EUROPEAN TELECOMMUNICATION STANDARD

ETS 300 073

November 1990

ICS: 33.020

Key words: Videotex

Videotex presentation layer data syntax
Geometric Display
(CEPT Recommendation T/TE 06-02, Edinburgh 1988)

ETSI

European Telecommunications Standards Institute

ETSI Secretariat

Postal address: F-06921 Sophia Antipolis CEDEX - FRANCE
Office address: 650 Route des Lucioles - Sophia Antipolis - Valbonne - FRANCE
X.400: c=fr, a=atlas, p=etsi, s=secretariat - **Internet:** secretariat@etsi.fr

Tel.: +33 92 94 42 00 - Fax: +33 93 65 47 16

0 Foreword

The text of the CEPT Recommendation T/TE 06-02 (Edinburgh 1988) was approved by the European Telecommunications Standards Institute (ETSI) as a European Telecommunication Standard (ETS) without any modification.

This ETS was recommended for endorsement by the Terminal Equipment (TE) Technical Committee of ETSI in May 1990 as part of an integrated package of 5 ETSs covering various aspects of videotex which comprises:

Final draft prETS 300 072	Terminal Equipment; Videotex presentation layer protocol Videotex presentation data layer syntax
Final draft prETS 300 074	Videotex presentation layer data syntax transparent data (CEPT Recommendation T/TE 06-03, Edinburgh 1988)
Final draft prETS 300 075	Terminal Equipment (TE); Videotex processable data
Final draft prETS 300 076	Terminal Equipment (TE); Videotex Terminal Facility Identifier (TFI)

Annex A

PART 2. GEOMETRIC DISPLAY

Table of contents

FIGURES

1. INTRODUCTION

ECMA-96 specifies a Graphics Data Syntax (GDS) 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 the graphics configuration. Following the GKS definition this allows advantage to be taken of the different capabilities of the various devices of which the graphics configuration is comprised.

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 Recommendation T. 101 (Data syntax II).

This part (part 2 of T/TE 06-02, geometric display) is a proper subset of GDS. This subset is defined in terms of GDS. All references to inquire functions and input functions are not applicable to the Videotex geometric display and should be ignored.

> *NOTE*
> *Wherever GDS is mentioned in this document it also reflects to this part 2 of T/TE 06-02.*

2. SCOPE AND FIELD OF APPLICATION

This document specifies the set of functions to be used in a graphics configuration and their encoding in a 7-bit or 8-bit environment. In addition the code tables are structured in accordance with ISO 646.

The intention of this document is to facilitate data interchange, not to standardize equipment. The specification of the concepts is included only to delimit the field of application. The definitions of the primitives may not be applicable to a graphics configuration, 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 graphics configuration depends on the required GKS level.

Figure A2-1 (T/TE 06-02) 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 graphics configuration through the multiple workstation interface. Above the multiple workstation interface are the GKS normalization transformations, which convert world coordinates to normalized device coordinates. Below the multiple workstation interface are the GKS workstations, which are connected to and driven by GKS. The workstations are mapped to the graphics configuration and the normalized device coordinates are transformed to device coordinates by the workstation transformation. Non-existent capabilities of the graphics configuration must be simulated by using emulation software.

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.*
— ISO 646 INFORMATION PROCESSING – *ISO 7-bit coded character set for information interchange.*
— ISO 2022 INFORMATION PROCESSING – *ISO 7-bit and 8-bit coded character sets. Code extension techniques.*
— ISO DP 8632 INFORMATION PROCESSING—COMPUTER GRAPHICS – *Metafile for the storage and transfer of picture description information.*
— CCITT-Recommendation T.101 – *International interworking for Videotex services.*
— ECMA-96 - *Syntax of graphical data for a multiple workstation interface.*

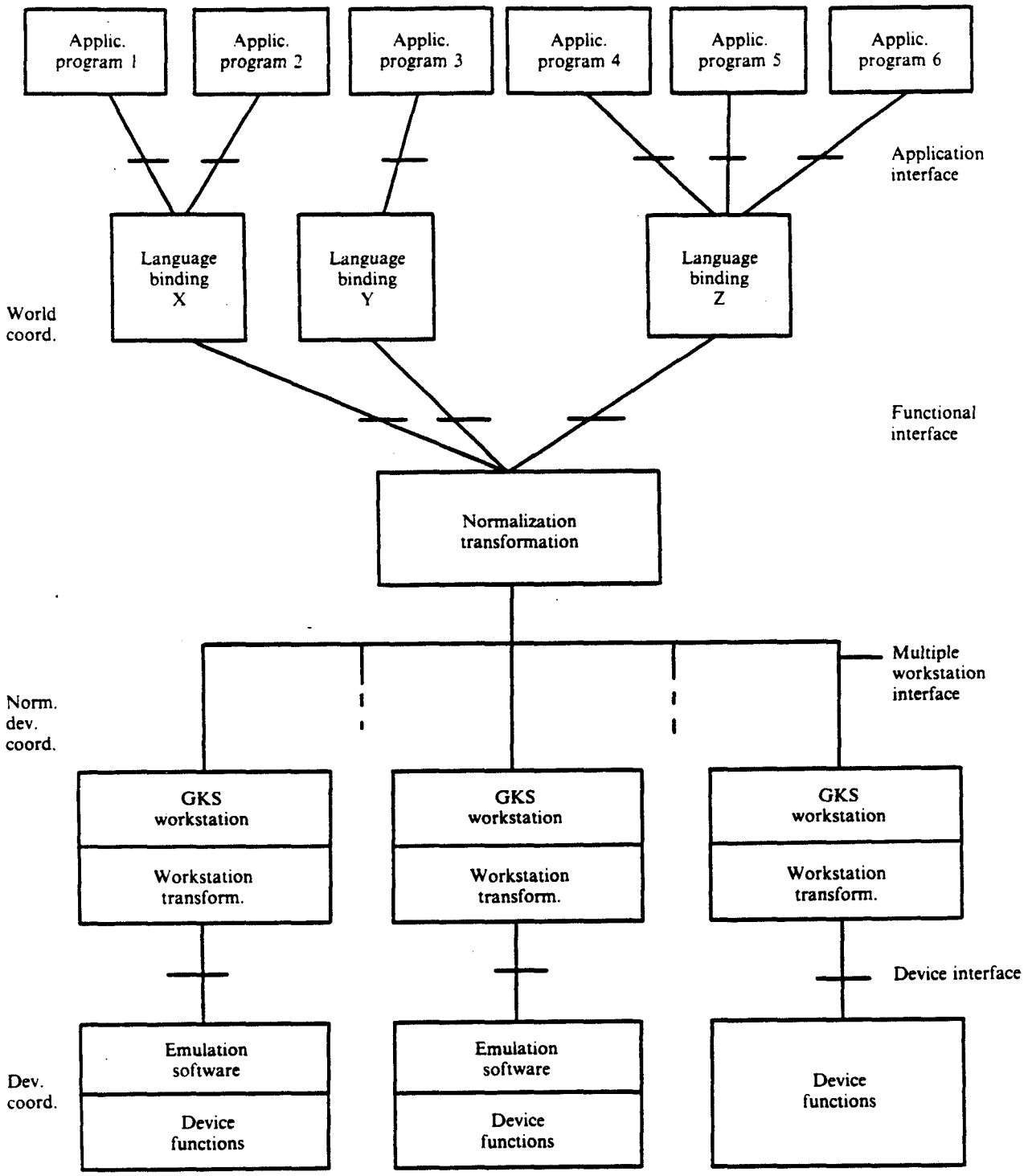


Figure A2-1 (T/TE 06-02). Model describing the GKS environment and its interfaces.

4. DEFINITIONS AND ABBREVIATIONS

This clause contains the definition and abbreviations 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 primitive) 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 an output primitive. Bundle tables exist for the following output primitives: POLYLINE, POLYMARKER, TEXT and FILL AREA.

CELL ARRAY
A 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 to 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 graphic 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
The portion of the device space that corresponds to the area available for displaying images.

DISPLAY SURFACE; VIEW SURFACE
The medium in a display device on which the display images may appear.
(For example: the screen of a display, the paper in a plotter.)

DOMAIN RING
A mechanism for defining the ring used in encoding incremental mode 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, which are otherwise not addressable by any primitive.

GENERALIZED DRAWING PRIMITIVE (GDP)
A display element (graphics primitive) used to address special geometric workstation capabilities such as curve drawing.

GKS INSTANCE
A combination of GKS and one or more graphics configurations.

GRAPHICAL KERNEL SYSTEM (GKS)
The application programmer's interface to graphics defined in ISO 7942.

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, STROCKE, 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 class 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 (primitive) that actually generates (parts of) a display image. These are: POLYLINE, FILL AREA, POLYMARKER, CELL ARRAY, TEXT and GDPs.

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 scale factor 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 scale factor and colour index.

PRIMITIVE

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

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 output primitives 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 with varying position (translation), size (scaling), 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 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 determines whether a segment is 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 the 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. **Abbreviations**

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
DC	Device Coordinates
DCC	Differential Chain Code
GDS	Graphics Data Syntax for a multiple workstation interface
GDP	Generalized Drawing Primitive
GKS	Graphical Kernel System
IMM	IMMediately
INPUT	INPUT only workstation
IRG	Implicit ReGenération
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 graphic 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 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, i.e. graphics configuration.

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 in 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 the 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 A2-2 (T/TE 06-02).

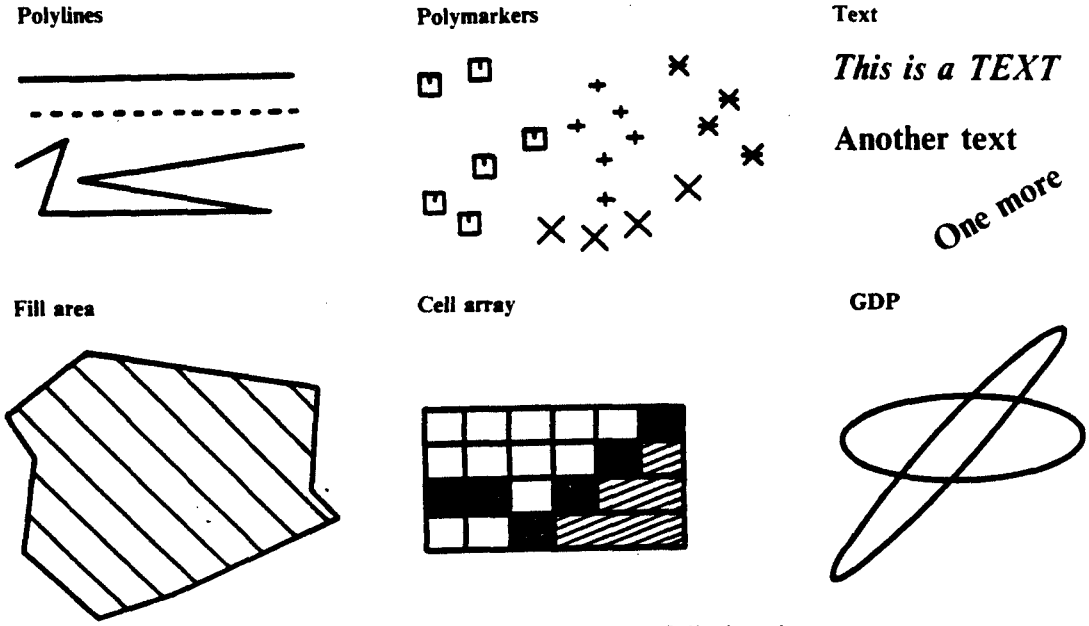


Figure A2-2 (T/TE 06-02). Examples of display elements.

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.

— FILL AREA
The display element is an area closed by a set of connected straight lines that may be hollow or filled with a 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 (GDPs)
A general display element addressing special geometrical drawing capabilities of a graphics terminal. All transformations are applied to the points of a 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 (CENTRE/3 POINT)
The display element is a circular arc.

- CIRCULAR ARC CHORD (CENTRE/3 POINT)
  The display element is an area closed by a circular arc and a chord that may be hollow or filled.
- CIRCULAR ARC PIE (CENTRE/3 POINT)
  The display element is an area closed by a circular arc and the two radii that may be hollow or filled.
- 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 elliptic 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. Non-geometric 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 Drawing Primitives 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 bundel 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 mode 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 aspects 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 or 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 a unique table (colour table) on each workstation. Similarly other attribute 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 facilities.

(a) POLYLINE	POLYLINE INDEX
	LINE TYPE
	LINE WIDTH SCALE FACTOR
	POLYLINE COLOUR INDEX
	LINE TYPE ASF
	LINE WIDTH SCALE FACTOR 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 SCALE FACTOR 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 DRAWING PRIMITIVE	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	set a)
	CIRCULAR ARC 3 POINT CHORD	set d)
	CIRCULAR ARC 3 POINT PIE	set d)
	CIRCULAR ARC CENTRE	set a)
	CIRCULAR ARC CENTRE CHORD	set d)
	CIRCULAR ARC CENTRE PIE	set d)
	ELLIPSE	set d)
	ELLIPTIC ARC	set a)
	ELLIPTIC ARC CHORD	set d)
	ELLIPTIC ARC PIE	set d)
	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 bundle table
Colour table

The values in these tables may be changed. The criterion "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. The predefined entries with indices up to the minimum number of predefined entries at a given level (see 5.10.) must be distinguishable from each other.

