colour list is :



The colours can be specified :

- indexed, with colour list elements representing colour indices pointing into a colour table. The colour list is called a colour index list;

- direct, with colour list elements representing the RGB components of the colour data. The colour list is called a colour direct list.

There are two ways to encode a colour list. If many adjacent colour list elements have the same colour value, runlength encoding is efficient. However, for short runs of only one or two colour list elements with the same colour value, it is more efficient to encode the colour list elements individually.

When using runlength encoding, for each run the colour of the colour list elements is encoded (run colour), followed by a count telling how many colour list elements are in the run.

7.2.4.1. COLOUR LIST ENCODING

The method, used in encoding a colour list is specified in the first byte of a colour list :

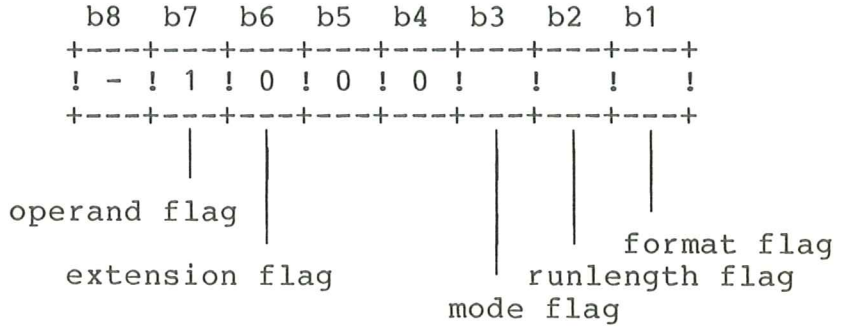


Fig. 35 : First byte of Colour list encoding

The mode flag, b3, specifies whether the colour list is encoded as a colour index list (mode flag is ZERO) or as a colour direct list (mode flag is ONE).

The runlength flag, bit b2, specifies whether runlength encoding is used (runlength flag is ONE) or not (runlength flag is ZERO).

The format flag, bit b1, specifies whether the Basic format (format flag is ZERO) or the Bitstream format (format flag is ONE) is used to encode the colour list.

Bit b5 and b4 are reserved for future use and now set to ZERO.

In the next sections the different methods of encoding colour lists are described in more detail.

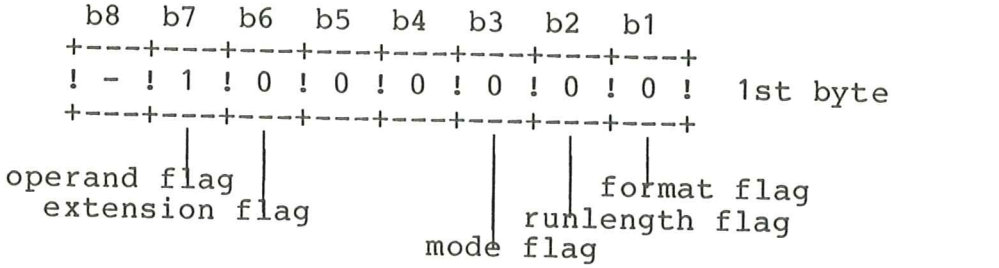
7.2.4.1.1. INDIVIDUAL ENCODING OF COLOUR INDEX LISTS

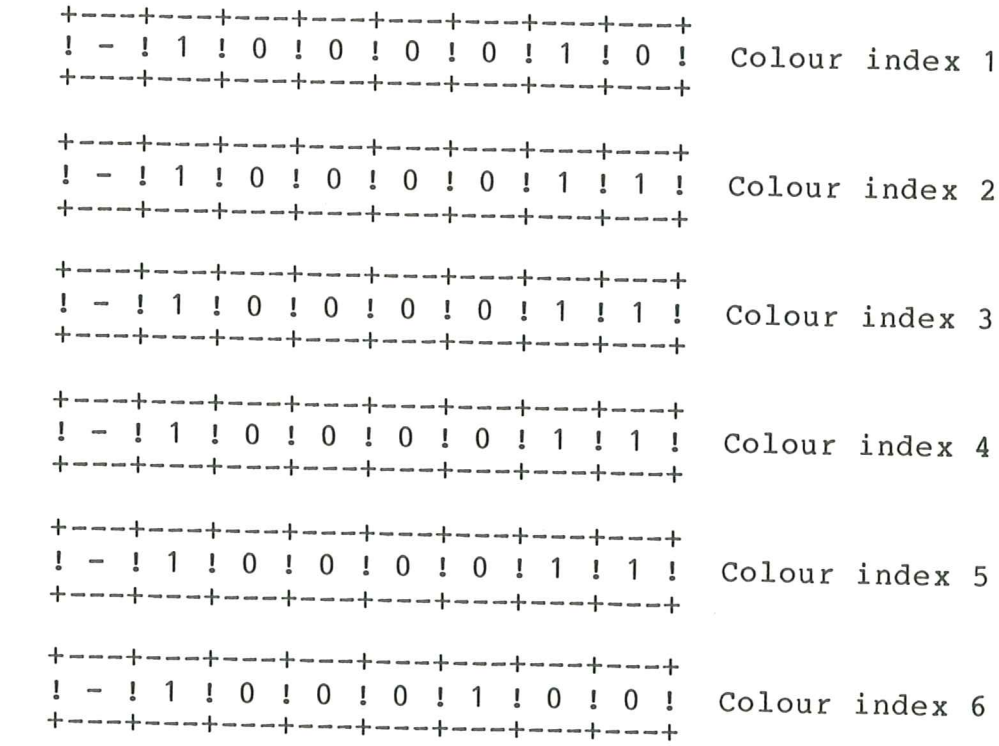
This encoding method specifies a colour list by individually encoding all colour list elements as colour indices.

The colour indices can be encoded using either the Basic format or the Bitstream format.

If Basic format encoding is specified, bit b3 of the first byte set to ZERO, the indices of the Colour index list are encoded using the Basic format described in section 7.2.1 for each separate index.

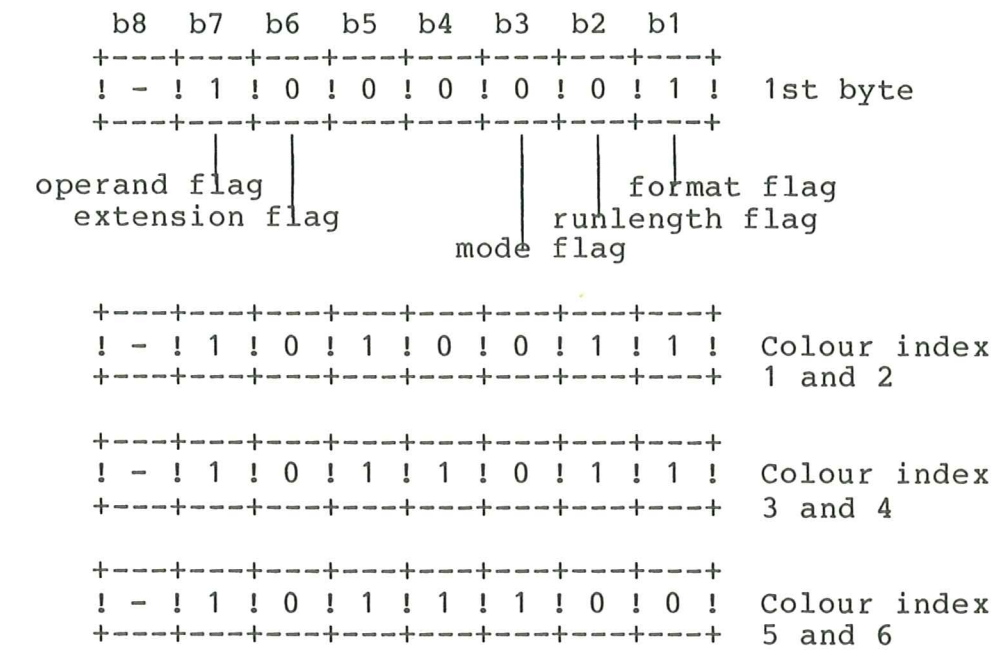
If Bitstream format encoding is specified, bit b3 of the first byte set to ONE, the indices of the Colour index list are encoded using the Bitstream format described in section 7.2.3. The length (in bits) of each index is identified by the SET COLOUR INDEX PRECISION primitive. These indices are packed in consecutive bits starting from bit b6 of the first Bitstream format byte. If an index does not fit in one byte, the most significant part is packed in the first byte, the remaining part is packed in the second byte and so on. Some examples are given in figures 36 and 37.





Parameter : Pattern cell colour index list
Datatype : Colour index list
Operand value : 2, 3, 3, 3, 3, 4,

Fig. 36 : Colour index list individually encoded using the Basic format.



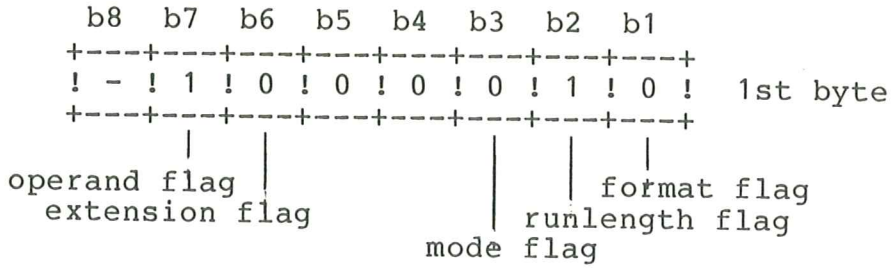
Parameter : Pattern cell colour index list
Datatype : Colour index list
Operand value : 2, 3, 3, 3, 3, 4,

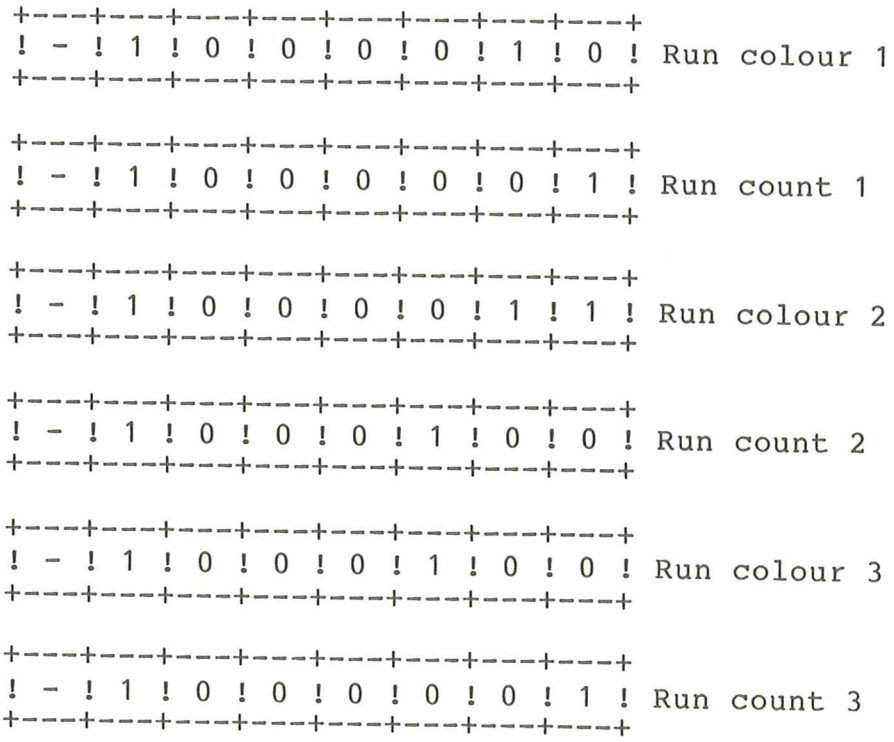
Fig. 37 : Colour index list individually encoded using the Bitstream format.

7.2.4.1.2. RUNLENGTH ENCODING OF COLOUR INDEX LISTS

When using runlength encoding of a colour index list both run colour and the run count are encoded using the Basic format (respectively encoded as a Colour index datatype and as an Integer datatype), so the format flag, bit b1 of the first byte, is always set to ZERO.

An example is given in figure 38.





Parameter : Pattern cell colour index list
Datatype : Colour index list
Operand value : 2, 3, 3, 3, 3, 4,

Fig. 38 : Colour index list encoded using runlength encoding.

7.2.4.1.3. ENCODING OF COLOUR DIRECT LISTS

In this standard only colour index lists are defined, so the mode flag, bit b3 of the first byte, is always set to ZERO. The encoding of colour direct lists is reserved for future use.

7.2.5. STRING FORMAT

The structure of a String format is given in figures 39 and 40.

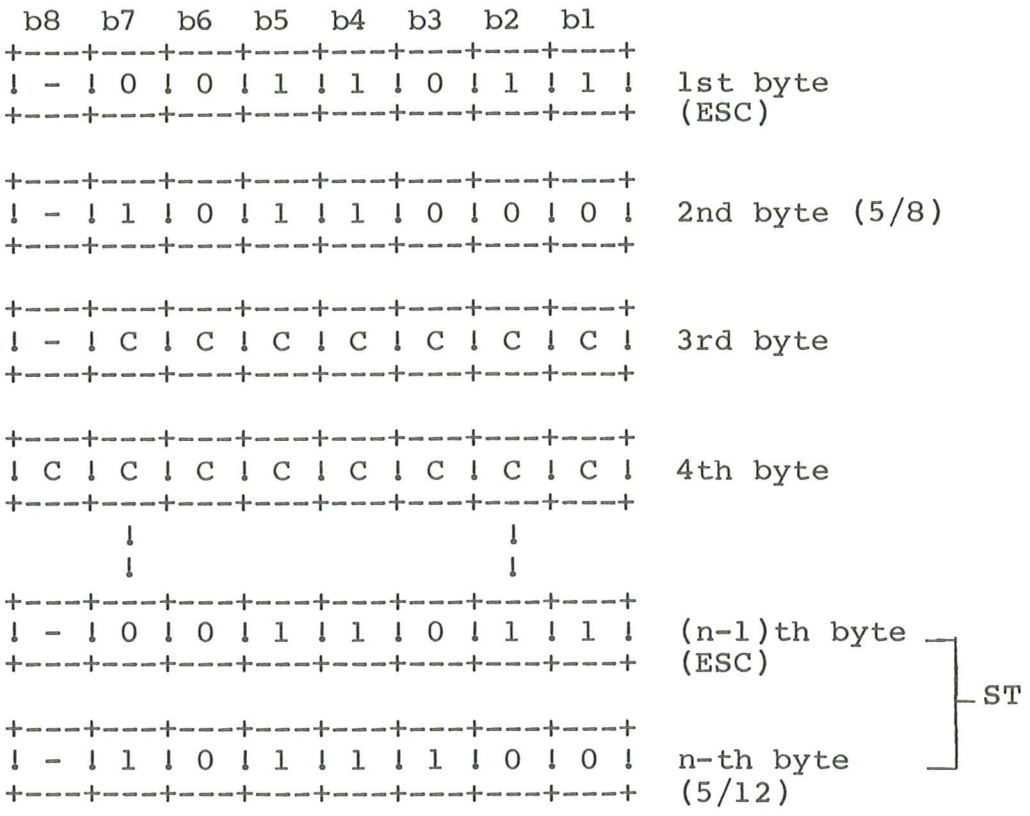


Fig. 39 : String format structure in a 7-bit environment.
Databits are marked C.

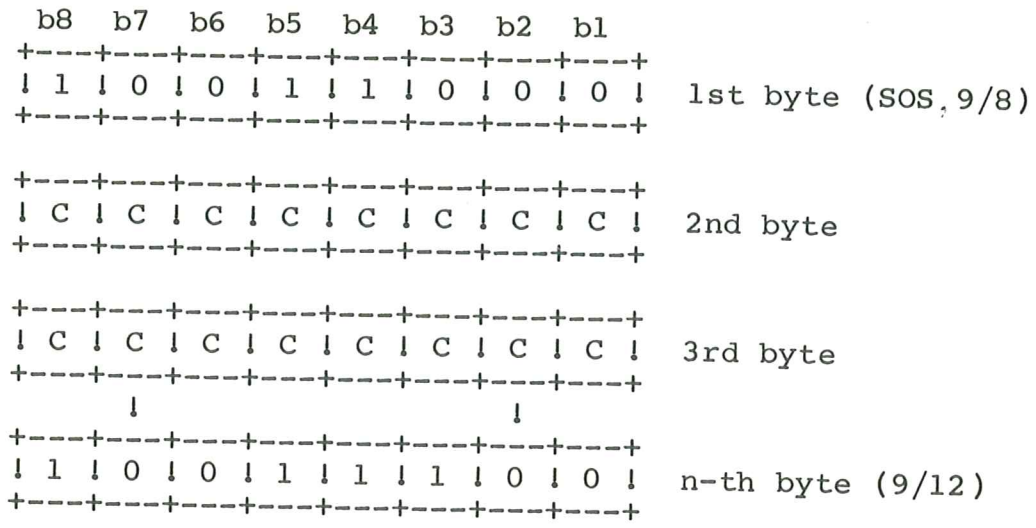


Fig. 40
String format structure in a 8-bit environment.
Databits are marked C.

The number of bytes needed to encode a string operand equals to the number of characters of the string plus four (for SOS and ST, coded 1/11, 5/8 and 1/11, 5/12) in a 7-bit environment and equals to the number of characters of the string plus two (for SOS and ST, coded 9/8, and 9/12) in a 8-bit environment.

The encoding of a string is the only exception of the general coding rules indicated in section 7.2.

7.2.6. RECORD FORMAT

The structure of a Record format is implementation dependent; its format must consist of one or more already defined formats (see sections 7.2.1. to 7.2.5.).

8. Encoding of the Primitives

The coding of the primitives is described in the following section. Section 8.1 gives the code of each primitive; section 8.2 defines the encoding of each primitive indicating the order of the parameters along with their specific data type.

8.1 PRIMITIVE Encoding

The following table gives the code and subcode for each primitive. In order to be able to determine the data flow direction of each primitive a specific opcode is assigned to each of these primitives.

Primitives	Opcode Byte 1	Opcode Byte 2
WORKSTATION MANAGEMENT		
OPEN WORKSTATION	3/2	3/0
CLOSE WORKSTATION	3/2	3/1
ACTIVATE WORKSTATION	3/2	3/2
DEACTIVATE WORKSTATION	3/2	3/3
CLEAR WORKSTATION	3/2	2/0
SET DEFAULTS	3/2	2/5
GDS ESCAPE 1	3/2	2/11
GDS ESCAPE 2 (REQUEST)	3/2	2/12
GDS ESCAPE 2 (RESPONSE)	3/2	3/12
MESSAGE	3/2	2/13
INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (REQUEST)	3/2	2/7
INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (RESPONSE)	3/2	3/7
OUTPUT DRAWING		
POLYLINE	2/0	-
POLYMARKER	2/2	-
FILL AREA	2/1	-
TEXT	2/4	-
CELL ARRAY	2/3	-
GENERALIZED DRAWING PRIMITIVE	2/5	-

Primitives	Opcode Byte 1	Opcode Byte 2
OUTPUT ATTRIBUTES		
SET POLYLINE REPRESENTATION	3/1	3/11
SET POLYLINE INDEX	3/1	2/3
SET LINE TYPE	3/1	2/2
SET LINE WIDTH SCALE FACTOR	3/1	2/1
SET POLYLINE COLOUR INDEX	3/1	2/0
SET POLYMARKER REPRESENTATION	3/1	3/13
SET POLYMARKER INDEX	3/1	2/14
SET MARKER TYPE	3/1	2/12
SET MARKER SIZE SCALE FACTOR	3/1	2/13
SET POLYMARKER COLOUR INDEX	3/1	2/11
SET FILL AREA REPRESENTATION	3/1	3/12
SET FILL AREA INDEX	3/1	2/10
SET FILL AREA INTERIOR STYLE	3/1	2/5
SET FILL AREA COLOUR INDEX	3/1	2/4
SET FILL AREA STYLE INDEX	3/1	2/6
SET PATTERN REPRESENTATION	3/1	3/15
SET PATTERN REFERENCE POINT	3/1	2/9
SET PATTERN VECTORS	3/1	2/8
SET TEXT REPRESENTATION	3/1	3/14
SET TEXT INDEX	3/1	3/6
SET TEXT FONT AND PRECISION	3/1	3/4
SET CHARACTER EXPANSION FACTOR	3/1	3/0
SET CHARACTER SPACING	3/1	3/3
SET TEXT COLOUR INDEX	3/1	2/15
SET TEXT PATH	3/1	3/2
SET CHARACTER VECTORS	3/1	3/1
SET TEXT ALIGNMENT	3/1	3/5
SET COLOUR REPRESENTATION	3/2	3/8
SET ASPECT SOURCE FLAGS	3/1	2/7
SET PICK IDENTIFIER	3/3	2/13
TRANSFORMATION		
SET WORKSTATION WINDOW	3/2	2/1
SET WORKSTATION VIEWPORT	3/2	2/2
CLIPPING		
SET CLIPPING RECTANGLE	3/2	2/3

Primitives	Opcode Byte 1	Opcode Byte 2
CONTROL		
UPDATE WORKSTATION	3/2	3/4
SET DEFERRAL STATE	3/2	3/5
EMERGENCY CLOSE	3/2	2/6
ERROR DETECTED (RESPONSE)	3/2	3/6
SEGMENT		
CREATE SEGMENT	3/3	2/0
CLOSE SEGMENT	3/3	2/1
RENAME SEGMENT	3/3	2/2
DELETE SEGMENT FROM WORKSTATION	3/3	2/3
DELETE SEGMENT	3/3	2/8
REDRAW ALL SEGMENTS ON WORKSTATION	3/3	2/9
SET HIGHLIGHTING	3/3	2/6
SET VISIBILITY	3/3	2/7
SET SEGMENT TRANSFORMATION	3/3	2/5
SET SEGMENT PRIORITY	3/3	2/12
SET DETECTABILITY	3/3	2/14
ASSOCIATE SEGMENT WITH WORKSTATION	3/3	2/10
COPY SEGMENT TO WORKSTATION	3/3	2/11
INSERT SEGMENT	3/3	2/4
INPUT		
INITIALIZE LOGICAL INPUT DEVICE	3/4	2/0
SET LOGICAL INPUT DEVICE MODE	3/4	2/1
REQUEST LOGICAL INPUT DEVICE (REQUEST)	3/4	2/2
REQUEST LOGICAL INPUT DEVICE (RESPONSE)	3/4	3/2
SAMPLE LOGICAL INPUT DEVICE (REQUEST)	3/4	2/3
SAMPLE LOGICAL INPUT DEVICE (RESPONSE)	3/4	3/3
AWAIT EVENT (REQUEST)	3/4	2/5
AWAIT EVENT (RESPONSE)	3/4	3/5
GET LOGICAL INPUT DEVICE (REQUEST)	3/4	2/4
GET LOGICAL INPUT DEVICE (RESPONSE)	3/4	3/4
FLUSH DEVICE EVENTS	3/4	2/6

Primitives	Opcode Byte 1	Opcode Byte 2
INQUIRIES OF WORKSTATION STATE LIST		
INQUIRE WORKSTATION STATE (REQUEST)	3/5	2/0
INQUIRE WORKSTATION STATE (RESPONSE)	3/5	3/0
INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (REQUEST)	3/5	2/1
INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (RESPONSE)	3/5	3/1
INQUIRE POLYLINE INDICES (REQUEST)	3/5	2/2
INQUIRE POLYLINE INDICES (RESPONSE)	3/5	3/2
INQUIRE POLYLINE REPRESENTATION (REQUEST)	3/5	2/3
INQUIRE POLYLINE REPRESENTATION (RESPONSE)	3/5	3/3
INQUIRE POLYMARKER INDICES (REQUEST)	3/5	2/4
INQUIRE POLYMARKER INDICES (RESPONSE)	3/5	3/4
INQUIRE POLYMARKER REPRESENTATION (REQUEST)	3/5	2/5
INQUIRE POLYMARKER REPRESENTATION (RESPONSE)	3/5	3/5
INQUIRE TEXT INDICES (REQUEST)	3/5	2/6
INQUIRE TEXT INDICES (RESPONSE)	3/5	3/6
INQUIRE TEXT REPRESENTATION (REQUEST)	3/5	2/7
INQUIRE TEXT REPRESENTATION (RESPONSE)	3/5	3/7
INQUIRE TEXT EXTENT (REQUEST)	3/5	2/8
INQUIRE TEXT EXTENT (RESPONSE)	3/5	3/8
INQUIRE FILL AREA INDICES (REQUEST)	3/5	2/9
INQUIRE FILL AREA INDICES (RESPONSE)	3/5	3/9
INQUIRE FILL AREA REPRESENTATION (REQUEST)	3/5	2/10
INQUIRE FILL AREA REPRESENTATION (RESPONSE)	3/5	3/10
INQUIRE PATTERN INDICES (REQUEST)	3/5	2/11
INQUIRE PATTERN INDICES (RESPONSE)	3/5	3/11
INQUIRE PATTERN REPRESENTATION (REQUEST)	3/5	2/12
INQUIRE PATTERN REPRESENTATION (RESPONSE)	3/5	3/12
INQUIRE COLOUR INDICES (REQUEST)	3/5	2/13
INQUIRE COLOUR INDICES (RESPONSE)	3/5	3/13
INQUIRE COLOUR REPRESENTATION (REQUEST)	3/5	2/14
INQUIRE COLOUR REPRESENTATION (RESPONSE)	3/5	3/14
INQUIRE WORKSTATION TRANSFORMATION (REQUEST)	3/5	2/15
INQUIRE WORKSTATION TRANSFORMATION (RESPONSE)	3/5	3/15
INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (REQUEST)	3/7	2/11
INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (RESPONSE)	3/7	3/11
INQUIRE LOGICAL INPUT DEVICE STATE (REQUEST)	3/7	2/12
INQUIRE LOGICAL INPUT DEVICE STATE (RESPONSE)	3/7	3/12

Primitives	Opcode Byte 1	Opcode Byte 2
WORKSTATION DESCRIPTION TABLE INQUIRIES		
INQUIRE WORKSTATION CATEGORY (REQUEST)	3/6	2/0
INQUIRE WORKSTATION CATEGORY (RESPONSE)	3/6	3/0
INQUIRE WORKSTATION CLASSIFICATION (REQUEST)	3/6	2/1
INQUIRE WORKSTATION CLASSIFICATION (RESPONSE)	3/6	3/1
INQUIRE MAXIMUM DISPLAY SURFACE SIZE (REQUEST)	3/6	2/2
INQUIRE MAXIMUM DISPLAY SURFACE SIZE (RESPONSE)	3/6	3/2
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (REQUEST)	3/6	2/3
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (RESPONSE)	3/6	3/3
INQUIRE DEFAULT DEFERRAL STATE VALUES (REQUEST)	3/6	2/4
INQUIRE DEFAULT DEFERRAL STATE VALUES (RESPONSE)	3/6	3/4
INQUIRE POLYLINE FACILITIES (REQUEST)	3/6	2/5
INQUIRE POLYLINE FACILITIES (RESPONSE)	3/6	3/5
INQUIRE PREDEFINED POLYLINE REPRESENTATION (REQUEST)	3/6	2/6
INQUIRE PREDEFINED POLYLINE REPRESENTATION (RESPONSE)	3/6	3/6
INQUIRE POLYMARKER FACILITIES (REQUEST)	3/6	2/7
INQUIRE POLYMARKER FACILITIES (RESPONSE)	3/6	3/7
INQUIRE PREDEFINED POLYMARKER REPRESENTATION (REQUEST)	3/6	2/8
INQUIRE PREDEFINED POLYMARKER REPRESENTATION (RESPONSE)	3/6	3/8
INQUIRE TEXT FACILITIES (REQUEST)	3/6	2/9
INQUIRE TEXT FACILITIES (RESPONSE)	3/6	3/9
INQUIRE PREDEFINED TEXT REPRESENTATION (REQUEST)	3/6	2/10
INQUIRE PREDEFINED TEXT REPRESENTATION (RESPONSE)	3/6	3/10
INQUIRE FILL AREA FACILITIES (REQUEST)	3/6	2/11
INQUIRE FILL AREA FACILITIES (RESPONSE)	3/6	3/11
INQUIRE PREDEFINED FILL AREA REPRESENTATION (REQUEST)	3/6	2/12
INQUIRE PREDEFINED FILL AREA REPRESENTATION (RESPONSE)	3/6	3/12
INQUIRE PATTERN FACILITIES (REQUEST)	3/6	2/13
INQUIRE PATTERN FACILITIES (RESPONSE)	3/6	3/13
INQUIRE PREDEFINED PATTERN REPRESENTATION (REQUEST)	3/6	2/14
INQUIRE PREDEFINED PATTERN REPRESENTATION (RESPONSE)	3/6	3/14
INQUIRE COLOUR FACILITIES (REQUEST)	3/6	2/15
INQUIRE COLOUR FACILITIES (RESPONSE)	3/6	3/15
INQUIRE PREDEFINED COLOUR REPRESENTATION (REQUEST)	3/7	2/0
INQUIRE PREDEFINED COLOUR REPRESENTATION (RESPONSE)	3/7	3/0
INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (REQUEST)	3/7	2/1
INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES(RESPONSE)	3/7	3/1
INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (REQUEST)	3/7	2/2
INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (RESPONSE)	3/7	3/2
INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICE (REQUEST)	3/7	2/3
INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICE (RESPONSE)	3/7	3/3
INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (REQUEST)	3/7	2/4
INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (RESPONSE)	3/7	3/4

Primitives	Opcode Byte 1	Opcode Byte 2
INQUIRES OF SEGMENT STATE LIST		
INQUIRE SET OF ASSOCIATED WORKSTATIONS (REQUEST)	3/7	2/5
INQUIRE SET OF ASSOCIATED WORKSTATIONS (RESPONSE)	3/7	3/5
INQUIRE SEGMENT ATTRIBUTES (REQUEST)	3/7	2/13
INQUIRE SEGMENT ATTRIBUTES (RESPONSE)	3/7	3/13
INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTE (REQUEST)	3/7	2/14
INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTE (RESPONSE)	3/7	3/14
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (REQUEST)	3/7	2/15
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (RESPONSE)	3/7	3/15
PIXEL INQUIRIES		
INQUIRE PIXEL (REQUEST)	3/7	2/8
INQUIRE PIXEL (RESPONSE)	3/7	3/8
INQUIRE PIXEL ARRAY (REQUEST)	3/7	2/9
INQUIRE PIXEL ARRAY (RESPONSE)	3/7	3/9
INQUIRE PIXEL ARRAY DIMENSION (REQUEST)	3/7	2/10
INQUIRE PIXEL ARRAY DIMENSION (RESPONSE)	3/7	3/10
PROTOCOL DESCRIPTORS		
SET DOMAIN RING	3/2	2/4
SET COLOUR HEADER	3/2	2/8
SET COORDINATE PRECISION	3/2	2/9
SET REAL PRECISION	3/2	3/9
SET COLOUR INDEX PRECISION	3/2	2/10

8.2 Coding of the Primitives

The notational conventions used are:

<symbols> = 1 occurence

<symbols>(n) = n or more occurences (with n $\geq$ 1)

<symbols>(=n)= n occurences (with n $\geq$ 1)

<symbols>(0) = optional, 0 or more occurences

[comment] = explanation of a production

<x: y> = construction x with meaning y

8.2.1 WORKSTATION MANAGEMENT PRIMITIVES

8.2.1.1 OPEN WORKSTATION

<OPEN_WORKSTATION_opcode : 3/2, 3/0>

<identifier : workstation_identifier>

8.2.1.2 CLOSE WORKSTATION

<CLOSE_WORKSTATION_opcode : 3/2, 3/1>

<identifier : workstation_identifier>

8.2.1.3 ACTIVATE WORKSTATION

<ACTIVATE_WORKSTATION_opcode : 3/2, 3/2>

<identifier : workstation_identifier>

8.2.1.4 DEACTIVATE WORKSTATION

<DEACTIVATE_WORKSTATION_opcode : 3/2, 3/3>

<identifier : workstation_identifier>

8.2.1.5 CLEAR WORKSTATION

<CLEAR_WORKSTATION_opcode : 3/2, 2/0>

<identifier : workstation_identifier>

8.2.1.6 SET DEFAULTS

<SET_DEFAULTS_opcode : 3/2, 2/5>

8.2.1.7. GRAPHICS ESCAPE PRIMITIVES

8.2.1.7.1 GDS ESCAPE 1

<ESCAPE_1 _opcode : 3/2, 2/11>

<integer : escape_identifier>

<record : data_record>

with

<integer : escape_identifier> =

<integer : ≥ 0> [for used in this standard]

|<integer : < 0> [for private use]

8.2.1.7.2 GDS ESCAPE 2 (REQUEST)

<ESCAPE_2_(REQUEST)_opcode : 3/2, 2/12>

<integer : escape_identifier> [see 8.2.1.7.1]

<record : output_data_record>

8.2.1.7.3 GDS ESCAPE 2 (RESPONSE)

<ESCAPE_(RESPONSE)_opcode : 3/2, 3/12>

<integer : escape_identifier> [see 8.2.1.7.1]

<record : input_data_record>

8.2.1.8 MESSAGE

<MESSAGE_opcode : 3/2, 2/13>

<identifier : workstation_identifier>

<string : message>

8.2.1.9 INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (REQUEST)

<INQUIRE_GRAPHICS_CONFIGURATION_IDENTIFICATION_(REQUEST)_opcode : 3/2, 2/7>

8.2.1.10 INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (RESPONSE)

<INQUIRE_GRAPHICS_CONFIGURATION_IDENTIFICATION_(RESPONSE)_opcode: 3/2, 3/7>

<identifier : graphics_configuration_name>

<integer : no._of_workstations>

<<identifier : workstation_identifier>

<enumerated : workstation_category>

<enumerated : level>> (n)

with:

<enumerated: workstation_category> =

<enumerated: 0> [OUTPUT]
|<enumerated: 1> [INPUT]
|<enumerated: 2> [OUTIN]
|<enumerated: 3> [WISS]
|<enumerated: >3> [reserved]

<enumerated: level> =

<enumerated: 0> [Level L0a]
|<enumerated: 1> [Level L0b]
|<enumerated: 2> [Level L0c]
|<enumerated: 3> [Level L1a]
|<enumerated: 4> [Level L1b]
|<enumerated: 5> [Level L1c]
|<enumerated: 6> [Level L2a]
|<enumerated: 7> [Level L2b]
|<enumerated: 8> [Level L2c]
|<enumerated: >8> [reserved]

8.2.2 OUTPUT WORKSTATION PRIMITIVES

8.2.2.1 OUTPUT DRAWING PRIMITIVES

8.2.2.1.1 POLYLINE

<POLYLINE_opcode : 2/0>

<point : point_list> (2)

8.2.2.1.2 POLYMARKER

<POLYMARKER_opcode: 2/2>

<point : point_list> (1)

8.2.2.1.3 FILL AREA

<FILL_AREA_opcode : 2/1>

<point : point_list> (3)

8.2.2.1.4 TEXT

<TEXT_opcode : 2/4>

<point : text_position>

<string : character_string>

8.2.2.1.5 CELL ARRAY

<CELL_ARRAY_opcode : 2/3>

<point : parallelogram (P, Q, R)> (= 3)

<integer : no._of_cells_in_P-R_direction> [value M]

<integer : no._of_cells_in_Q-R_direction> [value N]

<colour_index_list: cell_colour_index_list>

8.2.2.1.6 GENERALIZED DRAWING PRIMITIVE (GDP)

<GENERALIZED_DRAWING_PRIMITIVE_opcode : 2/5>

<integer : GDP_identifier>

with

<integer : < 0> [for private use]

|<integer : 0> [RECTANGLE]

|<integer : 1> [CIRCLE]

|<integer : 2> [CIRCULAR ARC 3 POINT]

|<integer : 3> [CIRCULAR ARC 3 POINT CHORD]

|<integer : 4> [CIRCULAR ARC 3 POINT PIE]

|<integer : 5> [CIRCULAR ARC CENTRE]

|<integer : 6> [CIRCULAR ARC CENTRE CHORD]

|<integer : 7> [CIRCULAR ARC CENTRE PIE]

|<integer : 8> [ELLIPSE]

|<integer : 9> [ELLIPTIC ARC]

|<integer : 10> [ELLIPTIC ARC CHORD]

|<integer : 11> [ELLIPTIC ARC PIE]

|<integer : 12> [SPLINE]

|<integer : >12> [reserved]

case <integer : GDP_identifier > <0 [private use]

case <integer : GDP_identifier > =0 [RECTANGLE] then

<point : point_list > (=2)

case <integer : GDP_identifier > =1 [CIRCLE] then
<point : centre>
<size_value : radius>

case <integer : GDP_identifier > =2 [CIRCULAR ARC 3 POINT] then
<point : starting_point>
<point : intermediate_point>
<point : ending_point>

```
case   <integer :  GDP_identifier > =3    [CIRCULAR ARC 3 POINT CHORD]   then
         <point :  starting_point>
         <point :  intermediate_point>
         <point :  ending_point>

case   <integer :  GDP_identifier > =4    [CIRCULAR ARC 3 POINT PIE]     then
         <point :  starting_point>
         <point :  intermediate_point>
         <point :  ending_point>

case   <integer :  GDP_identifier > =5    [CIRCULAR ARC CENTRE]          then
         <point :  centre>
         <size_value:  radius>
         <point :  start_vector>
         <point :  end_vector>

case   <integer :  GDP_identifier > =6    [CIRCULAR ARC CENTRE CHORD]    then
         <point :  centre>
         <size_value:  radius>
         <point :  start_vector>
         <point :  end_vector>

case   <integer :  GDP_identifier > =7    [CIRCULAR ARC CENTRE PIE]      then
         <point :  centre>
         <size_value:  radius>
         <point :  start_vector>
         <point :  end_vector>

case   <integer :  GDP_identifier > =8    [ELLIPSE]                     then
         <point :  centre_point>
         <point :  endpoints> (=2)        [(x1,y1) and (x2, y2)]

case   <integer : GDP_identifier > =9     [ELLIPTIC ARC]                then
         <point :  centre_point>
         <point :  endpoints> (=2)        [(x1,y1) and (x2, y2)]
         <real  :  T_start, T_end> (=2)

case   <integer : GDP_identifier > =10    [ELLIPTIC ARC CHORD]          then
         <point :  centre_point>
         <point :  endpoints> (=2)        [(x1,y1) and (x2, y2)]
         <real  :  T_start, T_end> (=2)

case   <integer : GDP_identifier > =11    [ELLIPTIC ARC PIE]            then
         <point :  centre_point>
         <point :  endpoints> (=2)        [(x1,y1) and (x2, y2)]
         <real  :  T_start, T_end> (=2)

case   <integer :  GDP_identifier > =12   [SPLINE]                      then
         <point :  point_list> (3)
```