5.1.2.1. POLYLINE attributes

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

— POLYLINE COLOUR INDEX
Determines the POLYLINE 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 width of the line to be used. The line width is calculated as a nominal line width multiplied by the line width scale factor. This value is mapped by the workstation to the nearest available line width. It is controlled with SET LINE WIDTH SCALE FACTOR and used if the "line width scale factor" 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 and 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
Determines the size of the marker, e.g. height and width. The marker size is calculated as a nominal size multiplied by the marker size scale factor. This size is mapped by the workstation to the nearest available size. The size is defined with SET MARKER SIZE SCALE FACTOR and is used when the "marker size scale factor" 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 character height vector fixes the up direction of the character. The length of the character height vector specifies the distance from baseline to capline along the up direction of the character. 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 height ratio of the character.

As a result of a segment transformation, the positive angle from the height vector to the width vector can be greater than 180 degrees. In this case the characters are mirror-imaged and the notions of right and left used for TEXT PATH and TEXT ALIGNMENT are reversed.

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. opposite 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 TEXT COLOUR INDEX of the generated text string. SET TEXT COLOUR INDEX specifies the TEXT 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:

	Normal
TEXT PATH	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 base line, a font half line, a capline and a centre line (see Figure A2-3 (T/TE 06-02)).

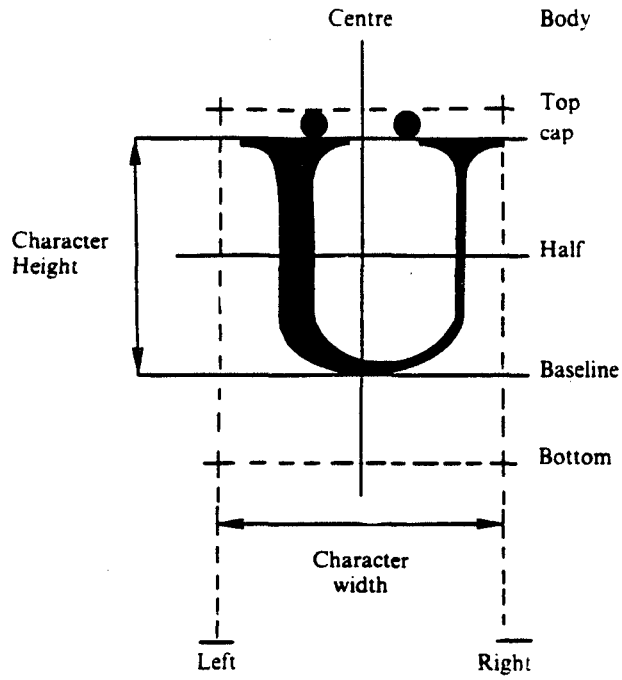


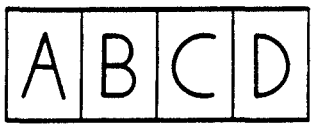
Figure A2-3 (T/TE 06-02). Font description coordinate system.

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.

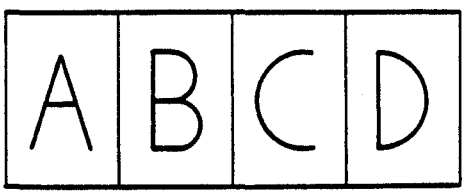
The height of a character in the font coordinate system is given by the height from the font baseline to the capline. The width may include space on either side of the character. It is given by the width of the character body. It is assumed that the characters lie within their body, except that kerned 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.

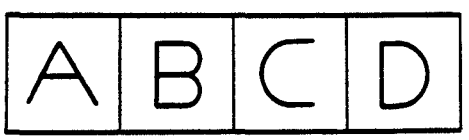
The Figures A2-4 (T/TE 06-02) to A2-8 (T/TE 06-02) are only inserted for clarification purposes. They show the effects of the different attributes on the display of the text "ABCD".



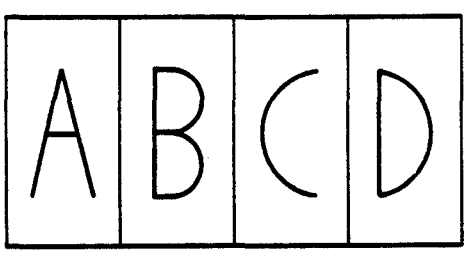
CHARACTER VECTORS = (0.0, 0.025) and (0.025, 0.0)
CHARACTER EXPANSION FACTOR = 1.0



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

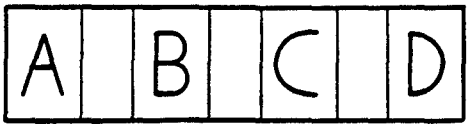


CHARACTER VECTORS = (0.0, 0.025) and (0.025, 0.0)
CHARACTER EXPANSION FACTOR = 1.5



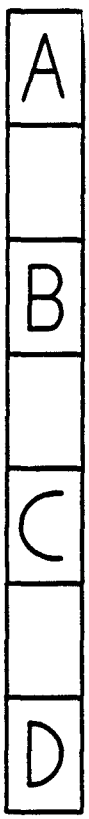
CHARACTER VECTORS = (0.0, 0.050) and (0.050, 0.0)
CHARACTER EXPANSION FACTOR = 0.75

Figure A2-4 (T/TE 06-02). CHARACTER VECTORS and CHARACTER EXPANSION FACTOR.



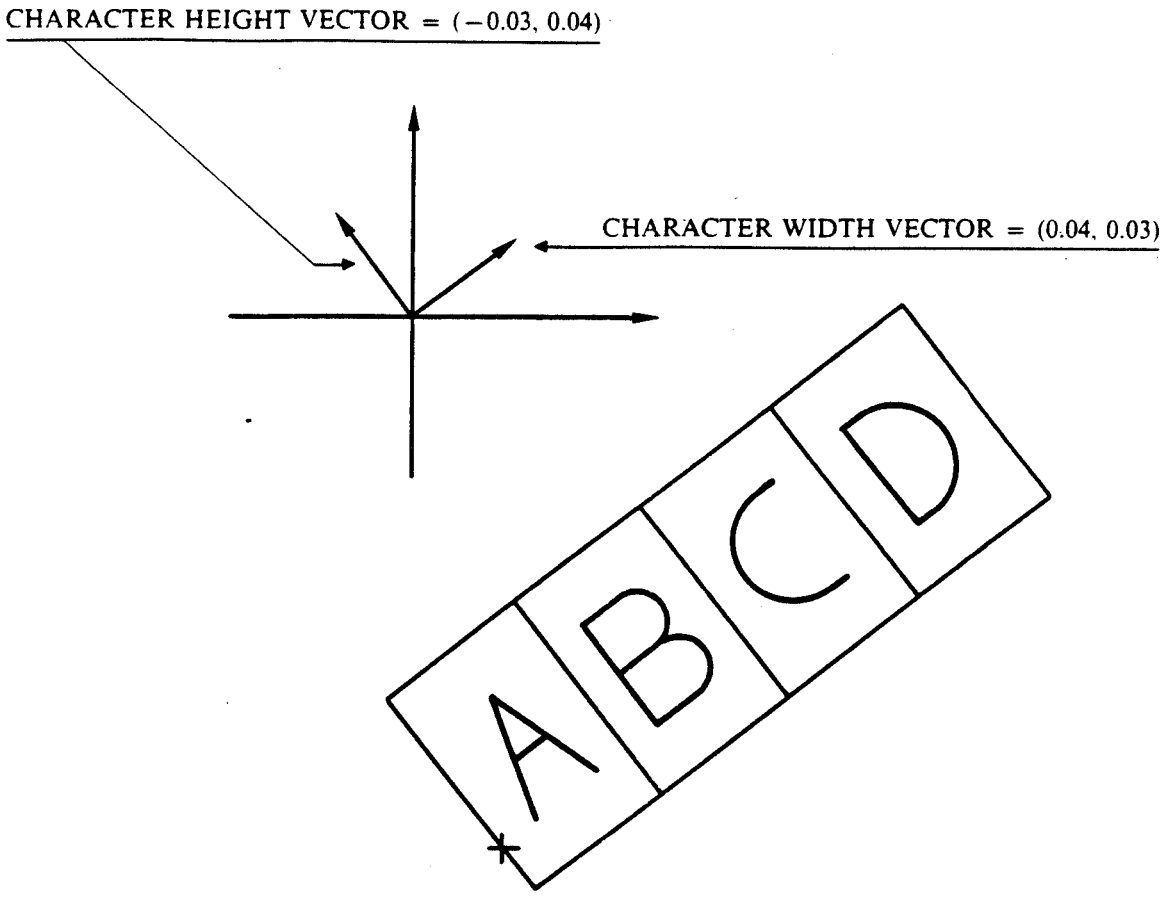
CHARACTER VECTORS = (0.0, 0.025) and (0.025, 0.0)
CHARACTER SPACING = 0.67
TEXT PATH = right

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



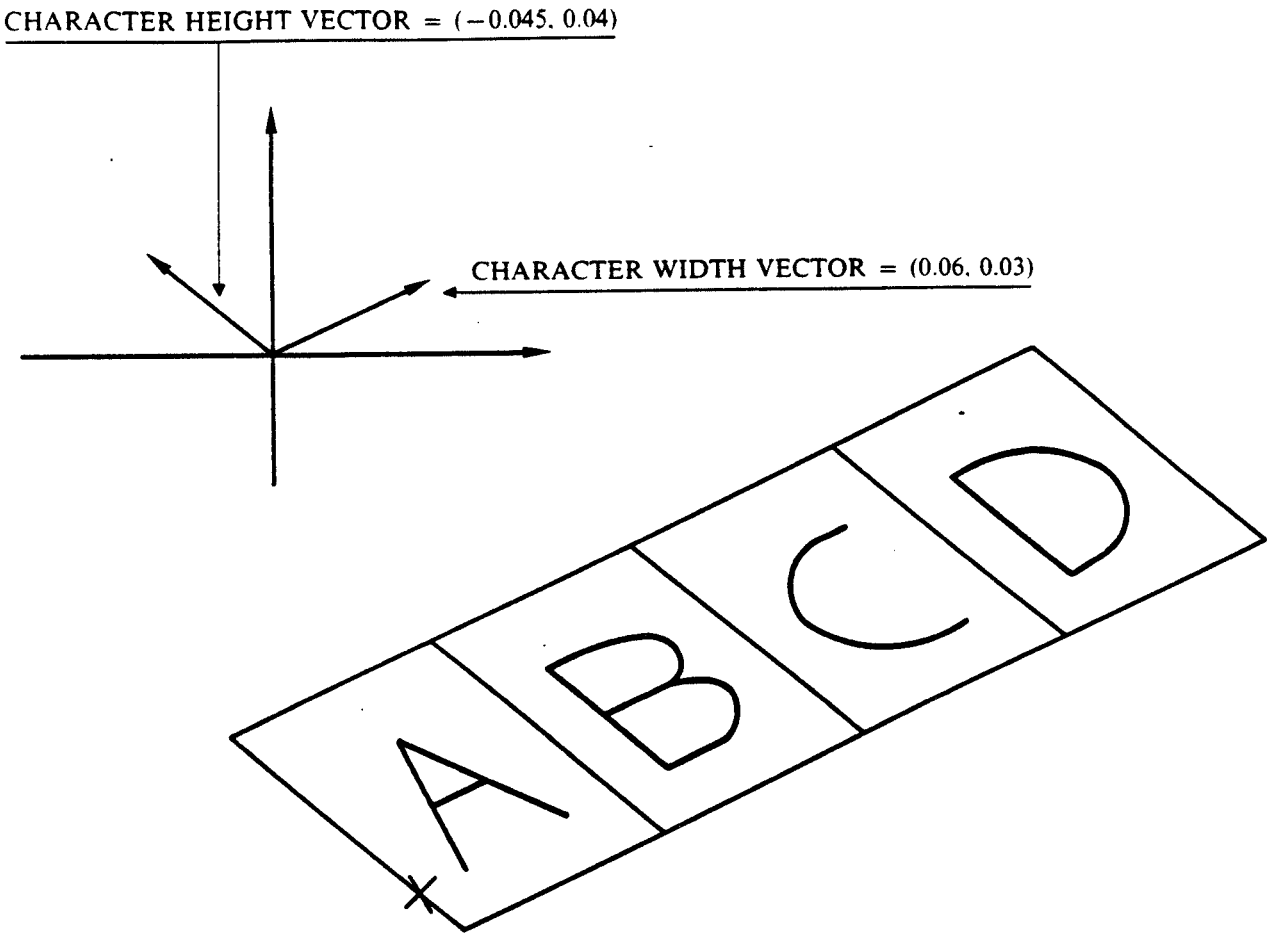
CHARACTER VECTORS = (0.0, 0.025) and (0.025, 0.0)
CHARACTER SPACING = 2.0
TEXT PATH = down

Figure A2-5 (T/TE 06-02). CHARACTER SPACING and TEXT PATH.



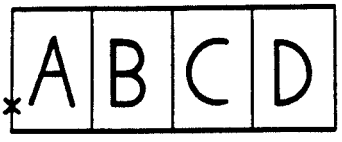
CHARACTER VECTORS = (−0.03, 0.04) and (0.04, 0.03)
TEXT PATH = right

Figure A2-6 (T/TE 06-02). CHARACTER VECTORS.