8.2.2.2 OUTPUT PRIMITIVES RELATED TO DISPLAY ELEMENT ATTRIBUTES

8.2.2.2.1 SET POLYLINE REPRESENTATION

<SET_POLYLINE_REPRESENTATION_opcode : 3/1, 3/11>

<identifier : workstation_identifier>

<index : polyline_index>

<integer : line_type>

<real : line_width_scale_factor>

<colour_index : polyline_colour_index>

with

<integer : line_type> =

<integer : 0>	[SOLID]
|<integer : 1>	[DASHED]
|<integer : 2>	[DOTTED]
|<integer : 3>	[DASHED-DOTTED]
|<integer : >3>	[reserved]
|<integer : negative>	[private line type]

8.2.2.2.2 SET POLYLINE INDEX

<SET_POLYLINE_INDEX_opcode : 3/1, 2/3>

<index : polyline_index>

8.2.2.2.3 SET LINE TYPE

<SET_LINE_TYPE_opcode : 3/1, 2/2>

<integer : line_type> [see 8.2.2.2.1]

8.2.2.2.4 SET LINE WIDTH SCALE FACTOR

<SET_LINE_WIDTH_SCALE_FACTOR_opcode : 3/1, 2/1>

<real : line_width_scale_factor>

8.2.2.2.5 SET POLYLINE COLOUR INDEX

<SET_POLYLINE_COLOUR_INDEX_opcode : 3/1, 2/0>

<colour_index : polyline_colour_index>

8.2.2.2.6 SET POLYMARKER REPRESENTATION

<SET_POLYMARKER_REPRESENTATION_opcode : 3/1, 3/13>

<identifier : workstation_identifier>

<index : polymarker_index>

<integer : marker_type>

<real : marker_size_scale-factor>

<colour_index : polymarker_colour_index>

with

<integer : marker_type> =

<integer : 0>	[DOT]
|<integer : 1>	[PLUS SIGN]
|<integer : 2>	[ASTERISK]
|<integer : 3>	[CIRCLE]
|<integer : 4>	[DIAGONAL CROSS]
|<integer : >4>	[reserved]
|<integer : negative>	[private marker type]

8.2.2.2.7 SET POLYMARKER INDEX

<SET_POLYMARKER_INDEX_opcode : 3/1, 2/14>

<index : polymarker_index>

8.2.2.2.8 SET MARKER TYPE

<SET_MARKER_TYPE_opcode : 3/1, 2/12>

<integer : marker_type> [see 8.2.2.2.6]

8.2.2.2.9 SET MARKER SIZE SCALE FACTOR

<SET_MARKER_SIZE_SCALE_FACTOR_opcode : 3/1, 2/13>

<real : marker_size_scale_factor>

8.2.2.2.10 SET POLYMARKER COLOUR INDEX

<SET_MARKER_COLOUR_INDEX_opcode : 3/1, 2/11>

<colour_index : polymarker_colour_index>

8.2.2.2.11 SET FILL AREA REPRESENTATION

<SET_FILL_AREA_REPRESENTATION_opcode : 3/1, 3/12>

<identifier : workstation_identifier>

<index : fill_area_index>

<enumerated : fill_area_interior_style>

<integer : fill_area_style_index> [interior style HATCH]

|<integer : fill_area_style_index> [interior style PATTERN]

<colour_index : fill_area_colour_index>

with

<enumerated : fill_area_interior_style> =

<enumerated : 0> [HOLLOW]
|<enumerated : 1> [SOLID]
|<enumerated : 2> [PATTERN]
|<enumerated : 3> [HATCH]
|<enumerated : >3> [reserved]

if <fill_area_interior_style> is [HATCH], then

<integer : fill_area_style_index> =

<integer : 0> [vertical lines]
|<integer : 1> [horizontal lines]
|<integer : 2> [45 degree lines]
|<integer : 3> [-45 degree lines]
|<integer : 4> [crossed lines, vertical and horizontal]
|<integer : 5> [crossed lines, 45 and -45 degrees]
|<integer : >5> [reserved]
|<integer : negative> [private hatch style]

8.2.2.2.12 SET FILL AREA INDEX

<SET_FILL_AREA_INDEX_opcode : 3/1, 2/10>

<index : fill_area_index>

8.2.2.2.13 SET FILL AREA INTERIOR STYLE

<SET_FILL_AREA_INTERIOR_STYLE opcode : 3/1, 2/5>

<enumerated : fill_area_interior_style> [see 8.2.2.2.11]

8.2.2.2.14 SET FILL AREA COLOUR INDEX

<SET_FILL_AREA_COLOUR_INDEX_opcode : 3/1, 2/4>

<colour_index : fill_area_colour_index>

8.2.2.2.15 SET FILL AREA STYLE INDEX

<SET_FILL_AREA_STYLE_INDEX_opcode : 3/1, 2/6>

<integer : fill_area_style_index> [interior style HATCH]

|<index : fill_area_style index> [interior style PATTERN]

with

<integer : hatch_type> =

<integer : 0> [vertical lines]
|<integer : 1> [horizontal lines]
|<integer : 2> [45 degree lines]
|<integer : 3> [-45 degree lines]
|<integer : 4> [crossed lines, vertical and horizontal]
|<integer : 5> [crossed lines, 45 and -45 degrees]
|<integer : >5> [reserved]
|<integer : negative> [private hatch style]

8.2.2.2.16 SET PATTERN REPRESENTATION

<SET_PATTERN_REPRESENTATION_opcode : 3/1, 3/15>

<identifier : workstation_identifier>

<index : pattern_index>

<integer : delta X> [DX]

<integer : delta Y> [DY]

<colour_index : pattern_cell_colour_index_list> (DX x DY)

8.2.2.2.17 SET PATTERN REFERENCE POINT

<SET_PATTERN_REFERENCE_POINT_opcode : 3/1, 2/9>

<point : reference_point>

8.2.2.2.18 SET PATTERN VECTORS

<SET_PATTERN_VECTORS_opcode : 3/1, 2/8>

<point : height_vector>

<point : width_vector>

8.2.2.2.19 SET TEXT REPRESENTATION

<SET_TEXT_REPRESENTATION_opcode : 3/1, 3/14>

<identifier : workstation_identifier>

<index : text_index>

<integer : text_font>

<enumerated : text_precision>

<real : character_expansion_factor>

<real : character_spacing>

<colour_index : text_colour_index>

with

<enumerated : text_precision> =

<enumerated : 0> [precision STRING]
|<enumerated : 1> [precision CHARACTER]
|<enumerated : 2> [precision STROKE]
|<enumerated : >2> [reserved]

<integer : text_font> =

<integer : 0> [font for ISO 6937/2 character set]
|<integer : >0> [reserved]
|<integer : <0> [private]

8.2.2.2.20 SET TEXT INDEX

<SET_TEXT_INDEX_opcode : 3/1, 3/6>

<index : text_index>

8.2.2.2.21 SET TEXT FONT AND PRECISION

<SET_TEXT_FONT_AND_PRECISION_opcode : 3/1, 3/4>

<integer : text_font> [see 8.2.2.2.19]

<enumerated : text_precision> [see 8.2.2.2.19]

8.2.2.2.22 SET CHARACTER EXPANSION FACTOR

<SET_CHARACTER_EXPANSION_FACTOR_opcode : 3/1, 3/0>

<real : character_expansion_factor>

8.2.2.2.23 SET CHARACTER SPACING

<SET_CHARACTER_SPACING_opcode : 3/1, 3/3>

<real : character_spacing>

8.2.2.2.24 SET TEXT COLOUR INDEX

<SET_TEXT_COLOUR_INDEX_opcode : 3/1, 2/15>

<colour_index : text_colour_index>

8.2.2.2.25 SET TEXT PATH

<SET_TEXT_PATH_opcode : 3/1, 3/2>

<enumerated : text_path>

with

<enumerated : text_patch> =

<enumerated : 0>	[RIGHT]
|<enumerated : 1>	[LEFT]
|<enumerated : 2>	[UP]
|<enumerated : 3>	[DOWN]
|<enumerated : >3>	[reserved]

8.2.2.2.26 SET CHARACTER VECTORS

<SET_CHARACTER_VECTORS_opcode : 3/1, 3/1>

<point : height_vector>

<point : width_vector>

8.2.2.2.27 SET TEXT ALIGNMENT

<SET_TEXT_ALIGNMENT_opcode : 3/1, 3/5>

<enumerated : horizontal_alignment>

<enumerated : vertical_alignment>

with

<enumerated : horizontal_alignment> =

<enumerated : 0>	[NORMAL]
|<enumerated : 1>	[LEFT]
|<enumerated : 2>	[CENTRE]
|<enumerated : 3>	[RIGHT]
|<enumerated : >3>	[reserved]

<enumerated : vertical_alignment> =

<enumerated : 0>	[NORMAL]
|<enumerated : 1>	[TOP]
|<enumerated : 2>	[CAP]
|<enumerated : 3>	[HALF]
|<enumerated : 4>	[BASE]
|<enumerated : 5>	[BOTTOM]
|<enumerated : >5>	[reserved]

8.2.2.2.28 SET COLOUR REPRESENTATION

<SET_COLOUR_REPRESENTATION_opcode : 3/2, 3/8>

<identifier : workstation_identifier>

<colour_index : starting_entry_in_colour_table>

<colour_direct: colour_data_list> (1)

8.2.2.2.29 SET ASPECT SOURCE FLAGS

<SET_ASPECT_SOURCE_FLAGS_opcode: 3/1, 2/7>

<enumerated : line_type_ASF>

<enumerated : line_width_scale_factor_ASF>

<enumerated : polyline_colour_ASF>

<enumerated : marker_type_ASF>

<enumerated : marker_size_scale_factor_ASF>

<enumerated : polymarker_colour_ASF>

<enumerated : text_font_and_precision_ASF>

<enumerated : character expansion factor ASF>

<enumerated : character spacing ASF>

<enumerated : text colour ASF>

<enumerated : fill area interior style ASF>

<enumerated : fill area style index ASF>

<enumerated : fill area-colour ASF>

with

<enumerated : 0> [INDIVIDUAL]

|<enumerated : 1> [BUNDLED]
|<enumerated : >1> [reserved]

8.2.2.2.30 SET PICK IDENTIFIER

<SET_PICK_IDENTIFIER_opcode : 3/3, 2/13>

<identifier : pick_identifier>

8.2.2.3 TRANSFORMATION PRIMITIVES

8.2.2.3.1 SET WORKSTATION WINDOW

<SET_WORKSTATION_WINDOW_opcode : 3/2, 2/1>

<identifier : workstation_identifier>

<point : window_limits> (= 2)

8.2.2.3.2 SET WORKSTATION VIEWPORT

<SET_WORKSTATION_VIEWPORT_opcode : 3/2, 2/2>

<identifier : workstation_identifier>

<real : viewport_limits> (= 4) [$x_{min}<x_{max}$, $y_{min}<y_{max}$]

8.2.2.4 CLIPPING PRIMITIVES

8.2.2.4.1 SET CLIPPING RECTANGLE

<SET_CLIPPING_RECTANGLE_opcode : 3/2, 2/3>

<point : clipping_rectangle_limits> (= 2)

8.2.2.5 CONTROL PRIMITIVES

8.2.2.5.1 UPDATE WORKSTATION

<UPDATE_WORKSTATION_opcode : 3/2, 3/4>

<identifier : workstation_identifier>

<enumerated : update_regeneration_flag>

with

<enumerated : update regeneration_flag> =

<enumerated : 0>	[PERFORM]
|<enumerated : 1>	[POSTPONE]

8.2.2.5.2 SET DEFERRAL STATE

<SET_DEFERRAL_STATE_opcode : 3/2, 3/5>

<identifier : workstation_identifier>

<enumerated : deferral_mode>

<enumerated : implicit_regeneration_flag

with

<enumerated : deferral_mode> =

<enumerated : 0>	[ASAP]
|<enumerated : 1>	[BNIL]
|<enumerated : 2>	[BNIG]
|<enumerated : 3>	[ASTI]
|<enumerated : >3>	[reserved]

<enumerated : implicit_regeneration_flag> =

<enumerated : 0>	[SUPPRESSED]
|<enumerated : 1>	[ALLOWED]
|<enumerated : >1>	[reserved]

8.2.2.5.3 EMERGENCY CLOSE

<EMERGENCY_CLOSE_opcode: 3/2, 2/6>

8.2.2.5.4 ERROR DETECTED (RESPONSE)

<ERROR_DETECTED_(RESPONSE)_opcode: 3/2, 3/6>

8.2.3 SEGMENT RELATED PRIMITIVES

8.2.3.1 WDSS RELATED PRIMITIVES

8.2.3.1.1 CREATE SEGMENT

<CREATE_SEGMENT_opcode : 3/3, 2/0>

<identifier : segment_name>

8.2.3.1.2 CLOSE SEGMENT

<CLOSE_SEGMENT_opcode : 3/3, 2/1>

8.2.3.1.3 RENAME SEGMENT

<RENAME_SEGMENT_opcode : 3/3, 2/2>

<identifier : old_segment_name>

<identifier : new_segment_name>

8.2.3.1.4 DELETE SEGMENT FROM WORKSTATION

<DELETE_SEGMENT_FROM_WORKSTATION_opcode : 3/3, 2/3>

<identifier : workstation_identifier>

<identifier : segment_name>

8.2.3.1.5 DELETE SEGMENT

<DELETE_SEGMENT_opcode : 3/3, 2/8>

<identifier : segment_name>

8.2.3.1.6 REDRAW ALL SEGMENTS ON WORKSTATION

<REDRAW_ALL_SEGMENTS_ON_WORKSTATION_opcode : 3/3, 2/9>

<identifier : workstation_identifier>

8.2.3.1.7 SET HIGHLIGHTING

<SET_HIGHLIGHTING_opcode : 3/3, 2/6>

<identifier : segment_name>

<enumerated : highlighting_flag>

with

<enumerated : highlighting_flag> =

<enumerated : 0>	[NOT HIGHLIGHTED]
|<enumerated : 1>	[HIGHLIGHTED]
|<enumerated : >1>	[reserved]

8.2.3.1.8 SET VISIBILITY

<SET_VISIBILITY_opcode : 3/3, 2/7>

<identifier : segment_name>

<enumerated : visibility_flag>

with

<enumerated : visibility_flag> =

<enumerated : 0>	[VISIBLE]
|<enumerated : 1>	[INVISIBLE]
|<enumerated : >1>	[reserved]

8.2.3.1.9 SET SEGMENT TRANSFORMATION

<SET_SEGMENT_TRANSFORMATION_opcode : 3/3, 2/5>

<identifier : segment_name>

<matrix : transformation_matrix>

8.2.3.1.10 SET SEGMENT PRIORITY

<SET_SEGMENT_PRIORITY_opcode : 3/3, 2/12>

<identifier : segment_name>

<real : segment_priority>

8.2.3.1.11 SET DETECTABILITY

<SET_DETECTABILITY_opcode : 3/3, 2/14>

<identifier : segment_name>

<enumerated : detectability_flag>

with

<enumerated : detectability_flag> =

<enumerated : 0>	[DETECTABLE]
|<enumerated : 1>	[UNDETECTABLE]
|<enumerated : >1>	[reserved]

8.2.3.2 WISS RELATED PRIMITIVES

8.2.3.2.1 ASSOCIATE SEGMENT WITH WORKSTATION

<ASSOCIATE_SEGMENT_WITH_WORKSTATION_opcode : 3/3, 2/10>

<identifier : workstation_identifier>

<identifier : segment_name>

8.2.3.2.2 COPY SEGMENT TO WORKSTATION

<COPY_SEGMENT_TO_WORKSTATION_opcode : 3/3, 2/11>

<identifier : workstation_identifier>

<identifier : segment_name>

8.2.3.2.3 INSERT SEGMENT

<INSERT_SEGMENT_opcode : 3/3, 2/4>

<identifier : segment_name>

<matrix : transformation_matrix>

8.2.4 INPUT PRIMITIVES

8.2.4.1 INITIALIZE LOGICAL INPUT DEVICE

<INITIALIZE_LOGICAL_INPUT_DEVICE_opcode : 3/4, 2/0>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class>

with

<enumerated : logical_input_device_class> =

<enumerated : 0> [LOCATOR]
|<enumerated : 1> [STROKE]
|<enumerated : 2> [VALUATOR]
|<enumerated : 3> [CHOICE]
|<enumerated : 4> [PICK]
|<enumerated : 5> [STRING]
|<enumerated : >5> [reserved]

case

<enumerated : logical_input_device_class> = 0 [LOCATOR] then:

<integer : logical_input_device_number>
<point : initial_position>
<integer : prompt_and_echo_type>
<point : lower_left_corner_of_the _echo_area>
<point : upper_right_corner_of_the_echo_area>

with

<integer : prompt_and_echo-type> =

<integer : 1> [implementation_dependent]
|<integer : 2> [CROSSHAIR]
|<integer : 3> [TRACKING CROSS]
|<integer : 4> [RUBBER_BAND]
|<integer : 5> [RECTANGLE]
|<integer : 6> [DIGITAL_REPRESENTATION]
|<integer : >6> [reserved]
|<integer : <0> [LOCATOR device dependent]

case

<enumerated : logical_input_device_class> = 1 [STROKE] then:

<integer : logical_input_device_number>
<integer : number_of_points_in_initial_stroke> [value N]
<point : sequence_of_points_in_initial_stroke> (N)
<integer : prompt_and_echo_type>
<point : lower_left_corner_of_the_echo_area>

```
        <point   : upper_right_corner_of_the_echo_area>
        <integer : input_buffer_size>
        <integer : initial_buffer_editing_position>

           with

           <integer : prompt_and_echo_type> =

              <integer : 1>    [implementation_dependent]
             |<integer : 2>    [DIGITAL_REPRESENTATION]
             |<integer : 3>    [MARKER_AT_EACH_POINT_OF_THE_STROKE]
             |<integer : 4>    [LINE_JOINING_POINTS_OF_THE_STROKE]
             |<integer : >4>   [reserved]
             |<integer : ≤ 0>  [STROKE_device_dependent]

case

   <enumerated : logical_input_device_class > = 2 [VALUATOR]  then:

      <integer : logical_input_device_number>
      <real    : initial_valuator_value>
      <integer : prompt_and_echo_type>
      <point   : lower_left_corner_of_the_echo_area>
      <point   : upper_right_corner_of_the_echo_area>
      <real    : low_value>
      <real    : high_value>

         with

         <integer : prompt_and_echo_type> =

            <integer : 1>    [implementation_dependent]
           |<integer : 2>    [GRAPHICAL_REPRESENTATION]
           |<integer : 3>    [DIGITAL_REPRESENTATION]
           |<integer : >3>   [reserved]
           |<integer : ≤ 0>  [VALUATOR_device_dependent]

case

   <enumerated : logical_input_device_class > = 3 [CHOICE]    then:

      <integer : logical_input_device_number>
      <integer : initial_choice_number>
      <integer : prompt_and_echo_type>

   case

      <integer : prompt_and_echo_type> = 1 [implementation_dependent] then:

         <point : lower_left_corner_of_the_echo_area>
         <point : upper_right_corner_of_the_echo_area>
```

```
case
<integer : prompt_and_echo_type> = 2 [PHYSICAL_PROMPT_AND_ECHO] then:

   <point      : lower_left_corner_of_the_echo_area>
   <point      : upper_right_corner_of_the_echo_area>
   <integer    : number_of_alternatives>                [value N]
   <enumerated : prompt_array>  (N)

      with

      <enumerated : prompt_array> =

         <enumerated : 0>   [ON]
        |<enumerated : 1>   [OFF]
        |<enumerated :>1>   [reserved]

case
<integer : prompt_and_echo_type> = 3 [CHOICE_NUMBER]    then:

   <point   : lower_left_corner_of_the_echo_area>
   <point   : upper_right_corner_of_the_echo_area>
   <integer : number_of_choice_string>                  [value N]
   <string  : list_of_strings> (N)

case
<integer : prompt_and_echo_type> = 4 [KEYBOARD_CHOICE_NUMBER] then:

   <point   : lower_left_corner_of_the_echo_area>
   <point   : upper_right_corner_of_the_echo_area>
   <integer : number_of_choice_string>                  [value N]
   <string  : list_of_strings> (N)

case
<integer : prompt_and_echo_type> = 5 [SEGMENT_NAME] then:

   <point      : lower_left_corner_of_the_echo_area>
   <point      : upper_right_corner_of_the_echo_area>
   <identifier : segment_name>

case
<integer : prompt_and_echo_type> > 5 [reserved]         then:

   <point : lower_left_corner_of_the_echo_area>
   <point : upper_right_corner_of_the_echo_area>


case
<integer : prompt_and_echo_type> <0 [CHOICE_devive_dependent] then:

   <point : lower_left_corner_of_the_echo_area>
   <point : upper_left_corner_of_the_echo_area>
```

```
case

   <enumerated :  logical_input_device_class > = 4 [PICK] then:

      <integer    : logical_input_device_number>
      <enumerated : initial_status>
      <identifier : initial_segment_name>
      <identifier : initial_pick_identifier>
      <integer    : prompt_and_echo_type>
      <point      : lower_left_corner_of_the_echo_area>
      <point      : upper_right_corner_of_the_echo_area>

         with

         <enumerated : initial_status> =

            <enumerated : 0>  [PICK]
           |<enumerated : 1>  [NO PICK]
           |<enumerated : >1> [reserved]

         <integer : prompt_and_echo_type> =

            <integer : 1>   [implementation_dependent]
           |<integer : 2>   [CONTIGUOUS_SET_OF_PRIMITIVES]
           |<integer : 3>   [WHOLE_SEGMENT]
           |<integer : >3>  [reserved]
           |<integer : < 0> [PICK_device_dependent]

case

   <enumerated : logical_input_device_class = 5 [STRING] then:

      <integer : logical_input_device_number>
      <string  : initial_string>
      <integer : prompt_and_echo_type>
      <point   : lower_left_corner_of_the_echo_area>
      <point   : upper_right_corner_of_the_echo_area>
      <integer : input_buffer_size>
      <integer : initial_cursor_position>

         with

         <integer : prompt_and_echo_type> =

            <integer : 1>   [implementation_dependent]
           |<integer : 2>   [CURRENT_STRING]
           |<integer : >2>  [reserved]
           |<integer : < 0> [STRING_device_dependent]
```

8.2.4.2 SET LOGICAL INPUT DEVICE MODE

<SET_LOGICAL_INPUT_DEVICE_MODE opcode : 3/4, 2/1>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

<enumerated : operating_mode>

<enumerated : echo_switch>

with

<enumerated : operating_mode> =

<enumerated : 0> [REQUEST]
|<enumerated : 1> [SAMPLE]
|<enumerated : 2> [EVENT]
|<enumerated : >2> [reserved]

<enumerated : echo_switch> =

<enumerated : 0> [ECHO]
|<enumerated : 1> [NO ECHO]
|<enumerated : >1> [reserved]

8.2.4.3 REQUEST LOGICAL INPUT DEVICE (REQUEST)

<REQUEST_LOGICAL_INPUT DEVICE_(REQUEST)_opcode : 3/4, 2/2>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

8.2.4.4 REQUEST LOGICAL INPUT DEVICE (RESPONSE)

```
<REQUEST_LOGICAL_INPUT_DEVICE_(RESPONSE)_opcode : 3/4, 3/2>
   <identifier    : workstation_identifier>
   <enumerated    : logical_input_device_class>          [see 8.2.4.1]
   <integer       : logical_input_device_number>
   <combined      : LID_values>
      with
      <combined    : LID_values> =
         case
         <enumerated : logical_input_device_class> = 0 [LOCATOR] then:
            <enumerated : status> = 0    [NONE]
          |<<enumerated : status> = 1     [OK ]
            <point      : position_point>>
         case
         <enumerated : logical_input_device_class> = 1 [STROKE] then:
            <enumerated : status> = 0    [NONE]
          |<<enumerated : status> = 1     [OK ]
            <integer    : number_of_points>
            <point      : point_list>>
         case
         <enumerated : logical_input_device_class> = 2 [VALUATOR] then:
            <enumerated : status> = 0    [NONE]
          |<<enumerated : status> = 1     [OK]
            <real       : valuator value>>
         case
         <enumerated : logical_input_device_class> = 3 [CHOICE] then:
            <enumerated : status> = 0    [NONE]
          |<<enumerated : status> = 1     [OK ]
            <integer    : choice_value>>
```

case

<enumerated : logical_input_device_class> = 4 [PICK] then:

<enumerated : status> = 0 [NONE]
|<<enumerated : status> = 1 [OK]
<identifier : segment_name>
<identifier : pick_identifier>>

case

<enumerated : logical_input_device_class> = 5 [STRING] then:

<enumerated : status> = 0 [NONE]
|<<enumerated : status> = 1 [OK]
<string : string_value>>

8.2.4.5 SAMPLE LOGICAL INPUT DEVICE (REQUEST)

<SAMPLE_LOGICAL_INPUT_DEVICE_(REQUEST)_opcode : 3/4, 2/3>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

8.2.4.6 SAMPLE LOGICAL INPUT DEVICE (RESPONSE)

<SAMPLE_LOGICAL_INPUT_DEVICE_(RESPONSE)_opcode : 3/4, 3/3>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

<combined : LID_values> see 8.2.4.4

8.2.4.7 AWAIT EVENT (REQUEST)

<AWAIT_EVENT_REQUEST_opcode : 3/4, 2/5>

<real : time_out>

8.2.4.8 AWAIT EVENT (RESPONSE)

<AWAIT_EVENT_(RESPONSE)_opcode : 3/4, 3/5>

<enumerated : status>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

with

<enumerated : status> =

<enumerated : 0> [NONE]
| <enumerated : 1> [NOT EMPTY]
| <enumerated : 2> [OVERFLOW]
| <enumerated : >2> [reserved]

8.2.4.9 GET LOGICAL INPUT DEVICE (REQUEST)

<GET_LOGICAL_INPUT_DEVICE_(REQUEST)_opcode : 3/4, 2/4>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

8.2.4.10 GET LOGICAL INPUT DEVICE (RESPONSE)

<GET_LOGICAL_INPUT_DEVICE_(RESPONSE)_opcode : 3/4, 3/4>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

<combined : LID_values> [see 8.2.4.4]

<real : time_stamp>

8.2.4.11 FLUSH DEVICE EVENTS

<FLUSH_DEVICE_EVENTS_opcode : 3/4, 2/6>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

8.2.5 INQUIRE PRIMITIVES

8.2.5.1 INQUIRE PRIMITIVES FOR WORKSTATION STATE LIST

8.2.5.1.1 INQUIRE WORKSTATION STATE (REQUEST)

<INQUIRE_WORKSTATION_STATE_(REQUEST)_opcode : 3/5, 2/0>

<identifier : workstation_identifier>

8.2.5.1.2 INQUIRE WORKSTATION STATE (RESPONSE)

INQUIRE_WORKSTATION_STATE_(RESPONSE)_opcode : 3/5, 3/0>

<enumerated : workstation_state>

with

<enumerated : workstation_state> =

<enumerated : 0> [ACTIVE]
|<enumerated : 1> [OPEN]
|<enumerated : 2> [CLOSED]
|<enumerated : >2> [reserved]

8.2.5.1.3 INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (REQUEST)

<INQUIRE_WORKSTATION_DEFERRAL_AND_UPDATE_STATES_(REQUEST)_opcode : 3/5, 2/1>

<identifier : workstation_identifier>

8.2.5.1.4 INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (RESPONSE)

<INQUIRE_WORKSTATION_DEFERRAL_AND_UPDATE_STATES_(RESPONSE)_opcode : 3/5, 3/1>

<enumerated : deferral_mode> [see 8.2.5.2]

<enumerated : implicit_regeneration_flag> [see 8.2.5.2]

<enumerated : display_surface_empty>

<enumerated : new_frame_action_necessary_at_update>

with

<enumerated : display_surface_empty> =

<enumerated : 0> [EMPTY]
|<enumerated : 1> [NOT EMPTY]
|<enumerated : >1> [reserved]

<enumerated : new_frame_action_necessary_at_update> =

<enumerated : 0> [NO]
|<enumerated : 1> [YES]
|<enumerated : >1> [reserved]

8.2.5.1.5 INQUIRE POLYLINE INDICES (REQUEST)

<INQUIRE_POLYLINE_INDICES_(REQUEST)_opcode : 3/5, 2/2>

<identifier : workstation_identifier>

8.2.5.1.6 INQUIRE POLYLINE INDICES (RESPONSE)

<INQUIRE_POLYLINE_INDICES_(RESPONSE)_opcode : 3/5, 3/2>

<integer : number_of_bundle_table_entries>

<integer : number_of_defined_indices> (n)

8.2.5.1.7 INQUIRE POLYLINE REPRESENTATION (REQUEST)

<INQUIRE_POLYLINE_REPRESENTATION_(REQUEST)_opcode : 3/5, 2/3>

<identifier : workstation_identifier>

<index : polyline_index>

8.2.5.1.8 INQUIRE POLYLINE REPRESENTATION (RESPONSE)

<INQUIRE_POLYLINE_REPRESENTATION_(RESPONSE)_opcode : 3/5, 3/3>

<integer : line_type>

<size_value : line_width_scale_factor>

<colour_index : polyline_colour_index>

8.2.5.1.9 INQUIRE POLYMARKER INDICES (REQUEST)

<INQUIRE_POLYMARKER_INDICES_(REQUEST)_opcode : 3/5, 2/4>

<identifier : workstation_identifier>

8.2.5.1.10 INQUIRE POLYMARKER INDICES (RESPONSE)

<INQUIRE_POLYMARKER_INDICES_(RESPONSE)_opcode : 3/5, 3/4>

<integer : number_of_bundle_table_entries>

<integer : polymarker_indices> (n)

8.2.5.1.11 INQUIRE POLYMARKER REPRESENTATION (REQUEST)

<INQUIRE_POLYMARKER_REPRESENTATION_(REQUEST)_opcode : 3/5, 2/5>

<identifier : workstation_identifier>

<index : polymarker_index>

8.2.5.1.12 INQUIRE POLYMARKER REPRESENTATION (RESPONSE)

<INQUIRE_POLYMARKER_REPRESENTATION_(RESPONSE)_opcode : 3/5, 3/5>

<identifier : marker_type>

<size_value : marker_size_scale_factor>

<colour_index : polymarker_colour_index>

8.2.5.1.13 INQUIRE TEXT INDICES (REQUEST)

<INQUIRE_TEXT_INDICES_(REQUEST)_opcode : 3/5, 2/6>

<identifier : workstation_identifier>

8.2.5.1.14 INQUIRE TEXT INDICES (RESPONSE)

<INQUIRE_TEXT_INDICES_(RESPONSE)_opcode : 3/5, 3/6>

<integer : number_of_bundle_table_entries>

<integer : number_of_defined_indices>

8.2.5.1.15 INQUIRE TEXT REPRESENTATION (REQUEST)

<INQUIRE_TEXT_REPRESENTATION_(REQUEST)_opcode : 3/5, 2/7>

<identifier : workstation_identifier>

<index : text_index>

8.2.5.1.16 INQUIRE TEXT REPRESENTATION (RESPONSE)

<INQUIRE_TEXT_REPRESENTATION_(RESPONSE)_opcode : 3/5, 3/7>

<integer : text_font> [see 8.2.2.2.19]
<enumerated : text_precision> [see 8.2.2.2.19]

<real : character_expansion_factor>

<real : character_spacing>

<index : text_colour_index>

8.2.5.1.17 INQUIRE TEXT EXTENT (REQUEST)

<INQUIRE_TEXT_EXTENT_(REQUEST)_opcode : 3/5, 2/8>

<identifier : workstation_identifier>

<point : text_position>

<string : character_string>

8.2.5.1.18 INQUIRE TEXT EXTENT (RESPONSE)

<INQUIRE_TEXT_EXTENT_(RESPONSE)_opcode : 3/5, 3/8>

<point : text_extent_parallelogram> (= 3)

8.2.5.1.19 INQUIRE FILL AREA INDICES (REQUEST)

<INQUIRE_FILL_AREA_INDICES_(REQUEST)_opcode : 3/5, 2/9>

<identifier : workstation_identifier>

8.2.5.1.20 INQUIRE FILL AREA INDICES (RESPONSE)

<INQUIRE_FILL_AREA_INDICES_(RESPONSE)_opcode : 3/5, 3/9>

<integer : number_of_bundle_table_entries>

<integer : number_of_defined_indices> (n)

8.2.5.1.21 INQUIRE FILL AREA REPRESENTATION (REQUEST)

<INQUIRE_FILL_AREA_INDICES_(REQUEST)_opcode : 3/5, 2/10>

<identifier : workstation_identifier>

<index : fill_area_index>

8.2.5.1.22 INQUIRE FILL AREA REPRESENTATION (RESPONSE)

<INQUIRE_FILL_AREA_REPRESENTATION_(RESPONSE)_opcode : 3/5, 3/10>

<enumerated : fill_area_interior_style> [see 8.2.2.2.11]

<integer : hatch_type> [in case of hatch_style], [see 8.2.2.2.15]

\<index : pattern_index> [in case of pattern_style]

<colour index : fill_area_colour_index>

8.2.5.1.23 INQUIRE PATTERN INDICES (REQUEST)

<INQUIRE_PATTERN_INDICES_(REQUEST)_opcode : 3/5, 2/11>

<identifier : workstation_identifier>

8.2.5.1.24 INQUIRE PATTERN INDICES (RESPONSE)

<INQUIRE_PATTERN_INDICES_(RESPONSE)_opcode : 3/5, 3/11>

<integer : number_of_pattern_table_entries>

<integer : number_of_defined_indices> (n)

8.2.5.1.25 INQUIRE PATTERN REPRESENTATION (REQUEST)

<INQUIRE_PATTERN_REPRESENTATION_(REQUEST)_opcode : 3/5, 2/12>

<identifier : workstation_identifier>

<index : pattern_index>

8.2.5.1.26 INQUIRE PATTERN REPRESENTATION (RESPONSE)

<INQUIRE_PATTERN_REPRESENTATION_(RESPONSE)_opcode : 3/5, 3/12>

<integer : delta X> [DX]

<integer : delta Y> [DY]

<colour_index_list : pattern_cell_colour_index_list> (DX x DY)

8.2.5.1.27 INQUIRE COLOUR INDICES (REQUEST)

<INQUIRE_COLOUR_INDICES_(REQUEST)_opcode : 3/5, 2/13>

<identifier : workstation_identifier>

8.2.5.1.28 INQUIRE COLOUR INDICES (RESPONSE)

<INQUIRE_COLOUR_INDICES_(RESPONSE)_opcode : 3/5, 3/13>

<integer : number_of_colour_table_entries>

<integer : number_of_defined_indices> (n)

8.2.5.1.29 INQUIRE COLOUR REPRESENTATION (REQUEST)

<INQUIRE_COLOUR_REPRESENTATION_(REQUEST)_opcode : 3/5, 2/14>

<identifier : workstation_identifier>

<colour_index : colour_table_index>

8.2.5.1.30 INQUIRE COLOUR REPRESENTATION (RESPONSE)

<INQUIRE_COLOUR_REPRESENTATION_(RESPONSE)_opcode : 3/5, 3/14>

<colour_direct: colour_data_list> (1)

8.2.5.1.31 INQUIRE WORKSTATION TRANSFORMATION (REQUEST)

<INQUIRE_WORKSTATION_TRANSFORMATION_(REQUEST)_opcode : 3/5, 2/15>

<identifier : workstation_identifier>

8.2.5.1.32 INQUIRE WORKSTATION TRANSFORMATION (RESPONSE)

<INQUIRE_WORKSTATION_TRANSFORMATION_(RESPONSE)_opcode : 3/5, 3/15>

<enumerated : update_state>

with

<enumerated : update_state> =

<enumerated : 0> [NOT PENDING]
!<enumerated : 1> [PENDING]
!<enumerated : >1> [reserved]

case <enumerated : update_state> = 0 then:

<real : workstation_window_limits> (= 4) [current]

<real : workstation_viewport_limits> (= 4) [current]

case <enumerated : update_state> = 1 then:

<real : workstation_window_limits> (= 4) [requested]

<real : workstation_viewport_limits> (= 4) [requested]

8.2.5.1.33 INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (REQUEST)

<INQUIRE_SET_OF_SEGMENT_NAMES_ON_WS_(REQUEST)_opcode : 3/7, 2/11>

<identifier : workstation_identifier>

8.2.5.1.34 INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (RESPONSE)

<INQUIRE_SET_OF_SEGMENT_NAMES_ON_WS_(RESPONSE)_opcode : 3/7, 3/11>

<integer : number_of_segment_names> [value N]

<identifier : name_of_stored_segments> (N)

```
        with

        <enumerated : prompt_array> =

           <enumerated : 0>    [ON]
          |<enumerated : 1>    [OFF]

    case <integer : prompt_and_echo_type> = 3 or 4   [TRACKING CROSS or
                                                      RUBBERBAND]        then:

        <integer : number_of_string>          [N]

        <string  : sequence_of_string>    (= N)

    case <integer : prompt_and_echo_type> = 5        [RECTANGLE]        then:

        <identifier : segment_name>

case <enumerated : logical_input_device> = 4 [PICK]                     then:

    <enumerated    : initial_state>

    <identifier    : initial_segment_name>

    <identifier    : initial_pick_identifier>

    with

    <enumerated : initial_state> =

      <enumerated : 0>    [PICK]
     |<enumerated : 1>    [NO PICK]
     |<enumerated : >1>   [reserved]

 case <enumerated : logical_input_device> = 5 [STRING]  then:

     <string        : initial_string>

     <integer       : initial_buffer_size>

     <integer       : initial_cursor_position>
```

8.2.5.2 INQUIRE PRIMITIVES FOR WORKSTATION DESCRIPTION TABLE

8.2.5.2.1 INQUIRE WORKSTATION CATEGORY (REQUEST)

<INQUIRE_WORKSTATION_CATEGORY_(REQUEST)_opcode : 3/6, 2/0>

<identifier : workstation_identifier>

8.2.5.2.2 INQUIRE WORKSTATION CATEGORY (RESPONSE)

<INQUIRE_WORKSTATION_CATEGORY_(RESPONSE)_opcode : 3/6, 3/0>

<enumerated : workstation_category> [see 8.2.1.10]

8.2.5.2.3 INQUIRE WORKSTATION CLASSIFICATION (REQUEST)

<INQUIRE_WORKSTATION_CLASSIFICATION_(REQUEST)_opcode : 3/6, 2/1>

<identifier : workstation_identifier>

8.2.5.2.4 INQUIRE WORKSTATION CLASSIFICAITON (RESPONSE)

<INQUIRE_WORKSTATION_CLASSIFICATION_(RESPONSE)_opcode : 3/6, 3/1>

<integer : workstation_classification> =

with

<integer : workstation_classification> =

<integer : 0> [vector workstation]
|<integer : 1> [raster workstation]
|<integer : >1> [reserved for future standardization]

8.2.5.2.5 INQUIRE MAXIMUM DISPLAY SURFACE SIZE (REQUEST)

<INQUIRE_MAXIMUM_DISPLAY_SURFACE_SIZE_(REQUEST)_opcode : 3/6, 2/2>

<identifier : workstation_identifier>

8.2.5.2.6 INQUIRE MAXIMUM DISPLAY SURFACE SIZE (RESPONSE)

<INQUIRE_MAXIMUM_DISPLAY_SURFACE_SIZE_(RESPONSE)_opcode : 3/6, 3/2>

<real : horizontal_size_in_meters>

<real : vertical_size_in_meters>

<integer : horizontal_size_in_raster_units>

<integer : vertical_size_in_raster_units>

8.2.5.2.7 INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (REQUEST)

<INQUIRE_DYNAMIC_MODIFICATION_OF_WS_ATTRIBUTES_(REQUEST)_opcode : 3/6, 2/3>

<identifier : workstation_identifier>

8.2.5.2.8 INQUIRE DYNAMIC MODIFICAITON OF WORKSTATION ATTRIBUTES (RESPONSE)

<INQUIRE_DYNAMIC_MODIFICATION_OF_WS_ATTRIBUTES_(RESPONSE)_opcode : 3/6, 3/3>

<enumerated : polyline_bundle_rep.>

<enumerated : polymarker_bundle_rep.>

<enumerated : text_bundle_rep.>

<enumerated : fill_area_bundle_rep.>

<enumerated : pattern_rep.>

<enumerated : colour_rep.>

<enumerated : workstation_transformation>

with

<enumerated : 0> [IRG]

| <enumerated : 1> [IMM]

| <enumerated : >1> [reserved]

8.2.5.2.9 INQUIRE DEFAULT DEFERRAL STATE VALUE (REQUEST)

<INQUIRE_DEFAULT_DEFERRAL_STATE_VALUE_(REQUEST)_opcode : 3/6, 2/4>

<identifier : workstation_identifier>

8.2.5.2.10 INQUIRE DEFAULT DEFERRAL STATE VALUE (RESPONSE)

<INQUIRE_DEFAULT_DEFERRAL_STATE_VALUE_(RESPONSE)_opcode : 3/6, 3/4>

<enumerated : deferral_mode [see 8.2.2.5.2]

<enumerated : implicit_regeneration_mode_flag> [see 8.2.2.5.2]

8.2.5.2.11 INQUIRE POLYLINE FACILITIES (REQUEST)

<INQUIRE_POLYLINE_FACILITIES_(REQUEST)_opcode : 3/6, 2/5>

<identifier : workstation_identifier>

8.2.5.2.12 INQUIRE POLYLINE FACILITIES (RESPONSE)

<INQUIRE_POLYLINE_FACILITIES_(RESPONSE)_opcode : 3/6, 3/5>

<integer : number_of_line_types> [value N]

<integer : sequence_of_line_types> (= N)

<integer : number_of_line_width_scale_factor>

<size_value : default_line_width>_scale_factor

<size_value : minimum_line_width_scale_factor>

<size_value : maximum_line_width_scale_factor>

<integer : number_of_predefined_polyline_indices>

8.2.5.2.13 INQUIRE PREDEFINED POLYLINE REPRESENTATION (REQUEST)

<INQUIRE_PREDEFINED_POLYLINE_REPRESENTATION_(REQUEST)_opcode : 3/6, 2/6>

<identifier : workstation_identifier>

<index : predefined_polyline_index>

8.2.5.2.14 INQUIRE PREDEFINED POLYLINE REPRESENTATION (RESPONSE)

<INQUIRE_PREDEFINED_POLYLINE_REPRESENTATION_(RESPONSE)_opcode : 3/6, 3/6>

<integer : line_type> [see 8.2.2.2.1]

<size value : line_width_scale_factor>

<colour index : polyline_colour_index>

8.2.5.2.15 INQUIRE POLYMARKER FACILITIES (REQUEST)

<INQUIRE_POLYMARKER_FACILITIES_(REQUEST)_opcode : 3/6, 2/7>

<identifier : workstation_identifier>

8.2.5.2.16 INQUIRE POLYMARKER FACILITIES (RESPONSE)

<INQUIRE_POLYMARKER_FACILITIES_(RESPONSE)_opcode : 3/6, 3/7>

<integer : number_of_marker_type> [value N]

<integer : marker_type> (= N) [see 8.2.2.2.6]

<integer : number_of_marker_sizes>

<size_value : default_marker_size_scale_factor>

<size_value : minimum_marker_size_scale_factor>

<size_value : maximum_marker_size_scale_factor>

<integer : number_of_predefined_polymarker_indices>

8.2.5.2.17 INQUIRE PREDEFINED POLYMARKER REPRESENTATION (REQUEST)

<INQUIRE_PREDEFINED_POLYMARKER_REPRESENTATION_(REQUEST)_opcode : 3/6, 2/8>

<identifier : workstation_identifier>

<integer : predefined_polymarker_index>

8.2.5.2.18 INQUIRE PREDEFINED POLYMARKER REPRESENTATION (RESPONSE)

<INQUIRE_PREDEFINED_POLYMARKER_REPRESENTATION_(RESPONSE)_opcode : 3/6, 3/8>

<integer : marker_type> [see 8.2.2.2.6]

<size_value : marker_size_scale_factor>

<colour_index : polymarker_colour_index>

8.2.5.2.19 INQUIRE TEXT FACILITIES (REQUEST)

<INQUIRE TEXT FACILITIES (REQUEST) opcode : 3/6, 2/9>

<identifier : workstation identifier>

8.2.5.1.35 INQUIRE LOGICAL INPUT DEVICE STATE (REQUEST)

<INQUIRE_LOGICAL_INPUT_DEVICE_STATE_(REQUEST)_opcode : 3/7, 2/12>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_device_number>

8.2.5.1.36 INQUIRE LOGICAL INPUT DEVICE STATE (RESPONSE)

<INQUIRE_LOGICAL_INPUT_DEVICE_STATE_(RESPONSE)_opcode : 3/7, 3/12>

<enumerated : operating_mode> [see 8.2.4.2]

<enumerated : echo_switch> [see 8.2.4.2]

<integer : prompt_and_echo_type> [see 8.2.4.1]

<point : echo_area> (= 2)

case <enumerated : logical_input_device> = 0 [LOCATOR] then:

<point : initial_position>

case <enumerated : logical_input_device> = 1 [STROKE] then:

<integer : initial_no_of_points> N

<point : initial_points> (= N)

<integer : buffer_size>

<integer : buffer_editing_position>

case <enumerated : logical_input_device> = 2 [VALUATOR] then:

<real : initial_value>

<real : low_value>

<real : high_value>

case <enumerated : logical_input_device> = 3 [CHOICE] then:

<integer : initial_choice_no>

case <integer : prompt_and_echo_type> = 2 [CROSSHAIR] then:

<integer : no_of_choice_alternatives>

<enumerated : prompt_array>

8.2.5.2.20 INQUIRE TEXT FACILITIES (RESPONSE)

<INQUIRE_TEXT_FACILITIES_(RESPONSE)_opcode : 3/6, 3/9>

<integer : number_of_text_fonts_and_precision_pairs> [value N]

<<integer : text_font> [see 8.2.2.2.20]

<enumerated : text_precision>> (= N) [see 8.2.2.2.19]

<integer : number_of_character_heights>

<size_value : minimum_character_heights>

<size_value : maximum_character_heights>

<integer : number_of_character_expansion_factor>

<real : minimum_character_expansion_factor>

<real : maximum_character_expansion_factor>

<integer : number_of_predefined_text_indices>

8.2.5.2.21 INQUIRE PREDEFINED TEXT REPRESENTATION (REQUEST)

<INQUIRE_PREDEFINED_TEXT_REPRESENTATION_(REQUEST)_opcode : 3/6, 2/10>

<identifier : workstation_identifier>

<integer : predefined_text_index>

8.2.5.2.22 INQUIRE PREDEFINED TEXT REPRESENTATION (RESPONSE)

<INQUIRE_PREDEFINED_TEXT_REPRESENTATION_(RESPONSE)_opcode : 3/6, 3/10>

<integer : text_font> [see 8.2.2.2.19]

<enumerated : text_precision> [see 8.2.2.2.19]

<real : character_expansion_factor>

<real : character_spacing>

<colour_index : text_colour_index>

8.2.5.2.23 INQUIRE FILL AREA FACILITIES (REQUEST)

<INQUIRE_FILL_AREA_FACILITIES_(REQUEST)_opcode : 3/6, 2/11>

<identifier : workstation_identifier>

8.2.5.2.24 INQUIRE FILL AREA FACILITIES (RESPONSE)

<INQUIRE_FILL_AREA_FACILITIES_(RESPONSE)_opcode : 3/6, 3/11>

<integer : number_of_fill_area_interior_styles> [value N]

<enumerated : fill_area_interior_styles> (= N) [see 8.2.2.2.11]

<integer : number_of_hatch_types> [value N]

<integer : hatch_types> (= N) [see 8.2.2.2.15]

<integer : number_of_predefined_fill_area_indices>

8.2.5.2.25 INQUIRE PREDEFINED FILL AREA REPRESENTATION (REQUEST)

<INQUIRE_PREDEFINED_FILL_AREA_REPRESENTATION_(REQUEST)_opcode : 3/6, 2/12>

<identifier : workstation_identifier>

<integer : predefined_fill_area_index>

8.2.5.2.26 INQUIRE PREDEFINED FILL AREA REPRESENTATION (RESPONSE)

<INQUIRE_PREDEFINED_FILL_AREA_REPRESENTATION_(RESPONSE)_opcode : 3/6, 3/12>

<enumerated : fill_area_interior_style> [see 8.2.2.2.11]

<integer : hatch_type> [see 8.2.2.2.15]

<index : pattern_index>

<colour_index : fill_area_colour_index>

8.2.5.2.27 INQUIRE PATTERN FACILITIES (REQUEST)

<INQUIRE_PATTERN_FACILITIES_(REQUEST)_opcode : 3/6, 2/13>

<identifier : workstation_identifier>

8.2.5.2.28 INQUIRE PATTERN FACILITIES (RESPONSE)

<INQUIRE_PATTERN_FACILITIES_(RESPONSE)_opcode : 3/6, 3/13>

<integer : number_of_predefined_pattern_indices>

8.2.5.2.29 INQUIRE PREDEFINED PATTERN REPRESENTATION (REQUEST)

<INQUIRE_PREDEFINED_PATTERN_REPRESENTATION_(REQUEST)_opcode : 3/6, 2/14>

<identifier : workstation_identifier>

<integer : predefined_pattern_index>

8.2.5.2.30 INQUIRE PREDEFINED PATTERN REPRESENTATION (RESPONSE)

<INQUIRE_PREDEFINED_PATTERN_REPRESENTATION_(RESPONSE)_opcode : 3/6, 3/14>

<integer : delta X> [DX]

<integer : delta Y> [DY]

<colour_index_list : pattern_cell_colour_index_list> (DX x DY)

8.2.5.2.31 INQUIRE COLOUR FACILITIES (REQUEST)

<INQUIRE_COLOUR_FACILITIES_(REQUEST)_opcode : 3/6, 2/15>

<identifier : workstation_identifier>

8.2.5.2.32 INQUIRE COLOUR FACILITIES (RESPONSE)

<INQUIRE_COLOUR_FACILITIES_(RESPONSE)_opcode : 3/6, 3/15>

<integer : number_of_colours>

<enumerated : colour_available>

<integer : number_of_colour_indices>

with

<enumerated : colour_available> =

<enumerated : 0> [COLOUR]
|<enumerated : 1> [MONOCHROME]
|<enumerated : >1> [reserved]

8.2.5.2.33 INQUIRE PREDEFINED COLOUR REPRESENTATION (REQUEST)

<INQUIRE_PREDEFINED_COLOUR_REPRESENTATION_(REQUEST)_opcode : 3/7, 2/0>

<identifier : workstation_identifier>

<colour_index : colour_table-index>

8.2.5.2.34 INQUIRE PREDEFINED COLOUR REPRESENTATION (RESPONSE)

<INQUIRE_PREDEFINED_COLOUR_REPRESENTATION_(RESPONSE)_opcode : 3/7, 3/0>

<colour_direct : colour_data_list> (1)

8.2.5.2.35 INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (REQUEST)

<INQ._LIST_OF_AVAIL._GENERAL._DRAWING_PRIMITIVES_(REQU.)_opcode : 3/7, 2/1>

<identifier : workstation_identifier>

8.2.5.2.36 INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (RESPONSE)

<INQ._LIST_OF_AVAIL._GENERAL._DRAWING_PRIMITIVES_(RESP.)_opcode : 3/7, 3/1>

<integer : number_of_generalized_drawing_primitives> [value N]

<integer : generalized_drawing_primitive> (= N)

with

<integer : generalized_drawing_primitive> =

<integer : < 0>	[for private use]
|<integer : 0>	[RECTANGLE]
|<integer : 1>	[CIRCLE]
|<integer : 2>	[CIRCULAR ARC 3 POINT]
|<integer : 3>	[CIRCULAR ARC 3 POINT CHORD]
|<integer : 4>	[CIRCULAR ARC 3 POINT PIE]
|<integer : 5>	[CIRCULAR ARC CENTRE]
|<integer : 6>	[CIRCULAR ARC CENTRE CHORD]
|<integer : 7>	[CIRCULAR ARC CENTRE PIE]
|<integer : 8>	[ELLIPSE]
|<integer : 9>	[ELLIPTIC ARC]
|<integer : 10>	[ELLIPTIC ARC CHORD]
|<integer : 11>	[ELLIPTIC ARC PIE]
|<integer : 12>	[SPLINE]
|<integer : >12>	[reserved]

8.2.5.2.37 INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLE (REQUEST)

<INQUIRE_MAXIMUM_LENGTH_OF_WORKST._STATE_TABLE_(REQUEST)_opcode : 3/7, 2/2>

<identifier : workstation_identifier>

8.2.5.2.38 INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLE (RESPONSE)

<INQUIRE_MAXIMUM_LENGTH_OF_WORKST._STATE_TABLE_(RESPONSE)_opcode : 3/7, 3/2>

<integer : maximum_number_of_polyline_bundle_table_entries>

<integer : maximum_number_of_polymarker_bundle_table_entries>

<integer : maximum_number_of_text_bundle_table_entries>

<integer : maximum_number_of_fill_area_bundle_table_entries>

<integer : maximum_number_of_pattern_table_entries>

<integer : maximum_number_of_pattern_table_entries>

<integer : maximum_number_of_colour_table_entries>

8.2.5.2.39 INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES (REQUEST)

<INQ._NUMBER_OF_AVAILABLE_LOGICAL_INPUT_DEVICES_(REQUEST)_opcode : 3/7, 2/3>

<identifier : workstation_identifier>

8.2.5.2.40 INQUIRE NUMBER OF AVIALABLE LOGICAL INPUT DEVICES (RESPONSE)

<INQ._NUMBER_OF_AVAILABLE_LOGICAL_INPUT_DEVICES_(RESPONSE)_opcode : 3/7, 3/3>

<integer : number_of_locator_devices>

<integer : number_of_stroke_devices>

<integer : number_of_valuator_devices>

<integer : number_of_choice_devices>

<integer : number_of_pick_devices>

<integer : number_of_string_devices>

8.2.5.2.41 INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (REQUEST)

<INQUIRE_DEFAULT_LOGICAL_INPUT_DEVICE_DATA_(REQUEST)_opcode : 3/7, 2/4>

<identifier : workstation_identifier>

<enumerated : logical_input_device_class> [see 8.2.4.1]

<integer : logical_input_device_number>

8.2.5.2.42 INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (RESPONSE)

<INQUIRE_DEFAULT_LOGICAL_INPUT_DEVICE_DATA_(RESPONSE)_opcode : 3/7, 3/4>

<integer : number_of_available_prompt_and_echo_types> [value N]

<integer : sequence_of_available_prompt_and_echo_types> (= N)

<real : default_echo_area> (= 4)

case <enumerated : logical_input_device> = 0 [LOCATOR] then:

<point : initial_locator_position>

case <enumerated : logical_input_device> = 1 [STROKE] then:

<integer : maximum_input_buffer_size>

<integer : maximum_buffer_size>

<integer : first_buffer_editing_position>

case <enumerated : logical_input_device> = 2 [VALUATOR] then:

<real : default_evaluator_initial_value>

<real : default_low_value>

<real : default_high_value>

case <enumerated : logical_input_device> = 3 [CHOICE] then:

<integer : default_choice_prompt_echo_type>

<integer : default_initial_choice_number>

case <integer : default_prompt_and_echo_type> = 2 [see 6.2.4.1] then:

<integer : maximum_choice_number>

<enumerated : default_prompt_array>

with

<enumerated : default_prompt_array> =

<enumerated : 0> [ON]
|<enumerated : 1> [OFF]
|<enumerated : >1> [reserved]

case <integer : default_prompt_and_echo_type> = 3 or 4 [see 6.2.4.1] then:

<integer : maximum_number_of_choice_strings> [value N]

<string : default_sequence_of_strings> (= N)

case <integer : default_prompt_and_echo_type> = 5 [see 6.2.4.1] then:

<identifier : default_segment_name>

case <enumerated : logical_input_device> = 4 [PICK] then:

<enumerated : default_status>

<identifier : default_segment_name>

<identifier : default_pick>

with

<enumerated : default_status> =

<enumerated : 0> [PICK]
|<enumerated : 1> [NO PICK]

case <enumerated : logical_input_device> = 5 [STRING] then:

<integer : maximum_buffer_size>

<integer : first_cursor_position>

8.2.5.3 INQUIRE PRIMITIVES FOR SEGMENT STATE LIST

8.2.5.3.1 INQUIRE SET OF ASSOCIATED WORKSTATIONS (REQUEST)

<INQUIRE_SET_OF_ASSOCIATED_WORKSTATIONS_(REQUEST)_opcode : 3/7, 2/5>

<identifier : segment_name>

8.2.5.3.2 INQUIRE SET OF ASSOCIATED WORKSTATIONS (RESPONSE)

<INQUIRE_SET_OF_ASSOCIATED_WORKSTATIONS_(RESPONSE)_opcode : 3/7, 3/5>

<integer : number_of_associated_workstations> [value N]

<identifier : set_of_associated_workstation_identifiers> (= N)

8.2.5.3.3 INQUIRE SEGMENT ATTRIBUTES (REQUEST)

<INQUIRE_SEGMENT_ATTRIBUTES_(REQUEST)_opcode : 3/7, 2/13>

<identifier : segment_name>

8.2.5.3.4 INQUIRE SEGMENT ATTRIBUTES (RESPONSE)

<INQUIRE_SEGMENT_ATTRIBUTES_(RESPONSE)_opcode : 3/7, 3/13>

<matrix : segment_transformation_matrix>

<enumerated : visibility_flag > [see 8.2.3.1.8]

<enumerated : highlighting> [see 8.3.2.1.7]

<real : segment_priority> [between 0 and 1]

<enumerated : detectability> [see 8.2.3.1.11]

8.2.5.3.5 INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES (REQUEST)

<INQ._DYNAMIC_MODIFICATION_OF_SEGMENT_ATTRIB._(REQUEST)_opcode : 3/7, 2/14>

<identifier : workstation_identifier>

8.2.5.3.6 INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES (RESPONSE)

<INQ._DYNAMIC_MODIFICATION_OF_SEGMENT_ATTRIB._(RESPONSE)_opcode : 3/7, 3/14>

<enumerated : segment_transformation_changeable>

<enumerated : changeable_to_invisible>

<enumerated : changeable_to_visible>

<enumerated : highlighting_changeable>

<enumerated : segment_priority_changeable>

<enumerated : adding_primitives_to_open_segment>

<enumerated : segment deletion immediately visible>

with

<enumerated : 0> [IRG]
<enumerated : 1> [IMM]
<enumerated : >1> [reserved]

8.2.5.3.7 INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (REQUEST)

<INQUIRE_NUMBER_OF_SEGMENT_PRIORITIES_SUPPORTED_(REQUEST)_opcode : 3/7, 2/15>

<identifier : workstation_identifier>

8.2.5.3.8 INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (RESPONSE)

<INQUIRE_NUMBER_OF_SEGMENT_PRIORITIES_SUPPORTED_(RESP.)_opcode : 3/7, 3/15>

<integer : number_of_segment_priorities_supported>

8.2.5.4 INQUIRE PRIMITIVES FOR PIXELS

8.2.5.4.1 INQUIRE PIXEL (REQUEST)

<INQUIRE_PIXEL_(REQUEST)_opcode : 3/7, 2/8>

<identifier : workstation_identifier>

<point : point_p>

8.2.5.4.2 INQUIRE PIXEL (RESPONSE)

<INQUIRE_PIXEL_(RESPONSE)_opcode : 3/7, 3/8>

<colour_index : pixel_colour_index>

8.2.5.4.3 INQUIRE PIXEL ARRAY (REQUEST)

<INQUIRE_PIXEL_ARRAY_(REQUEST)_opcode : 3/7, 2/9>

<identifier : workstation_identifier>

<point : reference_point>

<integer : number_of_pixels_on_horizontal_side> [value N]

<integer : number_of_pixels_on_vertical_side> [value M]

8.2.5.4.4 INQUIRE PIXEL ARRAY (RESPONSE)

<INQUIRE_PIXEL_ARRAY_(RESPONSE)_opcode : 3/7, 3/9>

<colour_index_list : pixel_colour_index_list> (M x N)

8.2.5.4.5 INQUIRE PIXEL ARRAY DIMENSION (REQUEST)

<INQUIRE_PIXEL_ARRAY_DIMENSION_(REQUEST)_opcode : 3/7, 2/10>

<identifier : workstation_identifier>

<point : point_list_P,Q> (= 2)

8.2.5.4.6 INQUIRE PIXEL ARRAY DIMENSION (RESPONSE)

<INQUIRE_PIXEL_ARRAY_DIMENSION_(RESPONSE)_opcode : 3/7, 3/10>

<integer : number_of_pixels_on_horizontal_side>

<integer : number_of_pixels_on_vertical_side>

8.2.6 PROTOCOL DESCRIPTOR PRIMITIVES

8.2.6.1 SET DOMAIN RING

<SET_DOMAIN_RING_opcode : 3/2, 2/4>

<enumerated : angular_resolution_factor>

<size_value : ring_size>

with

<enumerated : angular_resolution_factor> =

<enumerated : 0> [Resolution_1]
|<enumerated : 1> [Resolution_2]
|<enumerated : 2> [Resolution_3]
|<enumerated : 3> [Resolution_4]
|<enumerated : >1> [reserved]

8.2.6.2 SET COLOUR HEADER

<SET_COLOUR_HEADER_opcode : 3/2, 2/8>

<integer : unit_resolution>

<enumerated : coding_method>

with

<enumerated : coding_method> =

<enumerated : 1> [RGB_coding]
|<enumerated : >1> [reserved]

8.2.6.3 SET COORDINATE PRECISION

<SET_COORDINATE_PRECISION_opcode : 3/2, 2/9>

<integer : magnitude_code>

<integer : granularity_code>

<integer : default_exponent>

8.2.6.4 SET REAL PRECISION

<SET_REAL_PRECISION_opcode : 3/2, 3/9>

<integer : magnitude_code>

<integer : granularity_code>

<integer : default_exponent>

8.2.6.5 SET COLOUR INDEX PRECISION

<SET_COLOUR_INDEX_PRECISION_opcode : 3/2, 2/10>

<integer : index_number_of_bits>

9. DEFAULTS

The defaults in this clause define the values of parameters to be taken if a particular parameter value is out of range or not yet defined. The purpose of these default values is to be able to process primitives even if some parameters values are not yet defined.

Parameter	Value
ASPECT SOURCE FLAGS	INDIVIDUAL (all)
CHARACTER EXPANSION FACTOR	1.0
CHARACTER SPACING	0.0
CHARACTER VECTORS	Implementation dependent
CLIPPING RECTANGLE	(0.0, 0.0) and (1.0, 1.0)
DEFERRAL MODE	ASAP (= as soon as possible)
DOMAIN RING	
Angular resolution factor	RESOLUTION_0
Ring size	2**(-8-current granularity code) [basic radius]
FILL AREA BUNDLE TABLE	The entry of the FILL AREA BUNDLE TABLE corresponding to FILL AREA INDEX 1 contains the defaults values of FILL AREA INTERIOR STYLE, FILL AREA STYLE INDEX and FILL AREA COLOUR INDEX. Non defined entries use the attributes as specified by FILL AREA INDEX 1
FILL AREA COLOUR INDEX	1
FILL AREA INDEX	1
FILL AREA INTERIOR STYLE	SOLID
FILL AREA STYLE INDEX	1
HIGHLIGHTING FLAG	NOTHIGHLIGHTED
IMPLICIT REGENERATION FLAG	ALLOWED
LINE TYPE	SOLID
LINE WIDTH SCALE FACTOR	1.0
MARKER SIZE SCALE FACTOR	1.0
MARKER TYPE	Asterisk
PATTERN INDEX	1
PATTERN REFERENCE POINT	(0.0, 0.0)
PATTERN TABLE	
Delta x	1
Delta y	1
Pattern cell colour index list	1
PATTERN VECTORS	(0.0,1.0) and (1.0, 0.0)
POLYLINE BUNDLE TABLE	The entry of the POLYLINE BUNDLE TABLE corresponding to POLYLINE 1 contains the default values of LINE TYPE, LINE WIDTH and POLYLINE COLOUR INDEX. Non predefined entries use the attributes as specified by POLYLINE INDEX 1.

POLYLINE COLOUR INDEX	1
POLYLINE INDEX	1
POLYMARKER BUNDLE TABLE	The entry of the POLYMARKER BUNDLE TABLE corresponding to INDEX 1 contains the default values of MARKER TYPE, MARKER SIZE and POLYMARKER COLOUR INDEX. Non predefined entries use the attributes as specified by POLYMARKER INDEX 1.
POLYMARKER COLOUR INDEX	1
POLYMARKER INDEX	1
SEGMENT PRIORITY	0.0
SET COORDINATE PRECISION	
Exponent	- 9
Granularity code	- 9
Magnitude code	4
Explicit exponent ALLOWED	FORBIDDEN
SET REAL PRECISION	
Exponent	- 9
Granularity code	- 9
Magnitude code	4
Explicit exponent ALLOWED	FORBIDDEN
SET COLOUR INDEX PRECISION	
Number of bits	5
STARTING ENTRY IN THE COLOUR TABLE	1
TEXT ALIGNMENT	NORMAL, NORMAL
TEXT BUNDLE TABLE	The entry of the TEXT BUNDLE TABLE corresponding to TEXT INDEX 1 contains the default values of TEXT FONT AND PRECISION, CHARACTER EXPANSION FACTOR, CHARACTER SPACING and TEXT COLOUR INDEX. Non predefined entries use the attributes as specified by TEXT INDEX 1.
TEXT COLOUR INDEX	1
TEXT FONT AND PRECISION	0, STRING
TEXT INDEX	1
TEXT PATH	RIGHT
TRANSFORMATION MATRIX	$\begin{bmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \end{bmatrix}$
UNIT RESOLUTION	4
UPDATE REGENERATION FLAG	POSTPONE
VISIBILITY FLAG	VISIBLE
WORKSTATION VIEWPORT	The largest square fits into the display area (See figure 9)
WORKSTATION WINDOW	(0.0, 0.0) and (1.0, 1.0)

10. CONFORMANCE

Conformance to a standard means that all of its requirements are met.

GDS addresses those classes of devices which are capable of processing the encoded graphics primitives defined in it. Such devices can vary from each other depending on the application for which they have been specifically designed. To take this into account, GDS provides a set of primitives grouped in levels in a manner consistent with the GKS levels.

Therefore conformance to GDS means conformance to one of the levels defined in this document (see section 5). For a device, to be in conformance to one level means to offer at least all the no optional functionalities pertaining to this level.

APPENDIX A

PRIMITIVES, WORKSTATION CATEGORIES, LEVELS AND OPTIONS

This appendix forms an integral part of the standard

This attached table lists the primitives contained in the document

For each primitives is indicated :

- The applicability to workstation categories
- The level in which it appears.
- The state (mandatory or non-mandatory)

The following notational conventions are used :

- SS : workstation independent segment storage
- O : workstation of category OUTPUT
- OI : workstation of category OUTIN
- I : workstation of category INPUT
- SN : SS is fundamental to the primitive, but the workstation identifier parameter cannot be the one of the SS.
- M : Mandatory
- N : Non-mandatory

PRIMITIVE	LEVEL	APPLIES TO				OPTION
OPEN WORKSTATION	L0a	SS	O	OI	I	M
CLOSE WORKSTATION	L0a	SS	O	OI	I	M
ACTIVATE WORKSTATION	L0a	SS	O	OI	I	M
DEACTIVATE WORKSTATION	L0a	SS	O	OI	I	M
CLEAR WORKSTATION	L0a	SS	O	OI	I	M
REDRAW ALL SEGMENTS ON WORKSTATION	Lla		O	OI		M
UPDATE WORKSTATION	L0a		O	OI		M
SET DEFERRAL STATE	Lla		O	OI		M
MESSAGE	Lla		O	OI		N
ESCAPE	L0a	SS	O	OI	I	N
SET DEFAULTS	L0a	SS	O	OI	I	M
INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (REQUEST/RESPONSE)	L0a	not applicable				N
POLYLINE	L0a	SS	O	OI		M
POLYMARKER	L0a	SS	O	OI		M
TEXT	L0a	SS	O	OI		M
FILL AREA	L0a	SS	O	OI		M
CELL ARRAY	L0a	SS	O	OI		M
GENERALIZED DRAWING PRIMITIVE :	L0a	SS	O	OI		M
ARC						M
ARC CHORD						M
ARC PIE						M
CIRCLE						M
RECTANGLE						M
ELLIPSE						N
ELLIPTIC ARC						N
ELLIPTIC ARC CHORD						N
ELLIPTIC ARC PIE						N
SPLINE						N

PRIMITIVE	LEVEL	APPLIES TO				OPTION
SET POLYLINE INDEX	L0a	SS	O	OI		M
SET POLYLINE REPRESENTATION	Lla		O	OI		M
SET POLYLINE ASPECT SOURCE FLAGS	L0a	SS	O	OI		M
SET LINE TYPE	L0a	SS	O	OI		M
SET LINE WIDTH SCALE FACTOR	L0a	SS	O	OI		M
SET POLYLINE COLOUR INDEX	L0a	SS	O	OI		M
SET POLYMARKER INDEX	L0a	SS	O	OI		M
SET POLYMARKER REPRESENTATION	Lla		O	OI		M
SET POLYMARKER ASPECT SOURCE FLAGS	L0a	SS	O	OI		M
SET MARKER TYPE	L0a	SS	O	OI		M
SET MARKER SIZE SCALE FACTOR	L0a	SS	O	OI		M
SET POLYMARKER COLOUR INDEX	L0a	SS	O	OI		M
SET FILL AREA INDEX	L0a	SS	O	OI		M
SET FILL AREA REPRESENTATION	Lla		O	OI		M
SET FILL AREA ASPECT SOURCE FLAGS	L0a	SS	O	OI		M
SET FILL AREA INTERIOR STYLE	L0a	SS	O	OI		M
SET FILL AREA STYLE INDEX	L0a	SS	O	OI		M
SET FILL AREA COLOUR INDEX	L0a	SS	O	OI		M
SET PATTERN REPRESENTATION	Lla		O	OI		N
SET PATTERN REFERENCE POINT	L0a	SS	O	OI		N
SET PATTERN SIZE VECTORS	L0a	SS	O	OI		N
SET TEXT INDEX	L0a	SS	O	OI		M
SET TEXT REPRESENTATION	Lla		O	OI		M
SET TEXT ASPECT SOURCE FLAGS	L0a	SS	O	OI		M
SET TEXT FONT AND PRECISION	L0a	SS	O	OI		M
SET CHARACTER EXPANSION FACTOR	L0a	SS	O	OI		M
SET CHARACTER SPACING	L0a	SS	O	OI		M
SET TEXT COLOUR INDEX	L0a	SS	O	OI		M
SET TEXT PATH	L0a	SS	O	OI		M

PRIMITIVE	LEVEL	APPLIES TO				OPTION
SET TEXT ALIGNMENT	L0a	SS	O	OI		M
SET CHARACTER VECTORS	L0a	SS	O	OI		M
SET COLOUR HEADER	L0a	SS	O	OI		N
SET COLOUR REPRESENTATION	L0a		O	OI		M
SET PICK IDENTIFIER	Lla	SS	O	OI		M
SET WORKSTATION WINDOW	L0a		O	OI	I	M
SET WORKSTATION VIEWPORT	L0a		O	OI	I	M
SET CLIPPING RECTANGLE	L0a	SS	O	OI		M
CREATE SEGMENT	Lla	SS	O	OI		M
CLOSE SEGMENT	Lla	SS	O	OI		M
RENAME SEGMENT	Lla	SS	O	OI		M
DELETE SEGMENT	Lla	SS	O	OI		M
DELETE SEGMENT FROM WORKSTATION	Lla	SS	O	OI		M
ASSOCIATE SEGMENT WITH WORKSTATION	L2a	SS	O	OI		M
COPY SEGMENT TO WORKSTATION	L2a	SN	O	OI		M
INSERT SEGMENT	L2a	SS	O	OI		M
SET SEGMENT TRANSFORMATION	Lla	SS	O	OI		M
SET VISIBILITY	Lla	SS	O	OI		M
SET HIGHLIGHTING	Lla	SS	O	OI		M
SET SEGMENT PRIORITY	Lla	SS	O	OI		N
SET DETECTABILITY	Lla	SS	O	OI		M
SET DOMAIN RING	L0a	SS	O	OI	I	M
SET COORDINATE PRECISION	L0a	SS	O	OI	I	M
SET REAL PRECISION	L0a	SS	O	OI	I	M
SET COLOUR INDEX PRECISION	L0a	SS	O	OI	I	N
EMERGENCY CLOSE	L0a	SS	O	OI	I	M
ERROR DETECTED (RESPONSE)	L0a	SS	O	OI	I	N
INITIALIZE LOGICAL INPUT DEVICE	L0b			OI	I	M