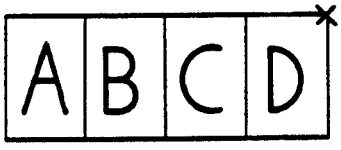


CHARACTER VECTORS = (−0.045, 0.04) and (0.06, 0.03)
TEXT PATH = right

Figure A2-7 (T/TE 06-02). CHARACTER VECTORS after anisotropic transformation.

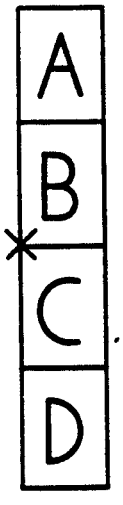


TEXT ALIGNMENT = (left, base)
TEXT PATH = right



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

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



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

Figure A2-8 (T/TE 06-02). TEXT ALIGNMENT and TEXT PATH.

Text strings are delimited by the OPEN CHARACTER STRING (OCS) and STRING TERMINATOR (ST) (see section 7.2.4.). These delimiters are not considered a part of the text string.

The characters in the text string can be defined in a 7-bit or 8-bit environment defined by ISO 2022. 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.

Besides the characters from the in-use G-set, in a 7-bit environment, the shift functions contained in Table A2-1 (T/TE 06-02) may be used for invocation purposes, ESC = 1/11. The coded representation of these shift functions is defined in part 1 of this document.

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 A2-1 (T/TE 06-02). Permitted invocation sequences in a 7-bit environment.

In an 8-bit environment the shift functions contained in Table A2-2 (T/TE 06-02) may be used for invocation purposes, see Table A2-2 (T/TE 06-02).

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 A2-2 (T/TE 06-02). Permitted invocation sequences in an 8-bit environment.

Characters in the text string not from the invoked G-set or not from Table A2-1 (T/TE 06-02), in a 7-bit environment, will be ignored. In an 8-bit environment characters not from the invoked G-sets or not from Table A2-2 (T/TE 06-02) will be ignored.

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) FILL AREA 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 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 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 style 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 parallelogram, 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 boundary by conceptually replicating the transformed pattern box in directions 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
ARC ELLIPTIC ARC CHORD
ELLIPTIC ARC PIE

use the FILL AREA attributes. These are described in section 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 section 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 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 intensity available.

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 output primitives 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 graphics configuration 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 manipulations 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);
(c) Last error condition (e.g. error in parameter, etc.).

The GDS state list can be inquired by a 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 graphics configuration. 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 changes accordingly 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 on 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 (BGU) 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:
1. The DC origin is at the bottom left corner of the display space;

2. 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;
3. x and y increase 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] × [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] × [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 Figure A2-9 (T/TE 06-02)).

Window and viewport limits specify rectangles parallel to the coordinate axes in NDC and DC. The rectangle includes their boundaries. To ensure that no output outside the workstation window is displayed, the picture is clipped at the workstation window boundaries, and this clipping cannot be disabled.

If the workstation window and the workstation viewport have different aspect ratios, the specified scaling would be different on each axis, if the window was mapped onto the viewport in its entirety. To ensure equal scaling on each axis, the transformation maps the window onto the largest rectangle that can fit within the viewport such that (see Figure A2-10 (T/TE 06-02))
— the 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 and 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 display 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 Figure A2-11 (T/TE 06-02).

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.

The precision of the displacement mode coordinates can be specified by SET COORDINATE PRECISION.

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:
— BASIC RADIUS
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.

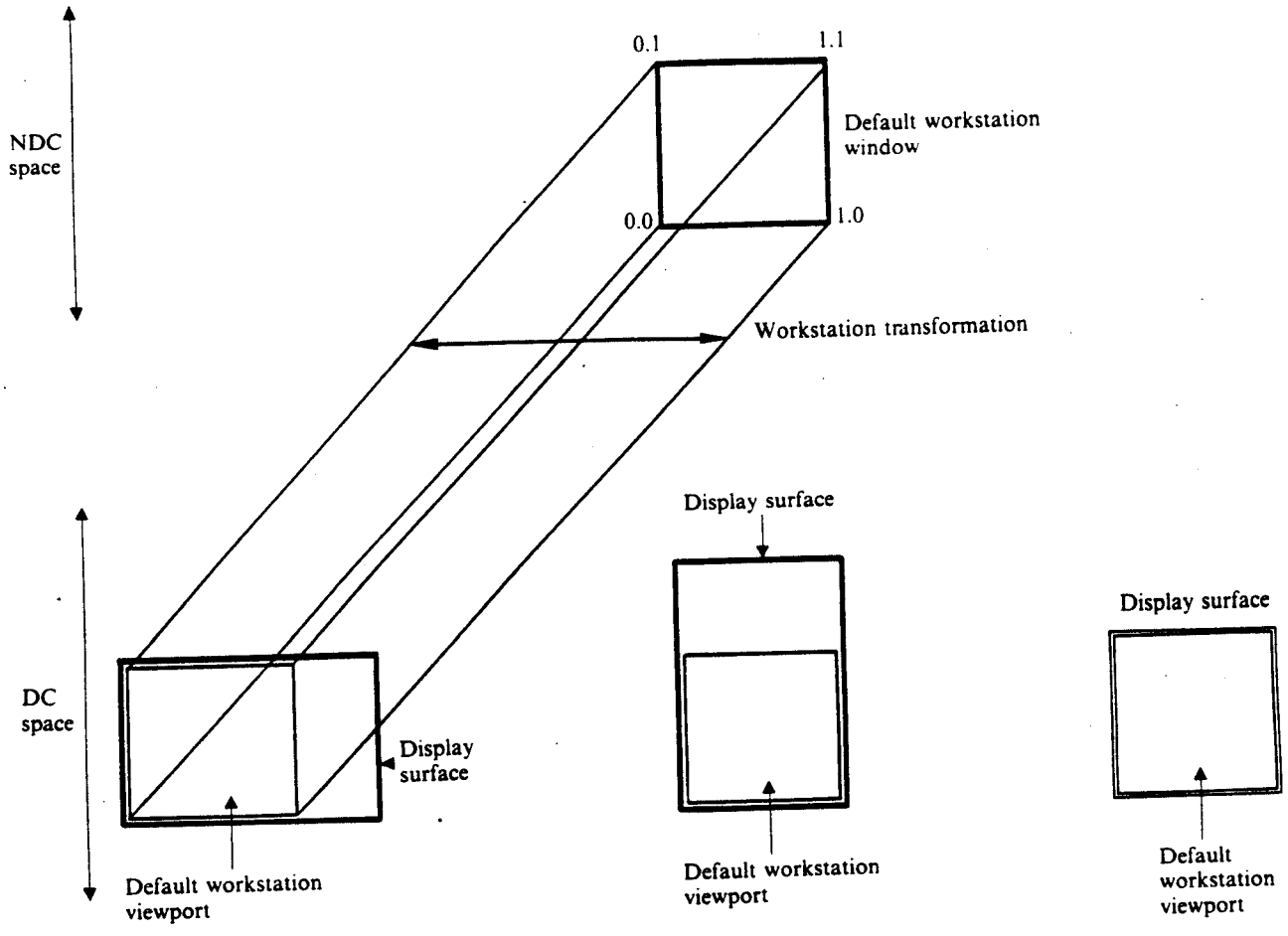


Figure A2-9 (T/TE 06-02). Default workstation transformation.

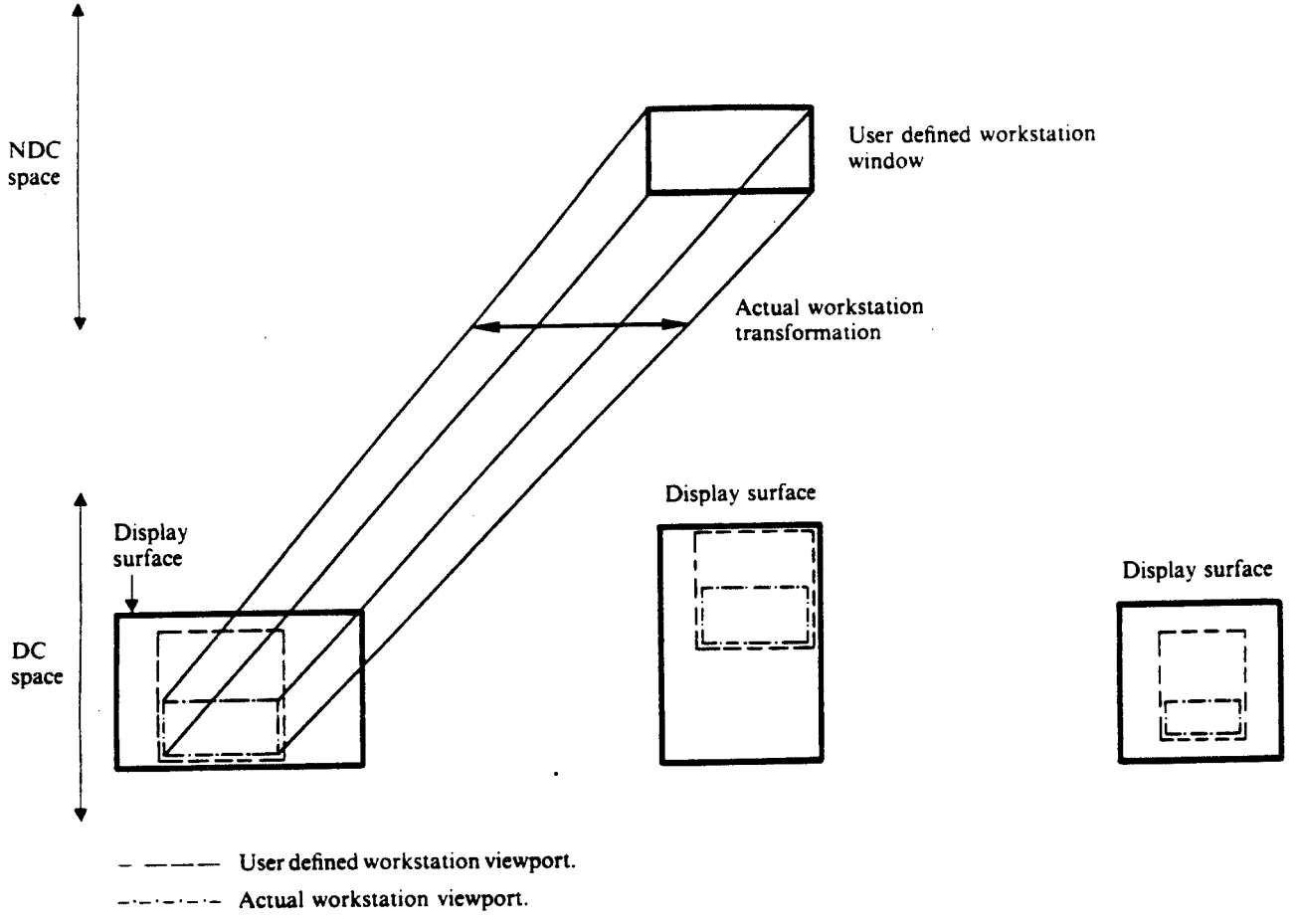


Figure A2-10 (T/TE 06-02). Workstation transformation with anisotropic workstation window and workstation viewport.

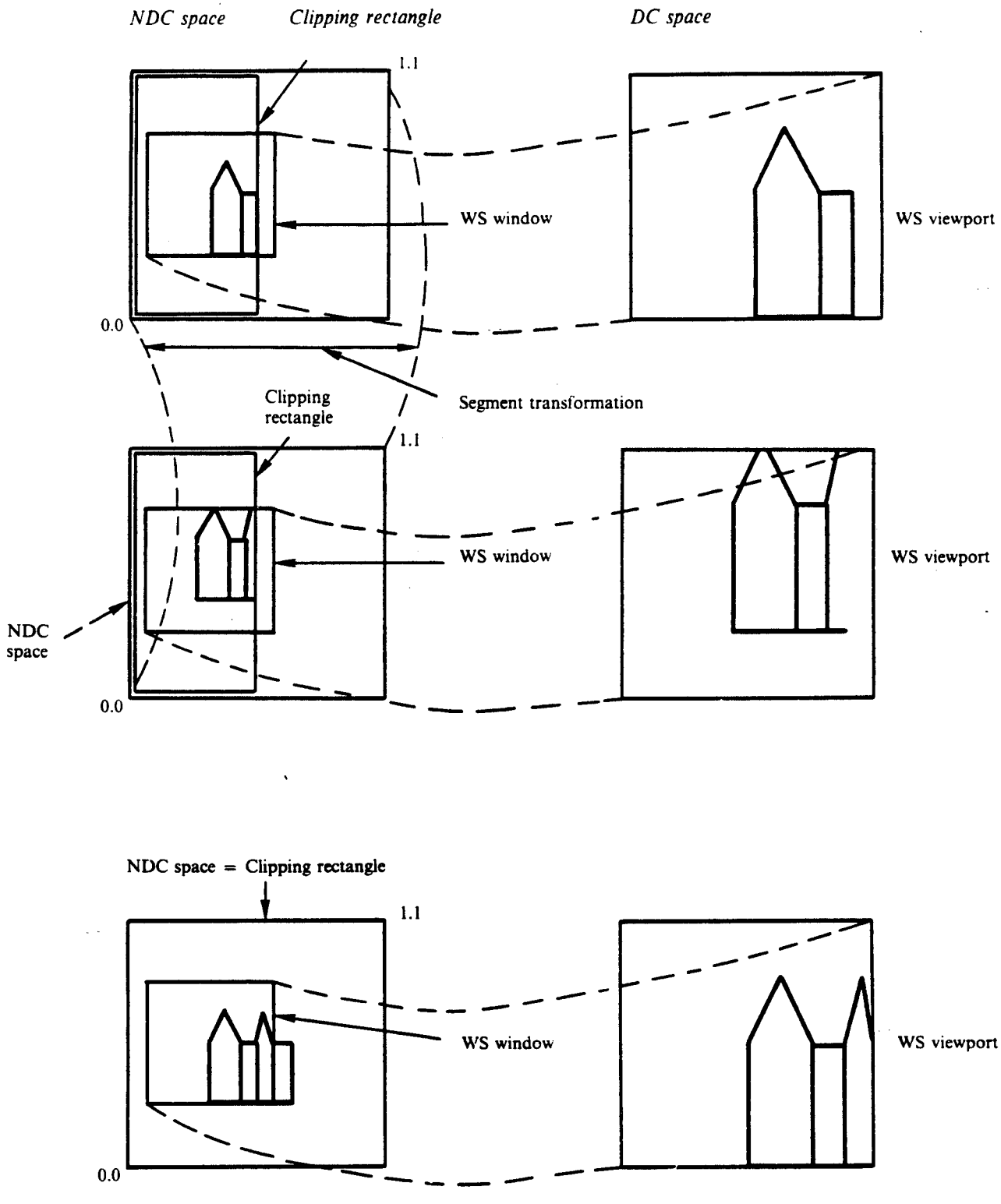


Figure A2-11 (T/TE 06-02). Clipping and workstation transformation.

A segment may be:
— transformed;
— made visible or invisible;
— highlighted or not;
— assigned different priorities;
— made detectable or undetectable;
— deleted;
— renamed;
— inserted into the open egment 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. 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 A2-12 (T/TE 06-02).

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 was 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 the 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 itself cannot 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.

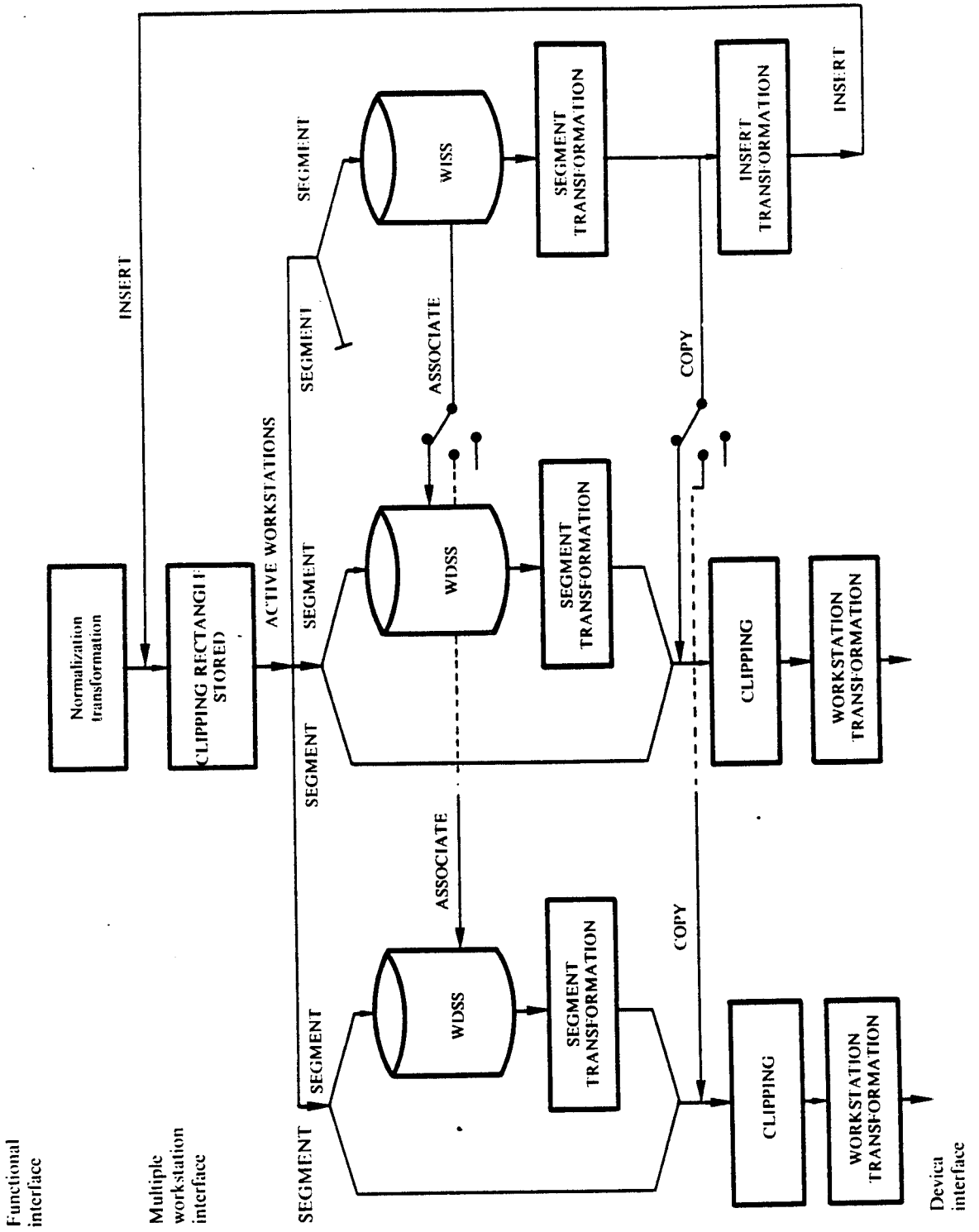


Figure A2-12 (T/TE 06-02). Data flow chart for graphical output.

A workstation state variable controlling if and how long such a delay of pictures 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 segments 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 System 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 ON WORKSTATION. 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 HIGHLIGHTING

(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*

A Videotex environment will contain no workstations of the type INPUT, so this section is not applicable for Videotex purposes.

5.6.2. *Logical input device model*

A Videotex environment will contain no workstations of the type INPUT, so this section is not applicable for Videotex purposes.

5.6.3. *Measures of each input class*

A Videotex environment will contain no workstations of the type INPUT, so this section is not applicable for Videotex purposes.

5.6.4. *Input queue and current event report*

A Videotex environment will contain no workstations of the type INPUT, so this section is not applicable for Videotex purposes.

5.6.5. *Initialization of input devices*

A Videotex environment will contain no workstations of the type INPUT, so this section is not applicable for Videotex purposes.

5.7. **Inquiry**

A Videotex environment will support no inquire primitives, so this section is not applicable for Videotex purposes.

5.8. **Error detection**

A Videotex environment will support no interactive synchronous error detection mechanism in the sense of what is defined within GKS, so this section is not applicable for Videotex purposes.

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. a marker type 6 will default to a marker type 2);
— if a primitive is received that is not supported, the primitive is ignored;
— a primitive which 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.

These general error handling rules may be overruled by specific error handling as described in the discussion sections of clause 6.

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 specification 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 configurations.

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: Minimal 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. Each graphics configuration in conformance with a specific level supports the capability at that level.
— An explicitly defined and non-required capability. A graphics configuration 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 configuration may provide the capability. Its implementation follows general rules given by the concepts (see section 5.2.) and functional definitions.

The set of explicitly defined and required capabilities includes:
(a) predefined bundle entries up to the required minimum;
(b) line types: solid, dashed, dotted and dashed-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 at that level. However, ranges of parameters may contain additional explicitly 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 includes 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 A2-3 (T/TE 06-02) 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	Input level b	Input level 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 A2-3 (T/TE 06-02). 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 A2-4 (T/TE 06-02) 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	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	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 factors (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	2	2	2	6	6	6	6	6	6
Settable text bundles	–	–	–	20	20	20	20	20	20
Predefined patterns (see *Note 2*)	1	1	1	1	1	1	1	1	1
Settable patterns (see *Notes 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	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	–	–	–	–	–	–	1	1	1

0 indicates explicitly defined and non-required at that level.

– indicates not defined at that level.

Table A2-4 (T/TE 06-02). 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 graphics configuration.

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 indices encoded in different ways
CD	Colour Direct	Colour definition with R, G and 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 fractional part
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: nP or 2R, 2I). Also, lists of types can be expressed (for example: I, E, R, E). nP with an unspecified number n denotes an implementation or application dependent number of points called a point list. nP with a specified number n denotes individual points.

6.2. Geometric primitives

6.2.1. *Workstation management primitives*

6.2.1.1. OPEN WORKSTATION

Level: 0a

Parameters: Workstation identifier (ID)

Description: 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 ensures that the display surface is cleared, but does not clear the display 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: An implicit UPDATE WORKSTATION, with the update regeneration flag set to PERFORM, is performed for the specified workstation. The workstation state list is deallocated. The workstation identifier is deleted from the set of open workstations in the GDS state list and from the set of associated workstations in the segment state list of every segment containing it. If the set of associated workstations of a segment becomes empty, the segment is deleted. 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: Output primitives are sent to and segments are stored on all active workstations. 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 effect:
— the specified workstation is not activated;
— 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: If the workstation identifier parameter is out of range the primitive is ignored.

6.2.1.6. SET DEFAULTS

Level: 0a

Paramaters: 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. GDS ESCAPE 1

Level: 0a

Paramaters: — GDS escape identifier (I)
— GDS data record (REC)

Description: The GDS ESCAPE 1 primitive allows use of device capabilities not specified by this standard. This primitive is best suited for access to non-standardized control features of graphics devices.
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.

6.2.2. *Output workstation primitives*

6.2.2.1. Output drawing primitives

6.2.2.1.1. POLYLINE

Level: 0a

Paramaters: Point list (2..n) (nP)

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

Discussion: If only one point is specified the primitive is ignored. The implementation of a zero length line segment is implementation dependent.

6.2.2.1.2. POLYMARKER

Level: 0a

Paramaters: Point list (1..n) (nP)

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

Discussion: None.

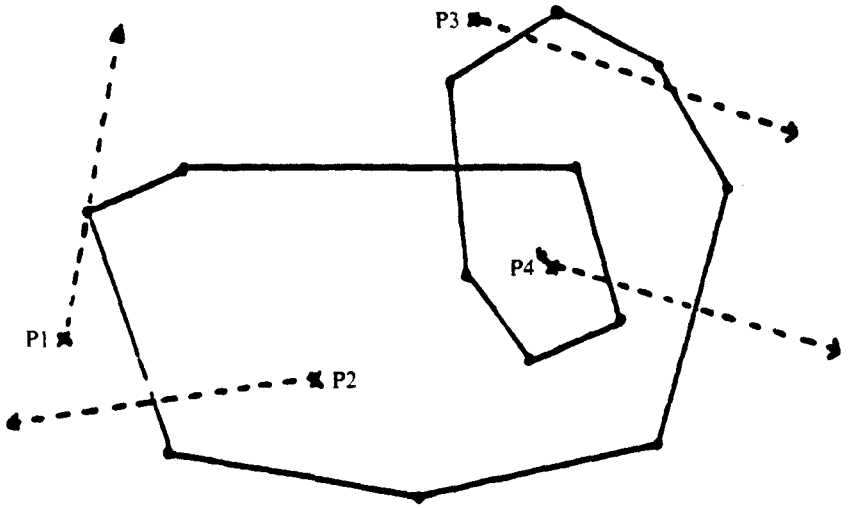
6.2.2.1.3. FILL AREA

Level: 0a

Parameters: Point list (3..n) (nP)

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 tangentially, the intersection count is not changed. If a polygon is clipped and subareas are generated, the new boundaries become part of the polygon boundaries.

An example is given below:



Points P3 and P4 are considered to be outside the polygon, because the intersection count is even (=2). Point P2 is considered to be inside the polygon, because the intersection count is odd (=1). Point P1 is considered to be outside the polygon, because the intersection count is even (=0). The line to infinity from P1 passes through a vertex point tangentially, but this does not affect the intersection count.

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 FILL AREA INTERIOR STYLE primitives. If a point is within the polygon, it is included in the area to be filled subject to the following rules for boundaries. The boundary is drawn for interior style HOLLOW and is not drawn for other interior styles.