PRIMITIVE	LEVEL	APPLIES TO				OPTION
SET LOGICAL INPUT DEVICE MODE	L0b			OI	I	M
REQUEST LOGICAL INPUT DEVICE (REQUEST/RESPONSE)	L0b			OI	I	M
SAMPLE LOGICAL INPUT DEVICE (REQUEST/RESPONSE)	L0c			OI	I	M
AWAIT EVENT (REQUEST/RESPONSE)	L0c			OI	I	M
GET LOGICAL INPUT DEVICE (REQUEST/RESPONSE)	L0c			OI	I	M
FLUSH DEVICE EVENTS	L0c			OI	I	M
INQUIRE PIXEL ARRAY DIMENSIONS (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PIXEL ARRAY (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PIXEL (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE WORKSTATION CONNECTION AND TYPE (REQUEST/RESPONSE)	L0a	SS	O	OI	I	N
INQUIRE WORKSTATION STATE (RESQUEST/RESPONSE)	L0a	SS	O	OI		N
INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE LIST OF POLYLINE INDICES (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE POLYLINE REPRESENTATION (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE LIST OF POLYMARKER INDICES (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE POLYMARKER REPRESENTATION (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE LIST OF TEXT INDICES (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE TEXT REPRESENTATION (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE TEXT EXTENT (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE LIST OF FILL AREA INDICES (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE FILL AREA REPRESENTATION (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE LIST OF PATTERN INDICES (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE PATTERN REPRESENTATION (REQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE LIST OF COLOUR INDICES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE COLOUR REPRESENTATION (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE WORKSTATION TRANSFORMATION (REQUEST/RESPONSE)	L0a		O	OI	I	N
INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (REQUEST/RESPONSE)	Lla	SS	O	OI		N

PRIMITIVE	LEVEL	APPLIES TO				OPTION
INQUIRE LOGICAL INPUT DEVICE STATE (REQUEST/RESPONSE)	L0b			OI	I	N
INQUIRE WORKSTATION CATEGORY (REQUEST/RESPONSE)	L0a	SS	O	OI	I	N
INQUIRE WORKSTATION CLASSIFICATION (REQUEST/RESPONSE)	L0a	SS	O	OI		N
INQUIRE MAXIMUM DISPLAY SURFACE SIZE (REQUEST/RESPONSE)	L0a		O	OI	I	N
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE DEFAULT DEFERRAL STATE VALUES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE POLYLINE FACILITIES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PREDEFINED POLYLINE REPRESENTATION (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE POLYMARKER FACILITIES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PREDEFINED POLYMARKER REPRESENTATION (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE TEXT FACILITIES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PREDEFINED TEXT REPRESENTATION (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE FILL AREA FACILITIES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PREDEFINED FILL AERA REPRESENTATION (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PATTERN FACILITIES (RESQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PREDEFINED PATTERN REPRESENTATION (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE COLOUR FACILITIES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE PREDEFINED COLOUR REPRESENTATION (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE GENERALIZED DRAWING PRIMITIVE (REQUEST/RESPONSE)	L0a		O	OI		N
INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (REQUEST/RESPONSE)	L1a		O	OI		N
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (REQUEST/RESPONSE)	L1a		O	OI		N

PRIMITIVE	LEVEL	APPLIES TO				OPTION
INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES (RESQUEST/RESPONSE)	Lla		O	OI		N
INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES (REQUEST/RESPONSE)	L0b		O	OI	I	N
INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (REQUEST/RESPONSE)	L0b			OI	I	N

APPENDIX B

B.1 SHORT NOTES ON B-SPLINE CURVES

A spline is a piecewise polynomial function passing through a set of points called knots (see figure 41).

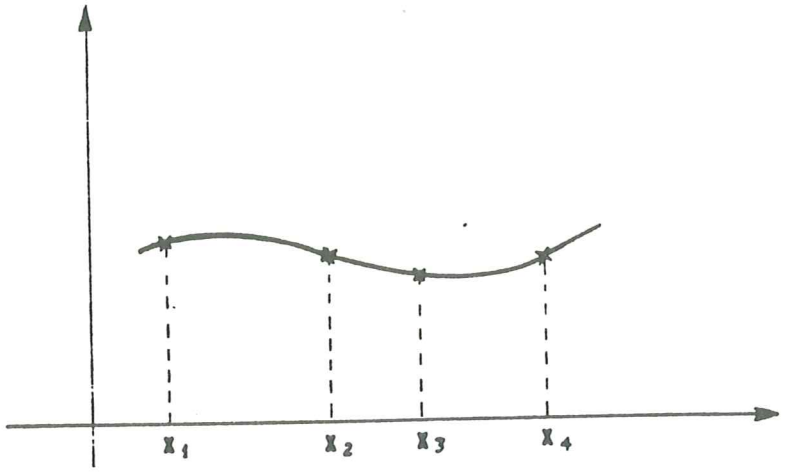


Figure 41

A spline passing through 4 knots

The values X_1, X_2, ..., X_k are called breakpoints. Between each breakpoint, f(x) is a degree-m polynomial and the j-first derivatives are continuous at each breakpoint. In most applications, polynomials of degree 2 or 3 with j=1 or 2 respectively are sufficient.

Each (X_i, X_{i+1}) defines a sub-interval. B-splines are splines that are zero at all sub-intervals except m+1 of them, where m is the degree of the polynomials. In most cases uniform B-splines are used, that are B-splines for which the breakpoints are equally spaced. In the two-dimensional space, a B-spline curve is defined as :

$$P(t) = \sum_{i=1}^{n} P_i \, N_{im}(t)$$

Where P(t) is a point on the curve, points Pi are called guiding points and Nim is a m-degree B-spline.

For a uniform quadratic B-spline, between two knots, we have :

$$P(t) = \frac{(P_{i+2} + P_i - P_{i+1})}{2}t^2 + (P_{i+1} - P_i)t + \frac{1}{2}(P_{i+1} + P_i)$$

The corresponding knots are :

$$\frac{P_i + P_{i+1}}{2} \quad \text{and} \quad \frac{P_{i+1} + P_{i+2}}{2}$$

An example of such a curve is given in figure 42. Knots are located on the middle of the segments joining the breakpoints and the curve is tangent to the segment at this point.

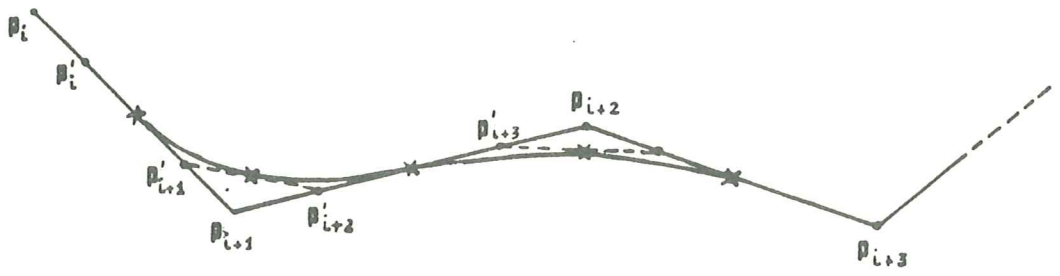


Figure 42

Such spline curves can easilly be generated using the sub-division or refinement properties of B-splines note 7.

We can apply this theory to quadratic B-splines by replacing Pi, Pi+1 and Pi+2 by the set of four points : P'_i, P'_{i+1}, P'_{i+2} and P'_{i+3} given by :

$$P'_i = \frac{1}{2}\left(P_i + \frac{P_i + P_{i+1}}{2}\right)$$

$$P'_{i+1} = \frac{1}{2}\left(\frac{P_{i+1} + P_i}{2} + P_{i+1}\right)$$

$$P'_{i+2} = \frac{1}{2}\left(P_{i+1} + \frac{P_{i+1} + P_{i+2}}{2}\right)$$

$$P'_{i+3} = \frac{1}{2}\left(\frac{P_{i+1} + P_{i+2}}{2} + P_{i+2}\right)$$

The new guiding points will produce the same curve as the former ones but they introduce a supplementary knot :

$$\frac{P'_{i+1} + P'_{i+2}}{2}$$

Thus the original curve segment has been divided into two parts. Furthermore the new guiding points are closer to the curve than the former ones (see figure 42).

By simply repeating this procedure, until the curve segments reach the size of a pixel, the spline curve can be drawn. Only very simple integer arithmetic is needed at each sub-division step (addition and shift). An algorithm of this type is given in note 8. Note that in this algorithm, the given end-points of the curve are no guiding points but knots.

The coordinates as specified in the GDP SPLINE-primitive will be considered as guiding points of a uniform quadratic B-spline curve. The curve can thus easily be generated using the above mentioned sub-division technique.

Note 7 : *LANE J.M., R.F. RIESENFELD. "A theorical Development for the Computer Generation and Display of Piecewise Polynomial Surfaces". I.E.E.E. Trans. on P.A.M.I. - Vol. PAMI - 2, No 1 (Janv 80) pp 35-46.*

Note 8 : *CHAIKIN G.M. "An Algorithm for High-Speed Curve Generation" Computer Graphics and Image Processing 1974 - Vol. 3 - pp 346-349.*

B.2 ELLIPSE PRIMITIVES

A central question is how to represent a generally oriented ellipse such that the necessary properties of the picture are preserved across all graphical transformations. Unfortunately, the major and minor axes cannot be used for an ellipse in a general orientation since, as shown in figure 43 below, these axes do not remain mutually orthogonal (do not remain axes) accross a scaling transformation which does not preserve aspect ratio.

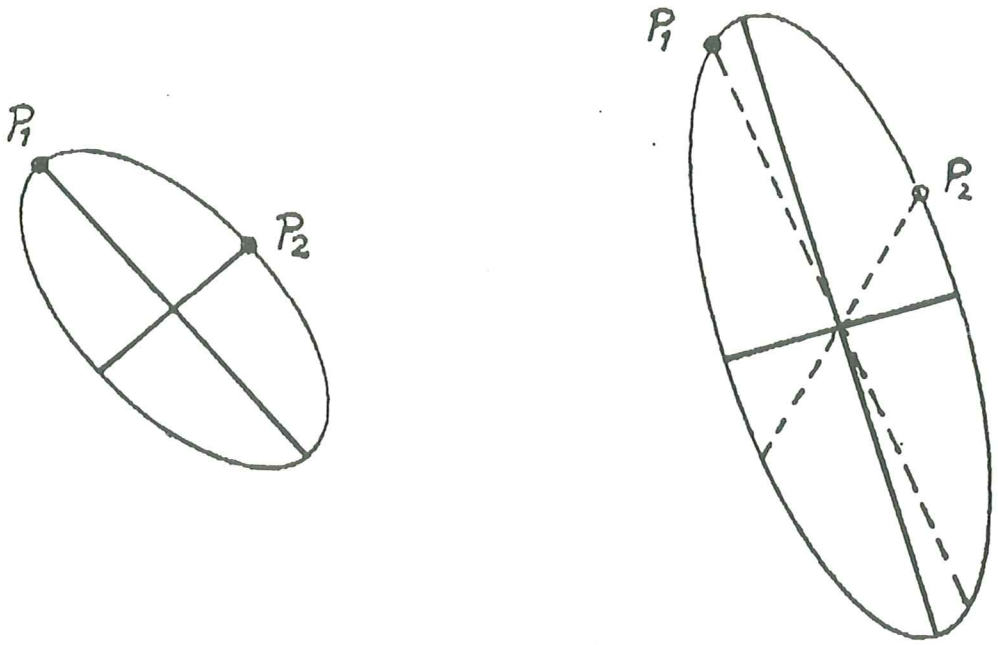


Figure 43 : The scaling of an ellipse and its axes such that X'=X and Y' = 2Y

The problem can be solved by utilizing the fact that any Conjugate Diameter Pair (CDP) of the ellipse remains a CDP across any graphical transformation.

A CDP is a pair D,d of diameters of the ellipse such that a tangent to the ellipse at each endpoint of a diameter is parallel to the other diameter. Note that the four tangents to the ellipse at the endpoints of the CDP form a parallelogram whose sides are bisected by the endpoints.

This is demonstrated below in figure 44 in which the ellipse has been scaled by a factor of two in the Y-direction only.

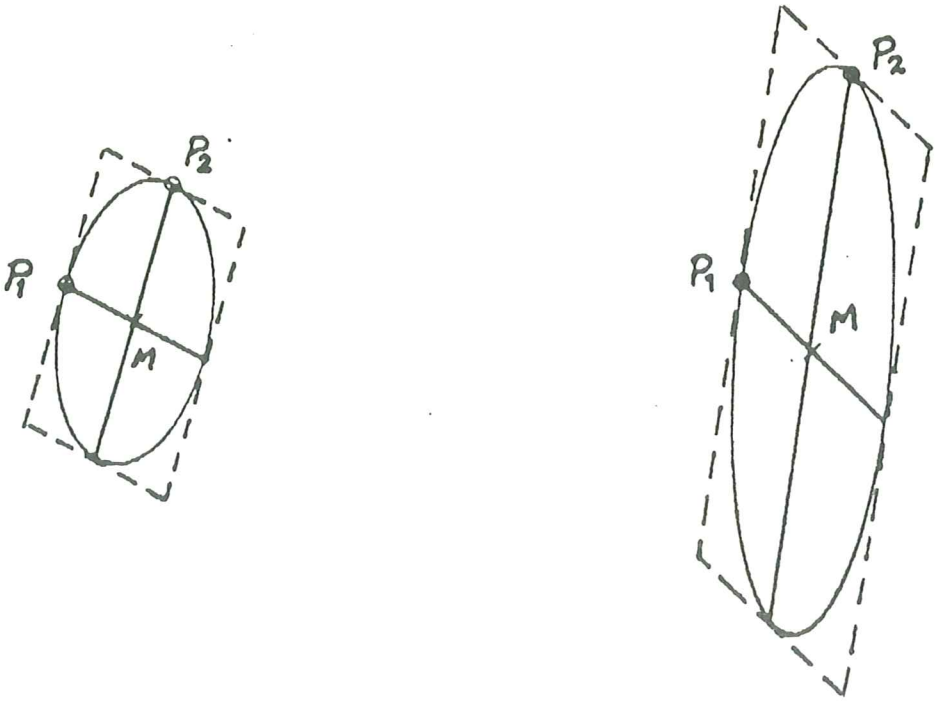


Figure 44 : Ellipses defined by a Conjugate Diameter Pair

Thus any CDP can be used to represent an ellipse. Note that the (mutually perpendicular) major and minor axes of an ellipse, and any pair of perpendicular diameters of a circle are CDPs, although they do not remain perpendicular across a transformation.

To thus represent an ellipse, we need only three points :

- the centre point M (Xm, Ym) of the ellipse ;

- two endpoints P1 (X1, Y1) and P2 (X2, Y2) of a CDP

The CDP vector components, relative to the centre point, are defined as follows :

```
DX1 = X1 - Xm
DY1 = Y1 - Ym
DX2 = X2 - Xm
DY2 = Y2 - Ym
```

The CDP vector components aree the coefficients of the parametric equations :

```
X = Xm + DX1 * cos(t) + DX2 * sin(t)
Y = Ym + DY1 * cos(t) + DY2 * sin(t)
```

in which t runs from 0 to 2 * Pi

Reference :

James C. Almond - A suggestion for ellipse primitives.

APPENDIX C 'Cross References'

Primitives	CLAUSE OF DESCRIPTION	CLAUSE OF ENCODING
WORKSTATION MANAGMENT		
OPEN WORKSTATION	6.2.1.1	8.2.1.1
CLOSE WORKSTATION	6.2.1.2	8.2.1.2
ACTIVATE WORKSTATION	6.2.1.3	8.2.1.3
DEACTIVATE WORKSTATION	6.2.1.4	8.2.1.4
CLEAR WORKSTATION	6.2.1.5	8.2.1.5
SET DEFAULTS	6.2.1.6	8.2.1.6
GDS ESCAPE	6.2.1.7	8.2.1.7
MESSAGE	6.2.1.8	8.2.1.8
INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (REQUEST)	6.2.1.9	8.2.1.9
INQUIRE GRAPHICS CONFIGURATION IDENTIFICATION (RESPONSE)	6.2.1.10	8.2.1.10
OUTPUT DRAWING		
POLYLINE	6.2.2.1.1	8.2.2.1.1
POLYMARKER	6.2.2.1.2	8.2.2.1.2
FILL AREA	6.2.2.1.3	8.2.2.1.3
TEXT	6.2.2.1.4	8.2.2.1.4
CELL ARRAY	6.2.2.1.5	8.2.2.1.5
GENERALIZED DRAWING PRIMITIVE	6.2.2.1.6	8.2.2.1.6
GDP RECTANGLE	6.2.2.1.6.1	8.2.2.1.6
GDP CIRCLE	6.2.2.1.6.2	8.2.2.1.6
GDP CIRCULAR ARC 3 POINT	6.2.2.1.6.3	8.2.2.1.6
GDP CIRCULAR ARC 3 POINT CHORD	6.2.2.1.6.4	8.2.2.1.6
GDP CIRCULAR ARC 3 POINT PIE	6.2.2.1.6.5	8.2.2.1.6
GDP CIRCULAR ARC CENTRE	6.2.2.1.6.6	8.2.2.1.6
GDP CIRCULAR ARC CENTRE CHORD	6.2.2.1.6.7	8.2.2.1.6
GDP CIRCULAR ARC CENTRE PIE	6.2.2.1.6.8	8.2.2.1.6
GDP ELLIPSE	6.2.2.1.6.9	8.2.2.1.6
GDP ELLIPTIC ARC	6.2.2.1.6.10	8.2.2.1.6
GDP ELLIPTIC ARC CHORD	6.2.2.1.6.11	8.2.2.1.6
GDP ELLIPTIC ARC PIE	6.2.2.1.6.12	8.2.2.1.6
GDP SPLINE	6.2.2.1.6.13	8.2.2.1.6

Appendix C 'Cross References'

Primitives	CLAUSE OF DESCRIPTION	CLAUSE OF ENCODING
OUTPUT ATTRIBUTES		
SET POLYLINE REPRESENTATION	6.2.2.2.1	8.2.2.2.1
SET POLYLINE INDEX	6.2.2.2.2	8.2.2.2.2
SET LINE TYPE	6.2.2.2.3	8.2.2.2.3
SET LINE WIDTH SCALE FACTOR	6.2.2.2.4	8.2.2.2.4
SET POLYLINE COLOUR INDEX	6.2.2.2.5	8.2.2.2.5
SET POLYMARKER REPRESENTATION	6.2.2.2.6	8.2.2.2.6
SET POLYMARKER INDEX	6.2.2.2.7	8.2.2.2.7
SET MARKER TYPE	6.2.2.2.8	8.2.2.2.8
SET MARKER SIZE SCALE FACTOR	6.2.2.2.9	8.2.2.2.9
SET POLYMARKER COLOUR INDEX	6.2.2.2.10	8.2.2.2.10
SET FILL AREA REPRESENTATION	6.2.2.2.11	8.2.2.2.11
SET FILL AREA INDEX	6.2.2.2.12	8.2.2.2.12
SET FILL AREA INTERIOR STYLE	6.2.2.2.13	8.2.2.2.13
SET FILL AREA COLOUR INDEX	6.2.2.2.14	8.2.2.2.14
SET FILL AREA STYLE INDEX	6.2.2.2.15	8.2.2.2.15
SET PATTERN REPRESENTATION	6.2.2.2.16	8.2.2.2.16
SET PATTERN REFERENCE POINT	6.2.2.2.17	8.2.2.2.17
SET PATTERN VECTORS	6.2.2.2.18	8.2.2.2.18
SET TEXT REPRESENTATION	6.2.2.2.19	8.2.2.2.19
SET TEXT INDEX	6.2.2.2.20	8.2.2.2.20
SET TEXT FONT AND PRECISION	6.2.2.2.21	8.2.2.2.21
SET CHARACTER EXPANSION FACTOR	6.2.2.2.22	8.2.2.2.22
SET CHARACTER SPACING	6.2.2.2.23	8.2.2.2.23
SET TEXT COLOUR INDEX	6.2.2.2.24	8.2.2.2.24
SET TEXT PATH	6.2.2.2.25	8.2.2.2.25
SET CHARACTER VECTORS	6.2.2.2.26	8.2.2.2.26
SET TEXT ALIGNMENT	6.2.2.2.27	8.2.2.2.27
SET COLOUR REPRESENTATION	6.2.2.2.28	8.2.2.2.28
SET ASPECT SOURCE FLAGS	6.2.2.2.29	8.2.2.2.29
SET PICK IDENTIFIER	6.2.2.2.30	8.2.2.2.30
TRANSFORMATION		
SET WORKSTATION WINDOW	6.2.2.3.1	8.2.2.3.1
SET WORKSTATION VIEWPORT	6.2.2.3.2	8.2.2.3.2
CLIPPING		
SET CLIPPING RECTANGLE	6.2.2.4.1	8.2.2.4.1