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

6.2.2.1.4. TEXT

Level: 0a

Parameters:
— Text position (P)
— Character string (S)

Description:
The character codes specified in the string are interpreted to obtain the associated symbols. Characters are displayed on the viewing surface as specified by the text attributes.
The characters are dimensioned 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 ASPECT 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) (3P)
— Number of cells in P-Q 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 of the parallelogram. R defines a third corner.
This parallelogram is subdivided in m ★ n contiguous parallelograms in the following way. The side PQ of the parallelogram is subdivided into m intervals of equal size. The side QR of the parallelogram is subdivided into n intervals of equal size. The grid, implied by this subdivision, consists of m ★ n equally dimensioned cells.
The cell colour index list consists of m ★ n colour indices, conceptually an array of dimensions m and n representing respectively the column and row dimensions. Array element (1,1) is mapped to the cell associated with P and array element (m,1) is mapped to the cell associated with R. Array element (m,n) is mapped to the cell associated with Q. Hence, the colour elements are mapped within rows running from P to R, and with the rows incrementing in order from R to Q. This is illustrated in Figure A2-13 (T/TE 06-02).
No current colour setting is changed.

Related primitives: None.

Discussion: The parameters 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 ★ 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 ★ n, the exceeding indices are ignored.
If the number of cells (parameters m and n) are less than or equal to 0 the primitive is ignored.

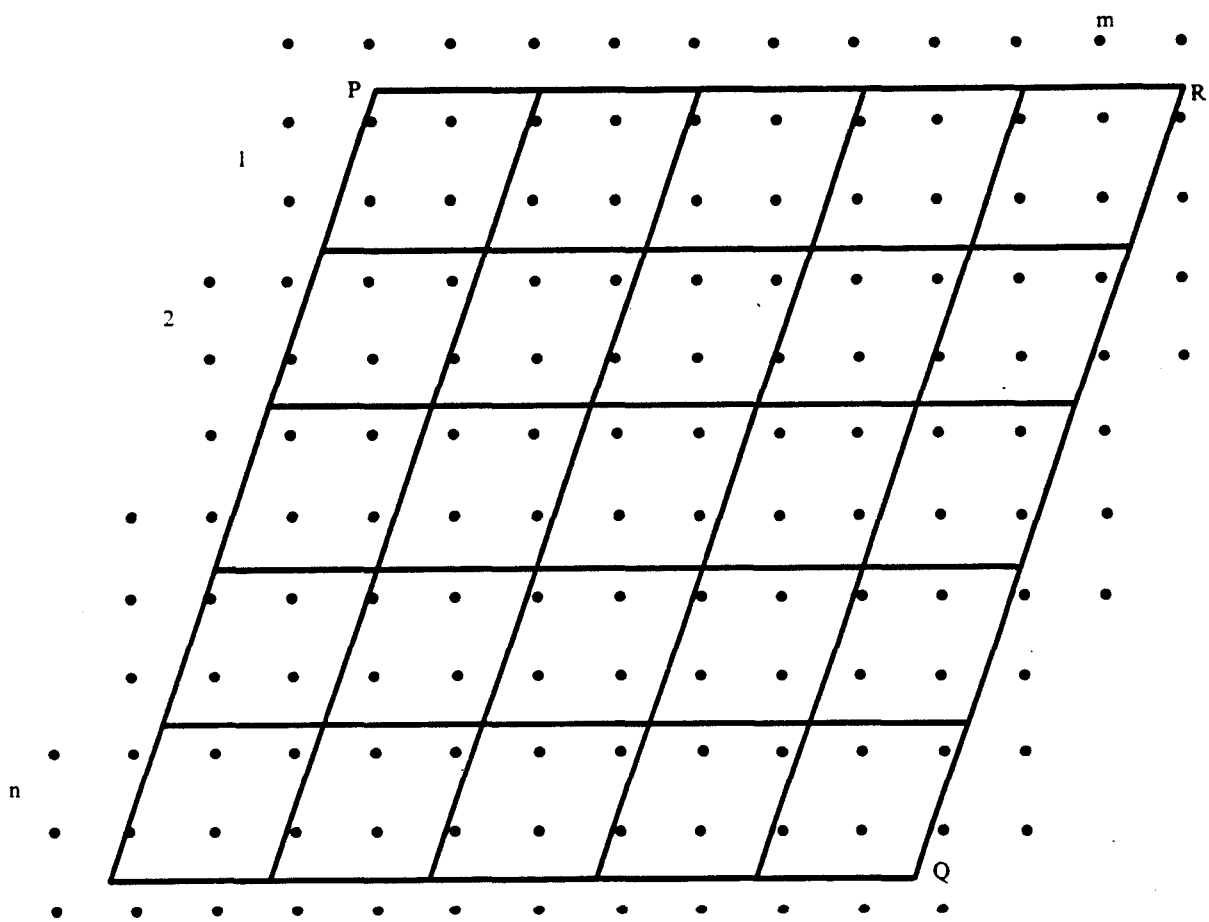


Figure A2-13 (T/TE 06-02). m by n parallelograms mapped onto the display surface.

6.2.2.1.6. GDP

Level: 0a

Parameters:
— GDP identifier (I)
— Point list (nP)
— Data record (REC)

Description: A Generalized Drawing Primitive (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 different scaling for the two axes.
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 (I)
(= rectangle)
— Point list (2P)

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 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.

Discussion: If the two points define a line parallel to either the x or y axis, the degenerated form of the rectangle is the line.
If the two points are identical, the degenerate form of the rectangle is the defined point.

6.2.2.1.6.2. GDP (circle)

Level: 0a

Parameters:
— GDP identifier (I)
(= circle)
— 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.

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 (I)
(= circular arc 3 point)
— Starting point, intermediate point, ending point (3P)

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 the 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.
It 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 (I)
(= circular arc 3 point chord)
— Starting point, intermediate point, ending point (3P)

Description: A circular arc 3 point chord defined by three points (starting point, intermediate point and ending point) is a boundary closed by the circular 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 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 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 drawn.
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.

6.2.2.1.6.5. GDP (circular arc 3 point pie)

Level: 0a

Parameters: — GDP identifier (I)
(= circular arc 3 point pie)
— Starting point, intermediate point, ending point (3P)

Description: A circular arc 3 point pie defined by three points (starting point, intermediate point, ending point) 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 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 arc 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 drawn.
If the three points coincide a dot is displayed.
If the three points are colinear the primitive is ignored.

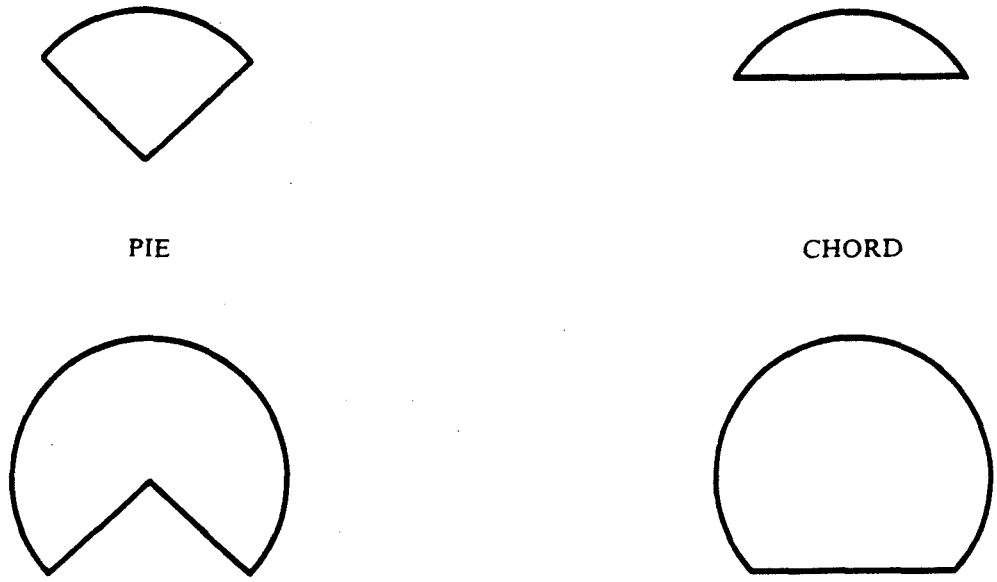


Figure A2-14 (T/TE 06-02). GDP specifications for "arc pie" and "arc chord".

6.2.2.1.6.6. GDP (circular arc centre)

Level: 0a

Parameters:
— GDP identifier (I)
(= circular arc centre)
— Centre (P)
— Radius (V)
— Start vector (P)
— End vector (P)

Description: The radius parameter and the centre parameter together define a circle. The fourth 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 counterclockwise 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 end vector coincide the primitive is ignored.
If a segment transformation is applied to the points defining a circular arc centre, the transformed points again define a circular arc centre in the same way as described above. This may result in unwanted effects.

6.2.2.1.6.7. GDP (circular arc centre chord)

Level: 0a

Parameters:
— GDP identifier (I)
(= circular arc centre chord)
— 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 FILL AREA.

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.
If a segment transformation is applied to the points defining a circular arc centre chord, the transformed points again define a circular arc centre chord in the same way as described above. This may result in unwanted effects.

6.2.2.1.6.8. GDP (circular arc centre pie)

Level: 0a

Parameters:
— GDP identifier (I)
(= circular arc centre pie)
— 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.
If a segment transformation is applied to the points defining a circular arc centre pie, the transformed points again define a circular arc centre pie in the same way as described above. This may result in unwanted effects.

6.2.2.1.6.9. GDP (ellipse)

Level: 0a

Parameters:
— GDP identifier (I)
(= ellipse)
— Centre (P)
— Endpoints (X1, Y1) and (X2, Y2) (2P)

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:
$X = XM + DX1 \star \cos(t) + DX2 \star \sin(t)$
$Y = YM + DY1 \star \cos(t) + DY2 \star \sin(t)$
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 $t = 2\star\pi$ 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 colinear a straight line connecting the two endpoints is displayed.
If any two of the three points are coinciding the primitive is ignored.

6.2.2.1.6.10. GDP (elliptic arc)

Level: 0a

Parameters:
— GDP identifier (I)
(= elliptic arc)
— Centre point (P)
— Endpoints (X1, Y1) and (X2, Y2) (2P)
— T-start, T-end (2R)

Description: The centre point and the two endpoints together define an ellipse (see 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 POLYLINE.

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-end the full ellipse is displayed.

6.2.2.1.6.11. GDP (elliptic arc chord)

Level: 0a

Parameters:
— GDP identifier (I)
(= elliptic arc chord)
— Centre point (P)
— Endpoints (X1, Y1) and (X2, Y2) (2P)
— T-start, T-end (2R)

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 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 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-end the full ellipse is displayed.

6.2.2.1.6.12. GDP (elliptic arc pie)

Level: 0a

Parameters:
— GDP identifier (I)
(= elliptic arc pie)
— Centre point (P)
— Endpoints (X1, Y1) and (X2, Y2) (2P)
— T-start, T-end (2R)

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 and 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-end the full ellipse is displayed.

6.2.2.1.6.13. GDP (spline)

Level: 0a

Parameters:
— GDP identifier (I)
(= spline)
— Point list (3..n) (nP)

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 (IX)
— Line type (I)
— Line width scale factor (R)
— 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 that are used depends upon the setting of the corresponding ASF's.

Related primitives:
POLYLINE
SET POLYLINE INDEX
GDP (spline)
GDP (circular arc 3 point)
GDP (circular arc centre)
GDP (elliptic arc)

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 (IX)

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

Related primitives: SET ASPECT SOURCE FLAGS

Discussion: If the parameter value is out of range the default value is set.

6.2.2.2.3. SET LINE TYPE

Level: 0a

Parameters: Line type (I)
(One of: SOLID, DASHED, DOTTED, DASHED-DOTTED, <other, implementation dependent>)

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.

6.2.2.4. SET LINE WIDTH SCALE FACTOR

Level: 0a

Parameters: Line width scale factor (R)

Description: The line width scale factor is set to the value specified by the parameter.
When the "line width scale factor ASF" is "INDIVIDUAL", subsequent display elements using POLYLINE attributes are displayed with this line width scale factor.
When the "line width scale factor 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 line width scale factor parameter has the value 0.0, the default line width scale factor is set.
If the line width scale factor parameter is negative, 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 are displayed using this polyline colour index.
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 arc centre)
GDP (spline)
GDP (elliptic arc)

Discussion: If the parameter value is out of range the default value is set.

6.2.2.2.6. SET POLYMARKER REPRESENTATION

Level: 1a

Parameters:
— Workstation identifier (ID)
— Polymarker index (IX)
— Marker type (I)
— Marker size scale factor (R)
— 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 ASF's.

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 (IX)

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

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 (I)
(One of: DOT, PLUS, ASTERISK, CIRCLE, DIAGONAL CROSS, <other, implementation dependent>)

Description: The marker type is set to the value specified by the parameter.
When the "marker type ASF" is "INDIVIDUAL", subsequent POLYMARKER elements are displayed with this marker type.
When the "marker 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
× DIAGONAL CROSS

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 scale factor is set to the value specified by the parameter.
When the "marker size scale factor ASF" is "INDIVIDUAL", subsequent POLYMARKER elements are displayed with this marker size scale factor.
When the "marker size scale factor 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 marker size scale factor parameter has the value 0.0, the default marker size scale factor is set.
If the marker size scale factor parameter is negative, 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 using 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 (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 (IX)
{interior style = PATTERN}
— Fill area colour index (CI)

Description: In the fill area bundle table of 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 ASF's.

Related primitives: FILL AREA
SET FILL AREA INDEX
GDP (rectangle)
GDP (circle)
GDP (circular arc 3 point chord)
GDP (circular arc centre chord)
GDP (circular arc 3 point pie)
GDP (circular arc centre pie)
GDP (ellipse)
GDP (elliptic arc chord)
GDP (elliptic arc pie)

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 (IX)

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

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 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 area 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 centre chord)
GDP (circular arc 3 point pie)
GDP (circular arc centre pie)
GDP (ellipse)
GDP (elliptic arc chord)
GDP (elliptic arc pie)
SET FILL AREA STYLE INDEX

Discussion: If the parameter value is out of range the default value is set.

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 centre chord)
GDP (circular arc 3 point pie)
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 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 centre chord)
GDP (circular arc 3 point pie)
GDP (circular arc centre pie)
GDP (ellipse)
GDP (elliptic arc chord)
GDP (elliptic arc pie)
SET FILL AREA INTERIOR STYLE

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 (IX)
— Deltax {DX} (I)
— Deltay {DY} (I)
— Pattern cell colour index list (CL)

Description: The deltax and deltay values define a horizontal by vertical (DX by DY) array into which colour values are mapped.

Related primitives: FILL AREA
GDP (rectangle)
GDP (circle)
GDP (circle arc 3 point chord)
GDP (circular arc centre chord)
GDP (circular arc 3 point pie)
GDP (circular arc centre 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 ★ DY, 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 ★ DY, the exceeding indices are omitted.
If deltax or deltay 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.

Related primitives:
FILL AREA
GDP (rectangle)
GDP (circle)
GDP (circular arc 3 point chord)
GDP (circular arc centre chord)
GDP (circular arc 3 point pie)
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 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 the currently selected interior style is PATTERN, these pattern vectors are used in conjunction with the pattern reference point for displaying filled areas.

Related primitives:
FILL AREA
GDP (rectangle)
GDP (circle)
GDP (circular arc 3 point chord)
GDP (circular arc centre chord)
GDP (circular arc 3 point pie)
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 (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
The text 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 ASF's.

Related primitives:
TEXT
SET TEXT INDEX

Discussion: If one ore 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 (IX)

Description: The text index is set to the value specified by the parameter.

Related primitives: SET ASPECT SOURCE FLAGS

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 (E)
(One of: STRING, CHARACTER, STROKE)

Description: The text font and precision are set to the values specified by the parameters.
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 alignment 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 text string is clipped exactly at the clipping rectangle.

Related primitives: TEXT

Discussion: If a parameter value is out of range the default value of that parameter is 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 display 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 display 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 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 CHARACTER VECTORS

Discussion: If the character expansion factor has the value less than or equal to 0, the default character 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 text 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 (E)
(One of: RIGHT, LEFT, UP, DOWN)

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

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 scaling factor when calculating the width of the character within the character body.
The character expansion factor is a nondimensional 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 (E)
(One of: NORMAL, LEFT, CENTRE, RIGHT)
— Vertical alignment (E)
(One of: NORMAL, TOP, CAP, HALF, BASE, BOTTOM)

Description:
The text alignment is set to the value specified by the parameter. Subsequent display elements using the TEXT attributes are displayed with this text alignment.

Related primitives:
TEXT
SET CHARACTER VECTORS

Discussion:
The "NORMAL" parameter values ar 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

If one of the parameter values is out of range the default value is set.

6.2.2.2.28. SET COLOUR REPRESENTATION

Level: 0a

Parameters:
— Workstation identifier (ID)
— Starting entry in the colour table (CI)
— Colour data list (nCD)

Description:
This primitive is used to load colour tables. 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 parameter (and therefore the number of entries to be loaded n) is implicitly indicated by the receipt of the next primitive to be processed.

Related primitives: SET COLOUR HEADER

Discussion: If the first or second parameters are invalid or out of range the primitive is ignored. If the last colour data are incomplete these last, truncated colour data are ignored.

6.2.2.2.29. SET ASPECT SOURCE FLAGS

Level: 0a

Parameters:
— Line type ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Line width scale factor ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Polyline colour ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Marker type ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Marker size scale factor ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Polymarker colour ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Text font and precision ADF (E)
(One of: INDIVIDUAL, BUNDLED)
— Character expansion factor ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Character spacing ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Text colour ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Fill area interior style ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Fill area style index ASF (E)
(One of: INDIVIDUAL, BUNDLED)
— Fill area colour ASF (E)
(One of: INDIVIDUAL, BUNDLED)

Description: The Aspect Source Flags (ASFs) are set to the values specified by the parameters.

Related primitives: None.

Discussion: If one of the parameter values is out of range the default value is set.

6.2.2.3. Transformation primitives

6.2.2.3.1. SET WORKSTATION WINDOW

Level: 0a

Parameters:
— Workstation identifier (ID)
— Window limits (2P)

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" entry for workstation transformation 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

Discussion: If the workstation identifier parameter is out of range, the primitive is ignored.

6.2.2.3.2. SET WORKSTATION VIEWPORT

Level: 0a

Parameters: — Workstation identifier (ID)
— Viewport limits <XMAX, XMIN <YMAX (4R)

Description: The parameters specify a rectangle in the DC space.
The "requested workstation viewport" entry in the workstation state list of the specified workstation is set to the value specified by the parameter.
If the "dynamic modification accepted" entry for workstation transformation in the workstation description table is set to IMM or if the "display surface empty" entry in the workstation state list is set to EMPTY, then the "current workstation viewport" entry in the workstation state list is set to the value specified by the parameter and the "workstation transformation update state" entry is set to NOTPENDING. Otherwise the "workstation transformation update state" entry in the workstation state list is set to PENDING and the "current workstation viewport" entry is not changed.

Related primitives: SET WORKSTATION WINDOW

Discussion: If the points are outside the display space, the default workstation viewport is set.
If the workstation identifier parameter is out of range the primitive is ignored.

6.2.2.4. Clipping primitives

6.2.2.4.1. SET CLIPPING RECTANGLE

Level: 0a

Parameters: Clipping rectangle limits (2P)

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 (E)
(One of: PERFORM, POSTPONE)

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

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

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

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

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

If the update regeneration flag is PERFORM, UPDATE WORKSTATION suspends the effect of SET DEFERRAL STATE. In that case, it is equivalent to the following sequence of 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 update regeneration flag is POSTPONE, UPDATE WORKSTATION merely initiates the transmission of blocked data. If the value of the "new frame necessary at update" entry is YES and the update regeneration flag is PERFORM, UPDATE WORKSTATION behaves as REDRAW ALL SEGMENTS ON WORKSTATION.
The "new frame action necessary at update" entry in the workstation state list is set to YES during deferred action generation if both of the following are true (see section 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: If the workstation identifier parameter is out of range the primitive is ignored.

6.2.2.5.2. SET DEFERRAL STATE

Level: 1a

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

Description: The deferral mode and implicit regeneration flag of the specified workstation are set to the values specified by the parameters.
Depending on the new value of deferral mode, deferred output may be unblocked.
If the new value of implicit regeneration flag is ALLOWED and the "new frame 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: If the workstation identifier parameter is out of range the primitive is ignored.

6.2.2.5.3. EMERGENCY CLOSE

Level: 0a

Parameters: None.

Description: The graphics configuration 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.

6.2.3. *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 the 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 named segment doesn't exist in the specified workstation, the primitive has no effect.
If the workstation identifier parameter is out of range the primitive is ignored.

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

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 in the workstation state list is PENDING. The entry is set to NOTPENDING.
(d) All visible segments stored on this workstation are redisplayed.

Related primitives: None.

Discussion: If the workstation identifier parameter is out of range the primitive is ignored.

6.2.3.1.7. SET HIGHLIGHTING

Level: 1a

Parameters: — Segment name (ID)
— Highlighting flag (E)
(One of: NOTHIGHLIGHTED, HIGHLIGHTED)

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 (E)
(One of: VISIBLE, INVISIBLE)

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:

$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} M11 & M12 & M13 \\ M21 & M22 & M23 \end{vmatrix} \times \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

The original coordinates are (x, y), the transformed coordinates are (x′, y′), both in NDC.
The values M13 and M23 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 M13 and M23 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 supported" is equal to 0, the implementation allows all values of segment priority to be differentiated.
This feature is intended to address appropriate hardware capabilities only. It cannot be used to force software checking of interference between segments on non-raster displays.

Related primitives: REDRAW ALL SEGMENTS ON WORKSTATION

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 were 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 is ignored.

6.2.3.2.3. INSERT SEGMENT

Level: 2a

Parameters: — Segment name (ID)
— Transformation matrix (M)

Description: Having been transformed, as described below, the primitives contained in the segment are drawn on the display surface and, eventually, stored in the open segment (see also section 5.4.6.).
The coordinates of the primitives contained in the inserted segment are transformed first by any segment transformation specified for it, and secondly by applying the following matrix multiplication to them:

$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} M11 & M12 & M13 \\ M21 & M22 & M23 \end{vmatrix} \times \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

The original coordinates are (x, y), the transformed coordinates are (x', y'), both in NDC.
The values M13 and M23 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 M13 and M23 are ignored.

Other than the segment transformation, attributes of the inserted segment are ignored.
All clipping rectangles in the inserted segment are ignored, each primitive processed is assigned a new clipping rectangle which is the 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 the insertion.

Related primitives: SET SEGMENT TRANSFORMATION

Discussion: None.

6.2.4. *Input primitives*

A Videotex environment will contain no workstations of the type INPUT, so this section is not applicable for Videotex purposes.

6.2.5. *Inquire primitives*

A Videotex environment will support no inquire primitives, so this section is not applicable for Videotex purposes.

6.2.6. *Protocol descriptor primitives*

6.2.6.1. SET DOMAIN RING

Level: 0a

Parameters:
— Angular resolution factor (E)
(One of: RESOLUTION_0, RESOLUTION_1, RESOLUTION_2, RESOLUTION_3)
— Basic Radius (I)

Description: This primitive specifies the precision of the coordinate encoding, when using incremental mode.

Related primitives: SET COORDINATE PRECISION

Discussion: If the Basic Radius parameter value is 0, the default Basic Radius according to the current granularity code (as set with the SET COORDINATE PRECISION primitive) is selected (see clause 9).
If the Basic Radius parameter is negative the primitive is ignored.
If the angular resolution factor parameter value is out of range the default value is set.

6.2.6.2. SET COLOUR HEADER

Level: 0a

Parameters:
— Unit resolution (1,2..9) (I)
— Coding method (E)
(Only: RGB_CODING)

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 (E)
(One of: ALLOWED, FORBIDDEN)

Description: The magnitude code parameter specifies 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 values 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 −11111111111.1111111111111111 (binary) to +11111111111.1111111111111111 (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 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 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 (E)
(One of: ALLOWED, FORBIDDEN)

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 specifies 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 whether 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 −11111111111.1111111111111111 (binary) to +11111111111.1111111111111111 (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 colour indices.
Related primitives:	None.
Discussion:	None.

7. ENCODING PRINCIPLES

The encoding of the geometric display is independent of the encoding of other sets, e.g. alphamosaic or photographic display. The geometric display in general is selected by means of the appropriate US sequence, as described in part 0 of this document. The specification of different levels within the geometric display is described in Appendix A2D.

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.

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	1	0	1	0	1
b4	b3	b2	b1		0	1	2	3	4	5	6	7
0	0	0	0	0								
0	0	0	1	1								
0	0	1	0	2								
0	0	1	1	3								
0	1	0	0	4								
0	1	0	1	5								
0	1	1	0	6								
0	1	1	1	7								
1	0	0	0	8								
1	0	0	1	9								
1	0	1	0	10								
1	0	1	1	11								
1	1	0	0	12								
1	1	0	1	13								
1	1	1	0	14								
1	1	1	1	15								

reserved for control functions — opcodes — operands

primitives

Table A2-5 (T/TE 06-02). 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 Figure A2-15 (T/TE 06-02).

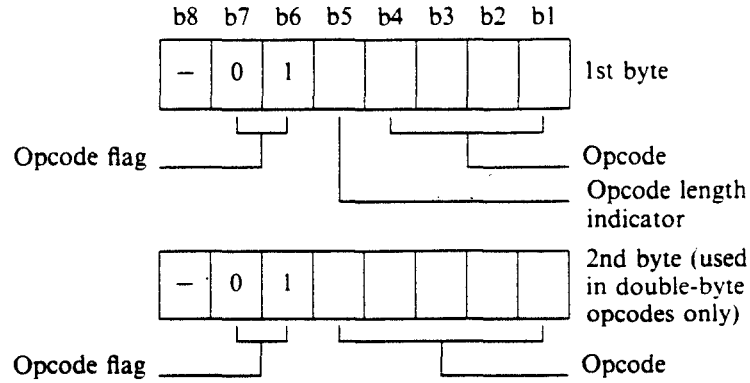


Figure A2-15 (T/TE 06-02). Opcode encoding structure.

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);

— $15 \times 32 = 480$ 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 selects the basic set.
n = 1 selects the first extension opcode set.
n = N selects 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 Figure A2-16 (T/TE 06-02).

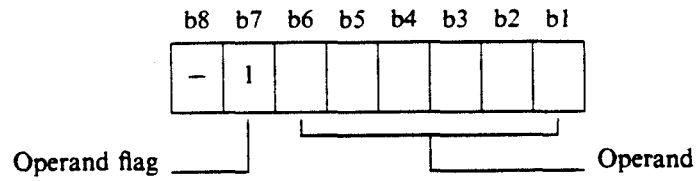


Figure A2-16 (T/TE 06-02). Operand encoding structure.