Appendix C 'Cross References'

Primitives	CLAUSE OF DESCRIPTION	CLAUSE OF ENCODING
CONTROL		
UPDATE WORKSTATION	6.2.2.5.1	8.2.2.5.1
SET DEFERRAL STATE	6.2.2.5.2	8.2.2.5.2
EMERGENCY CLOSE	6.2.2.5.3	8.2.2.5.3
ERROR DETECTED (RESPONSE)	6.2.2.5.4	8.2.2.5.4
SEGMENT		
CREATE SEGMENT	6.2.3.1.1	8.2.3.1.1
CLOSE SEGMENT	6.2.3.1.2	8.2.3.1.2
RENAME SEGMENT	6.2.3.1.3	8.2.3.1.3
DELETE SEGMENT FROM WORKSTATION	6.2.3.1.4	8.2.3.1.4
DELETE SEGMENT	6.2.3.1.5	8.2.3.1.5
REDRAW ALL SEGMENTS ON WORKSTATION	6.2.3.1.6	8.2.3.1.6
SET HIGHLIGHTING	6.2.3.1.7	8.2.3.1.7
SET VISIBILITY	6.2.3.1.8	8.2.3.1.8
SET SEGMENT TRANSFORMATION	6.2.3.1.9	8.2.3.1.9
SET SEGMENT PRIORITY	6.2.3.1.10	8.2.3.1.10
SET DETECTABILITY	6.2.3.1.11	8.2.3.1.11
ASSOCIATE SEGMENT WITH WORKSTATION	6.2.3.2.1	8.2.3.2.1
COPY SEGMENT TO WORKSTATION	6.2.3.2.2	8.2.3.2.2
INSERT SEGMENT	6.2.3.2.3	8.2.3.2.3
INPUT		
INITIALIZE LOGICAL INPUT DEVICE	6.2.4.1	8.2.4.1
SET LOGICAL INPUT DEVICE MODE	6.2.4.2	8.2.4.2
REQUEST LOGICAL INPUT DEVICE (REQUEST)	6.2.4.3	8.2.4.3
REQUEST LOGICAL INPUT DEVICE (RESPONSE)	6.2.4.4	8.2.4.4
SAMPLE LOGICAL INPUT DEVICE (REQUEST)	6.2.4.5	8.2.4.5
SAMPLE LOGICAL INPUT DEVICE (RESPONSE)	6.2.4.6	8.2.4.6
AWAIT EVENT (REQUEST)	6.2.4.7	8.2.4.7
AWAIT EVENT (RESPONSE)	6.2.4.8	8.2.4.8
GET LOGICAL INPUT DEVICE (REQUEST)	6.2.4.9	8.2.4.9
GET LOGICAL INPUT DEVICE (RESPONSE)	6.2.4.10	8.2.4.10
FLUSH DEVICE EVENTS	6.2.4.11	8.2.4.11

Appendix C 'Cross References'

Primitives	CLAUSE OF DESCRIPTION	CLAUSE OF ENCODING
INQUIRES OF WORKSTATION STATE LIST		
INQUIRE WORKSTATION STATE (REQUEST)	6.2.5.1.1	8.2.5.1.1
INQUIRE WORKSTATION STATE (RESPONSE)	6.2.5.1.2	8.2.5.1.2
INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (REQUEST)	6.2.5.1.3	8.2.5.1.3
INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES (RESPONSE)	6.2.5.1.4	8.2.5.1.4
INQUIRE POLYLINE INDICES (REQUEST)	6.2.5.1.5	8.2.5.1.5
INQUIRE POLYLINE INDICES (RESPONSE)	6.2.5.1.6	8.2.5.1.6
INQUIRE POLYLINE REPRESENTATION (REQUEST)	6.2.5.1.7	8.2.5.1.7
INQUIRE POLYLINE REPRESENTATION (RESPONSE)	6.2.5.1.8	8.2.5.1.8
INQUIRE POLYMARKER INDICES (REQUEST)	6.2.5.1.9	8.2.5.1.9
INQUIRE POLYMARKER INDICES (RESPONSE)	6.2.5.1.10	8.2.5.1.10
INQUIRE POLYMARKER REPRESENTATION (REQUEST)	6.2.5.1.11	8.2.5.1.11
INQUIRE POLYMARKER REPRESENTATION (RESPONSE)	6.2.5.1.12	8.2.5.1.12
INQUIRE TEXT INDICES (REQUEST)	6.2.5.1.13	8.2.5.1.13
INQUIRE TEXT INDICES (RESPONSE)	6.2.5.1.14	8.2.5.1.14
INQUIRE TEXT REPRESENTATION (REQUEST)	6.2.5.1.15	8.2.5.1.15
INQUIRE TEXT REPRESENTATION (RESPONSE)	6.2.5.1.16	8.2.5.1.16
INQUIRE TEXT EXTENT (REQUEST)	6.2.5.1.17	8.2.5.1.17
INQUIRE TEXT EXTENT (RESPONSE)	6.2.5.1.18	8.2.5.1.18
INQUIRE FILL AREA INDICES (REQUEST)	6.2.5.1.19	8.2.5.1.19
INQUIRE FILL AREA INDICES (RESPONSE)	6.2.5.1.20	8.2.5.1.20
INQUIRE FILL AREA REPRESENTATION (REQUEST)	6.2.5.1.21	8.2.5.1.21
INQUIRE FILL AREA REPRESENTATION (RESPONSE)	6.2.5.1.22	8.2.5.1.22
INQUIRE PATTERN INDICES (REQUEST)	6.2.5.1.23	8.2.5.1.23
INQUIRE PATTERN INDICES (RESPONSE)	6.2.5.1.24	8.2.5.1.24
INQUIRE PATTERN REPRESENTATION (REQUEST)	6.2.5.1.25	8.2.5.1.25
INQUIRE PATTERN REPRESENTATION (RESPONSE)	6.2.5.1.26	8.2.5.1.26
INQUIRE COLOUR INDICES (REQUEST)	6.2.5.1.27	8.2.5.1.27
INQUIRE COLOUR INDICES (RESPONSE)	6.2.5.1.28	8.2.5.1.28
INQUIRE COLOUR REPRESENTATION (REQUEST)	6.2.5.1.29	8.2.5.1.29
INQUIRE COLOUR REPRESENTATION (REQUEST)	6.2.5.1.30	8.2.5.1.30
INQUIRE WORKSTATION TRANSFORMATION (REQUEST)	6.2.5.1.31	8.2.5.1.31
INQUIRE WORKSTATION TRANSFORMATION (RESPONSE)	6.2.5.1.32	8.2.5.1.32
INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (REQUEST)	6.2.5.1.33	8.2.5.1.33
INQUIRE SET OF SEGMENT NAMES ON WORKSTATION (RESPONSE)	6.2.5.1.34	8.2.5.1.34
INQUIRE LOGICAL INPUT DEVICE STATE (REQUEST)	6.2.5.1.35	8.2.5.1.35
INQUIRE LOGICAL INPUT DEVICE STATE (RESPONSE)	6.2.5.1.36	8.2.5.1.36

Appendix C 'Cross References'

Primitives	CLAUSE OF DESCRIPTION	CLAUSE OF ENCODING
WORKSTATION DESCRIPTION TABLE INQUIRES		
INQUIRE WORKSTATION CATEGORY (REQUEST)	6.2.5.2.1	8.2.5.2.1
INQUIRE WORKSTATION CATEGORY (RESPONSE)	6.2.5.2.2	8.2.5.2.2
INQUIRE WORKSTATION CLASSIFICATION (REQUEST)	6.2.5.2.3	8.2.5.2.3
INQUIRE WORKSTATION CLASSIFICATION (RESPONSE)	6.2.5.2.4	8.2.5.2.4
INQUIRE MAXIMUM DISPLAY SURFACE SIZE (REQUEST)	6.2.5.2.5	8.2.5.2.5
INQUIRE MAXIMUM DISPLAY SURFACE SIZE (RESPONSE)	6.2.5.2.6	8.2.5.2.6
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (REQUEST)	6.2.5.2.7	8.2.5.2.7
INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES (RESPONSE)	6.2.5.2.8	8.2.5.2.8
INQUIRE DEFAULT DEFERRAL STATE VALUES (REQUEST)	6.2.5.2.9	8.2.5.2.9
INQUIRE DEFAULT DEFERRAL STATE VALUES (RESPONSE)	6.2.5.2.10	8.2.5.2.10
INQUIRE POLYLINE FACILITIES (REQUEST)	6.2.5.2.11	8.2.5.2.11
INQUIRE POLYLINE FACILITIES (RESPONSE)	6.2.5.2.12	8.2.5.2.12
INQUIRE PREDEFINED POLYLINE REPRESENTATION (REQUEST)	6.2.5.2.13	8.2.5.2.13
INQUIRE PREDEFINED POLYLINE REPRESENTATION (RESPONSE)	6.2.5.2.14	8.2.5.2.14
INQUIRE POLYMARKER FACILITIES (REQUEST)	6.2.5.2.15	8.2.5.2.15
INQUIRE POLYMARKER FACILITIES (RESPONSE)	6.2.5.2.16	8.2.5.2.16
INQUIRE PREDEFINED POLYMARKER REPRESENTATION (REQUEST)	6.2.5.2.17	8.2.5.2.17
INQUIRE PREDEFINED POLYMARKER REPRESENTATION (RESPONSE)	6.2.5.2.18	8.2.5.2.18
INQUIRE TEXT FACILITIES (REQUEST)	6.2.5.2.19	8.2.5.2.19
INQUIRE TEXT FACILITIES (RESPONSE)	6.2.5.2.20	8.2.5.2.20
INQUIRE PREDEFINED TEXT REPRESENTATION (REQUEST)	6.2.5.2.21	8.2.5.2.21
INQUIRE PREDEFINED TEXT REPRESENTATION (RESPONSE)	6.2.5.2.22	8.2.5.2.22
INQUIRE FILL AREA FACILITIES (REQUEST)	6.2.5.2.23	8.2.5.2.23
INQUIRE FILL AREA FACILITIES (RESPONSE)	6.2.5.2.24	8.2.5.2.24
INQUIRE PREDEFINED FILL AREA REPRESENTATION (REQUEST)	6.2.5.2.25	8.2.5.2.25
INQUIRE PREDEFINED FILL AREA REPRESENTATION (RESPONSE)	6.2.5.2.26	8.2.5.2.26
INQUIRE PATTERN FACILITIES (REQUEST)	6.2.5.2.27	8.2.5.2.27
INQUIRE PATTERN FACILITIES (RESPONSE)	6.2.5.2.28	8.2.5.2.28
INQUIRE PREDEFINED PATTERN REPRESENTATION (REQUEST)	6.2.5.2.29	8.2.5.2.29
INQUIRE PREDEFINED PATTERN REPRESENTATION (RESPONSE)	6.2.5.2.30	8.2.5.2.30
INQUIRE COLOUR FACILITIES (REQUEST)	6.2.5.2.31	8.2.5.2.31
INQUIRE COLOUR FACILITIES (RESPONSE)	6.2.5.2.32	8.2.5.2.32
INQUIRE PREDEFINED COLOUR REPRESENTATION (REQUEST)	6.2.5.2.33	8.2.5.2.33
INQUIRE PREDEFINED COLOUR REPRESENTATION (RESPONSE)	6.2.5.2.34	8.2.5.2.34
INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (REQUEST)	6.2.5.2.35	8.2.5.2.35
INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES (RESPONSE)	6.2.5.2.36	8.2.5.2.36
INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (REQUEST)	6.2.5.2.37	8.2.5.2.37
INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES (RESPONSE)	6.2.5.2.38	8.2.5.2.38
INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICE (REQUEST)	6.2.5.2.39	8.2.5.2.39
INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICE (RESPONSE)	6.2.5.2.40	8.2.5.2.40
INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (REQUEST)	6.2.5.2.41	8.2.5.2.41
INQUIRE DEFAULT LOGICAL INPUT DEVICE DATA (RESPONSE)	6.2.5.2.42	8.2.5.2.42

Appendix C 'Cross References'

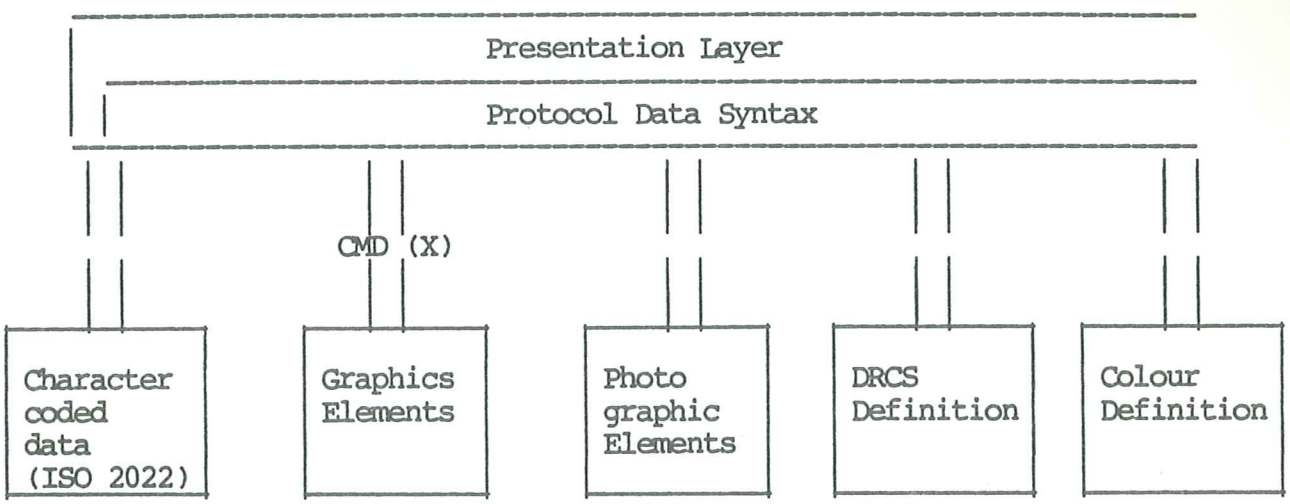
Primitives	CLAUSE OF DESCRIPTION	CLAUSE OF ENCODING
INQUIRES OF SEGMENT STATE LIST		
INQUIRE SET OF ASSOCIATED WORKSTATIONS (REQUEST)	6.2.5.3.1	8.2.5.3.1
INQUIRE SET OF ASSOCIATED WORKSTATIONS (RESPONSE)	6.2.5.3.2	8.2.5.3.2
INQUIRE SEGMENT ATTRIBUTES (REQUEST)	6.2.5.3.3	8.2.5.3.3
INQUIRE SEGMENT ATTRIBUTES (RESPONSE)	6.2.5.3.4	8.2.5.3.4
INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTE (REQUEST)	6.2.5.3.5	8.2.5.3.5
INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTE (RESPONSE)	6.2.5.3.6	8.2.5.3.6
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (REQUEST)	6.2.5.3.7	8.2.5.3.8
INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED (RESPONSE)	6.2.5.3.8	8.2.5.3.9
PIXEL INQUIRIES		
INQUIRE PIXEL (REQUEST)	6.2.5.4.1	8.2.5.4.1
INQUIRE PIXEL (RESPONSE)	6.2.5.4.2	8.2.5.4.2
INQUIRE PIXEL ARRAY (REQUEST)	6.2.5.4.3	8.2.5.4.3
INQUIRE PIXEL ARRAY (RESPONSE)	6.2.5.4.4	8.2.5.4.4
INQUIRE PIXEL ARRAY DIMENSION (REQUEST)	6.2.5.4.5	8.2.5.4.5
INQUIRE PIXEL ARRAY DIMENSION (RESPONSE)	6.2.5.4.6	8.2.5.4.6
PROTOCOL DESCRIPTORS		
SET DOMAIN RING	6.2.6.1	8.2.6.1
SET COLOUR HEADER	6.2.6.2	8.2.6.2
SET COORDINATE PRECISION	6.2.6.3	8.2.6.3
SET REAL PRECISION	6.2.6.4	8.2.6.4
SET COLOUR INDEX PRECISION	6.2.6.5	8.2.6.5

APPENDIX D

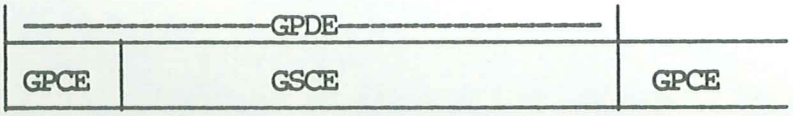
STRUCTURE OF THE PROTOCOL DATA SYNTAX

The graphic elements defined in this standard can be used together with other data elements.

Graphic elements are grouped together into Graphic Protocol Data Elements, GPDE as show below.



Each GPDE is introduced by a Graphics Protocol Control Element (GPCE), which includes a Coding Method Delimiter (CMD).



This structure allows to address Protocol Data Elements directly via the level of the Protocol Data Syntax, using the appropriate CMD of a GPCE.

The GPCE announces the category of the Protocol Data Elements contained in the Graphics Service Control Element (GSCE).

The above structure makes the encoding of the primitives independent of other coding methods.

The GPCE is in the form CMD (X), where CMD is represented by 1/15 and (X) indicates the category of the Protocol Data Elements, with the following assigned value : X = 3/0 Graphics Primitives

All data of a GSCE are regarded as graphic primitives. Selection of other Protocol Data Elements is only allowed after complete Graphics Protocol Data Elements.