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 index list (CL)

The encode the data types 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 A2-17 (T/TE 06-02):

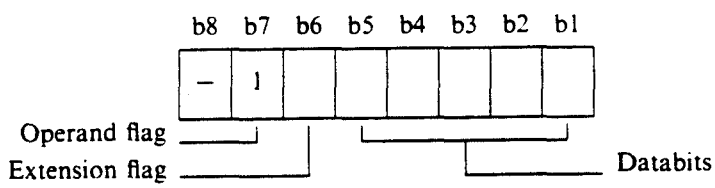


Figure A2-17 (T/TE 06-02). Basic format Structure.

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.

Signed integers are encoded using the modulus and sign notation. The range of integer numbers is subdivided into a non-negative range and a negative range, using bit b5 of the first byte 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.). Bits b4 to b1 of the first byte and bits b5 to b1 of subsequent bytes are used to encode the modulus of the integer.

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 datatype Enumerated type is encoded in the same manner as an Integer. The datatypes Identifier, Colour index and Index are encoded in the Basic format without a sign bit.

In the next figures some examples are given.

b8	b7	b6	b5	b4	b3	b2	b1
–	1	0	1	1	1	1	1

Parameter: Segment name
Datatype: Identifier
Operand value: 31

Figure A2-18 (T/TE 06-02). Identifier encoding.

b8	b7	b6	b5	b4	b3	b2	b1
–	1	0	0	0	0	0	1

Parameter: Polyline colour ASF
Datatype: Enumerated type
Operand value: BUNDLED

Figure A2-19 (T/TE 06-02). Enumerated type encoding.

b8	b7	b6	b5	b4	b3	b2	b1
–	1	0	1	0	0	0	0

Parameter: Polyline colour index
Datatype: Colour index
Operand value: 16

Figure A2-20 (T/TE 06-02). Colour index encoding.

b8	b7	b6	b5	b4	b3	b2	b1
–	1	0	0	0	0	1	0

Parameter: Line type
Datatype: Integer
Operand value: +2 (standardized line type 2)

Figure A2-21 (T/TE 06-02). Integer encoding for standardized use.

b8	b7	b6	b5	b4	b3	b2	b1
–	1	0	1	0	0	1	0

Parameter: Line type
Datatype: Integer
Operand value: −2 (private line type 2)

Figure A2-22 (T/TE 06-02). Integer encoding for private use.

b8	b7	b6	b5	b4	b3	b2	b1
–	1	1	0	0	0	1	0
–	1	1	0	0	0	0	0
–	1	0	1	1	1	1	1

Parameter: Segment name
Datatype: Identifier
Operand value: 2079

Figure A2-23 (T/TE 06-02). Identifier encoding (more than one byte).

b8	b7	b6	b5	b4	b3	b2	b1
–	1	1	0	0	0	0	1
–	1	0	0	0	0	0	0

Parameter: Bundle index
Datatype: Index
Operand value: 32

Figure A2-24 (T/TE 06-02). Index encoding (more than one byte).

b8	b7	b6	b5	b4	b3	b2	b1
–	1	1	1	1	0	0	0
–	1	0	0	0	0	0	0

Parameter: Marker type
Datatype: Integer
Operand value: −256 (private marker type 256)

Figure A2-25 (T/TE 06-02). 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 or the 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 A2-26 (T/TE 06-02) and A2-27 (T/TE 06-02) show the general format of a mantissa. Figure A2-28 (T/TE 06-02) shows an example of a multiple-byte mantissa.

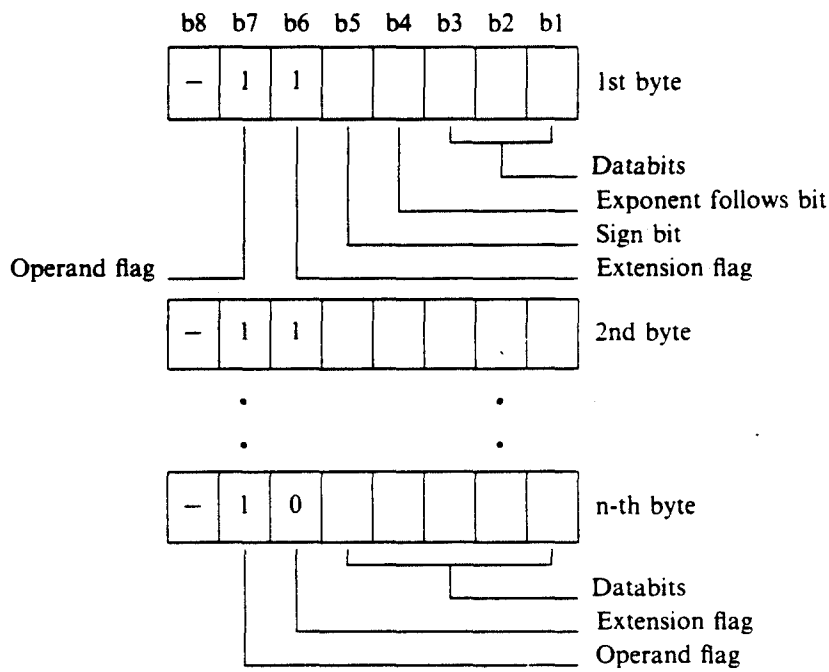


Figure A2-26 (T/TE 06-02). Encoding of a mantissa using the Basic format.

Explicit exponent allowed = ALLOWED

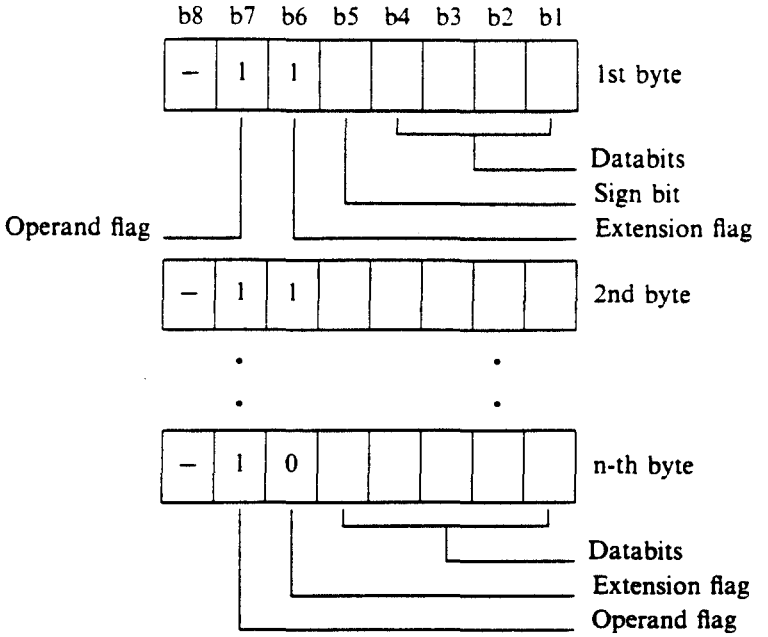


Figure A2-27 (T/TE 06-02). Encoding of a mantissa using the Basic format.
Explicit exponent allowed = FORBIDDEN

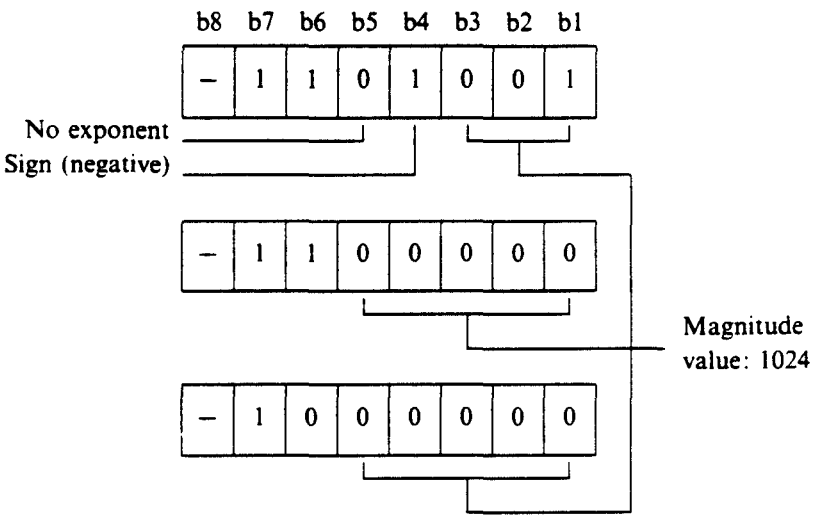


Figure A2-28 (T/TE 06-02). A multiple-byte mantissa.
Mantissa value = −1024

7.2.2.2. Exponent

If the current "explicit exponent allowed" value is ALLOWED and the "exponent follows" bit in a mantissa is ONE, then an explicit exponent follows 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 datatype 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 individual points (points not part of a point list) 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 may be coded with either one or two coding structures or a mixture of the two. These structures are called "Displacement Mode" and "Incremental Mode". Individual points (datatype P) and the first point of each point list, 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.

<P> = <V: delta x> <V: delta y>

For individual points (datatype P) and for the first point of each point list, the displacement values delta x and delta y are displacements measured from the origin. For points after the first point in a 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 whose 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:

$$N = \frac{8R}{2^p} \text{ with } p = 0, 1, 2, 3$$

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 A2-29 (T/TE 06-02) shows a Ring with Radius $R = 3$ and Angular resolution factor $p = 0$ respectively 1.

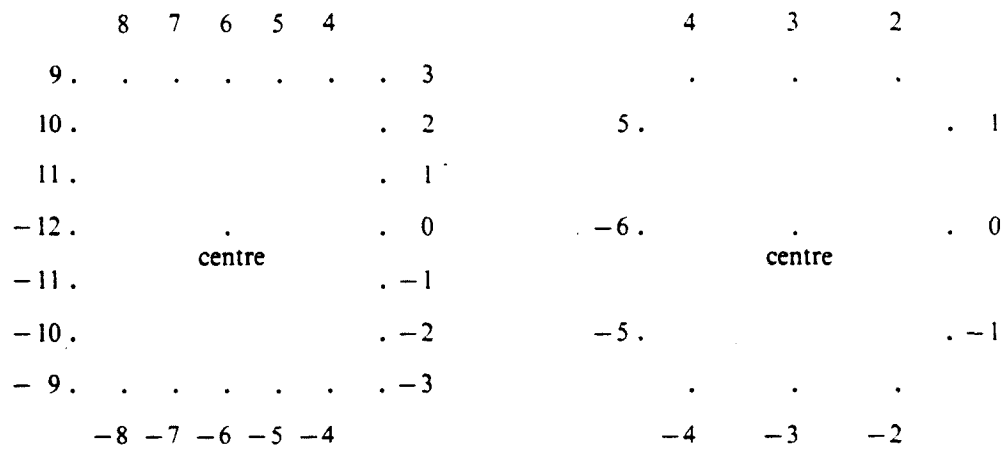


Figure A2-29 (T/TE 06-02). Some example Rings with point numbering.

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 A2-30 (T/TE 06-02), 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 A2-29 (T/TE 06-02). The Direction (position of the point with number ZERO) is identified by D.

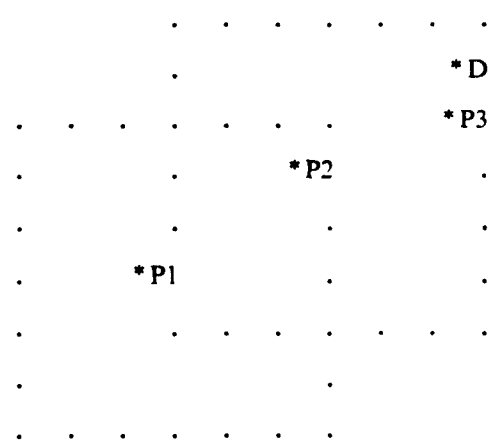


Figure A2-30 (T/TE 06-02). Change of direction with R = 3.

The basic Radius is expressed in Basic Grid Units (BGUs). 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 default basic Radius follows from:

$$\text{default basic Radius} = 2^{\max(0,\ -8\text{-granularity code})}$$

The basic Radius as derived from the granularity code may be changed to any value greater than ZERO 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 basic Radius must be equal to or greater than ONE. If the basic Radius is less than ONE, the value of the default basic Radius is assumed for the basic Radius.

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 basic Radius and the Angular resolution factor temporarily. The Radius actually used (= Rt) can have a value of R, 2R, 4R or 8R, where R is the basic Radius as set before entering the Incremental mode. The Angular resolution factor actually used (= pt) 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 table is given in Table A2-6 (T/TE 06-02).

Length	Code-word	Point number
2	0 0	0
2	1 0	1
2	0 1	−1
4	1 1 0 0	2
4	1 1 0 1	−2
6	1 1 1 0 0 0	3
6	1 1 1 0 0 1	−3
6	1 1 1 0 1 0	4
6	1 1 1 0 1 1	−4
8	1 1 1 1 0 0 0 0	5
8	1 1 1 1 0 0 0 1	−5
8	1 1 1 1 0 0 1 0	6
8	1 1 1 1 0 0 1 1	−6
8	1 1 1 1 0 1 0 0	7
8	1 1 1 1 0 1 0 1	−7
8	1 1 1 1 0 1 1 0	8
8	1 1 1 1 0 1 1 1	−8
10	1 1 1 1 1 0 0 0 0 0	9
10	1 1 1 1 1 0 0 0 0 1	−9
10	1 1 1 1 1 0 0 0 1 0	10
10	1 1 1 1 1 0 0 0 1 1	−10
10	1 1 1 1 1 0 0 1 0 0	11
10	1 1 1 1 1 0 0 1 0 1	−11
10	1 1 1 1 1 0 0 1 1 0	12
10	1 1 1 1 1 0 0 1 1 1	−12
10	1 1 1 1 1 0 1 0 0 0	13
10	1 1 1 1 1 0 1 0 0 1	−13
10	1 1 1 1 1 0 1 0 1 0	14
10	1 1 1 1 1 0 1 0 1 1	−14
10	1 1 1 1 1 0 1 1 0 0	15
10	1 1 1 1 1 0 1 1 0 1	−15
10	1 1 1 1 1 0 1 1 1 0	16
10	1 1 1 1 1 0 1 1 1 1	−16
10	1 1 1 1 1 1 0 0 0 0	17
10	1 1 1 1 1 1 0 0 0 1	−17
10	1 1 1 1 1 1 0 0 1 0	18
10	1 1 1 1 1 1 0 0 1 1	−18
10	1 1 1 1 1 1 0 1 0 0	19
10	1 1 1 1 1 1 0 1 0 1	−19
10	1 1 1 1 1 1 0 1 1 0	C1
10	1 1 1 1 1 1 0 1 1 1	−20
10	1 1 1 1 1 1 1 1 0 0 0	C2
10	1 1 1 1 1 1 1 1 0 0 1	C3
10	1 1 1 1 1 1 1 1 0 1 0	C4
10	1 1 1 1 1 1 1 1 0 1 1	C5
10	1 1 1 1 1 1 1 1 1 0 0	C6
10	1 1 1 1 1 1 1 1 1 0 1	IM-ESC 1
10	1 1 1 1 1 1 1 1 1 1 0	IM-ESC 2
10	1 1 1 1 1 1 1 1 1 1 1	End of Block

Table A2-6 (T/TE 06-02). 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 to 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 temporally the parameters that define the Ring to be used to Rt and pt. The values of Rt are taken from the range R, 2R, 4R and 8R, where R is the value of the Ring Radius before entering the Incremental Mode. The values of pt are taken from the range 0, 1, 2 and 3. The function of these codes is as follows:

(a) C1
Change the Ring parameters, Rt and pt, to the next higher value e.g. if the Rt = R, the next higher is 2R, if pt = 0 the next higher value is 1. If Rt = 8R or pt = 3 the code C1 has no effect.

(b) C2
Change the Ring parameters, Rt and pt, to the next lower value. The effect of the code C2 is the inverse of code C1. If Rt = R or pt = 0, the code C2 has no effect.

(c) C3
Change the Rt to the next higher value. The code C3 has no effect if the current basic Radius Rt = 8R.

(d) C4
Change the Angular resolution factor pt to the next higher value. The code C4 has no effect if the current pt = 3.

(e) C5
Change the Ring Radius Rt to the next lower value. The code C5 has no effect if the current basic Radius = R.

(f) C6
Change the Angular resolution factor pt to the next lower value. The code C6 has no effect if the current pt = 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 centre point of the Ring. Changing the Ring parameters from R to Rt and from p to pt by means of these codes will have effect up to the next Incremental mode End of Block code. The basic Radius R and the Angular resolution factor p are not affected. This implies that the Ring parameters of subsequent Incremental mode point lists are not affected by these codes.

7.2.2.3.3. Incremental mode encoding

The structure of the Incremental mode format is given in Figure A2-31 (T/TE 06-02):

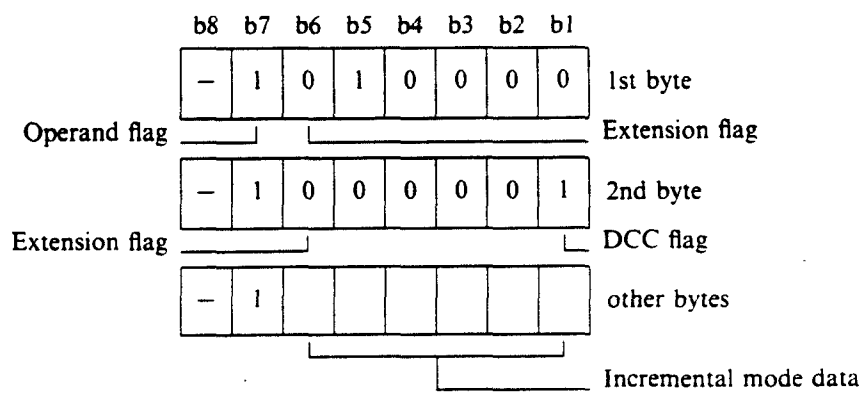


Figure A2-31 (T/TE 06-02). Incremental mode structure.

The first byte is 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 now is set to ONE to identify the use of Differential 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 Figure A2-32 (T/TE 06-02).

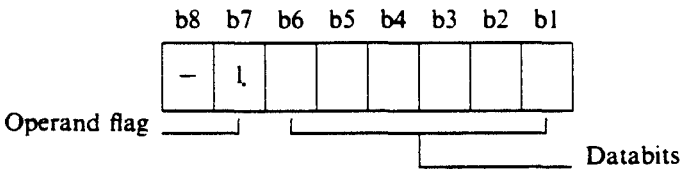


Figure A2-32 (T/TE 06-02). Bitstream format structure.

The Bitstream format is used to encode:
— Incremental mode coordinates (see section 7.2.2.3.3.);
— Colour index lists (see section 7.2.3.1.);
— Colour direct (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 Figure A2-33 (T/TE 06-02)). The first byte contains the most significant bits. In the last byte, bit b3, b2 and b1 may be left unused.

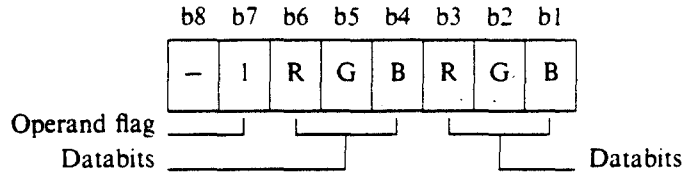


Figure A2-33 (T/TE 06-02). Colour direct encoding.

Figure A2-34 (T/TE 06-02) gives an example of the colour loading with a unit resolution of 4.

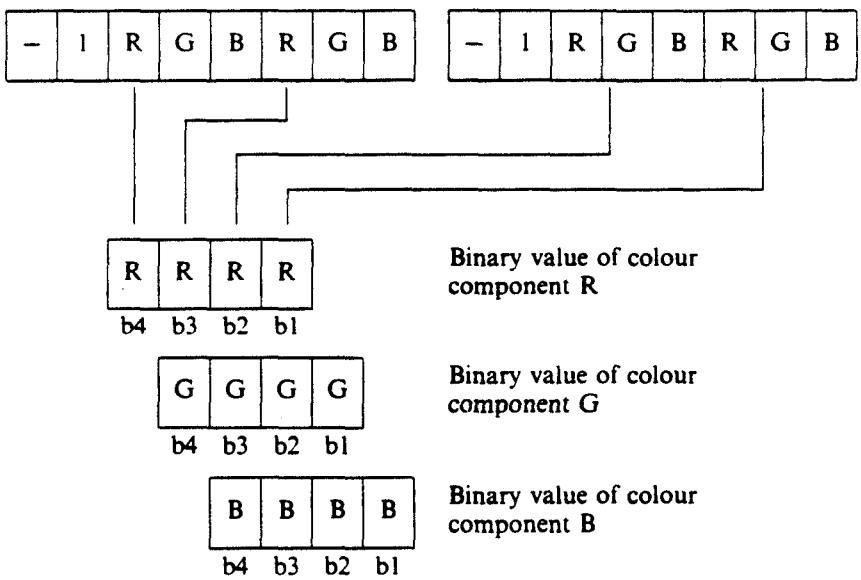


Figure A2-34 (T/TE 06-02). Colour loading with a unit resolution of 4.

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:

<colour list> = <colour list element> n = 1, 2, ...

n times

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 (run count).

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:

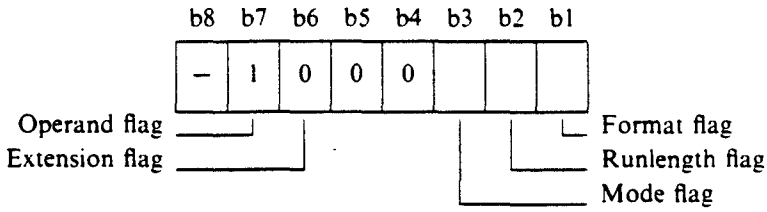


Figure A2-35 (T TE 06-02). First byte of Colour list encoding.

The mode flag, bit 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.

Bits 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

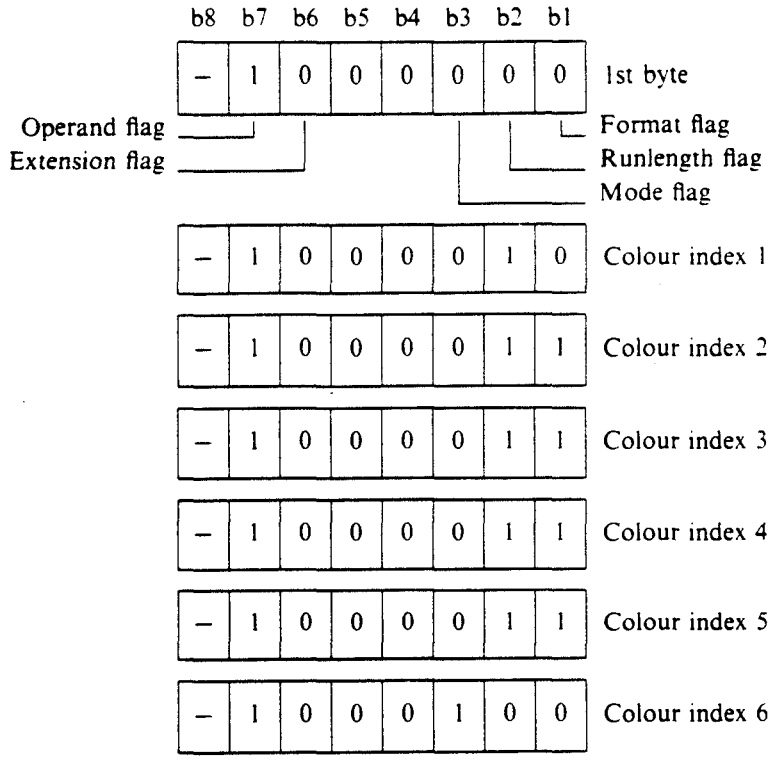
This encoding method specifies a colour list by individually encoding all colour list elements as colour indices.

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.

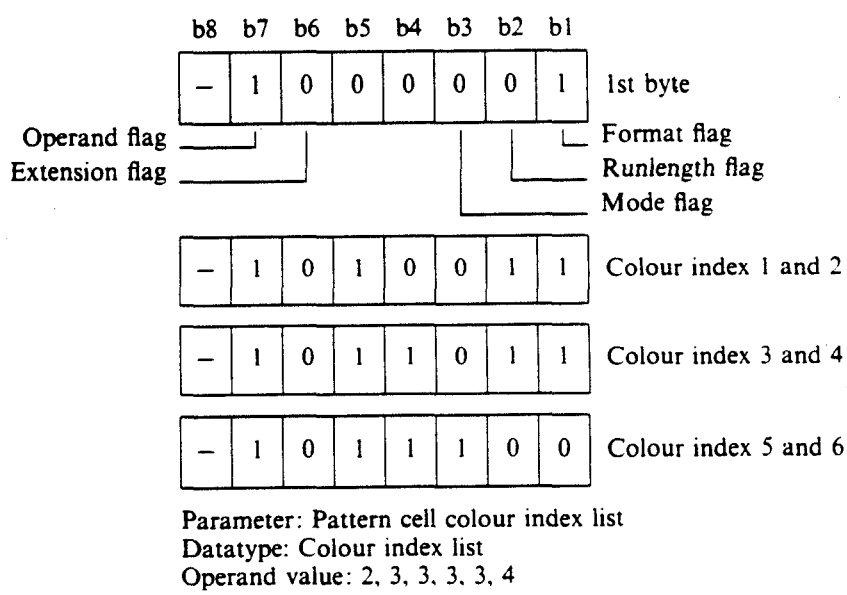
If Bitstream format encoding is specified, bit b3 of the first byte set to ONE, the indices of the Colour index list are encoded using the Bitstream format described in section 7.2.3. The length (in bits) of each index is identified by the SET COLOUR INDEX PRECISION primitive. These indices are packed 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 A2-36 (T/TE 06-02) and A2-37 (T/TE 06-02).



Parameter: Pattern cell colour index list
Datatype: Colour index list
Operand value: 2, 3, 3, 3, 3, 4

Figure A2-36 (T/TE 06-02). Colour index list individually encoded using the Basic format.



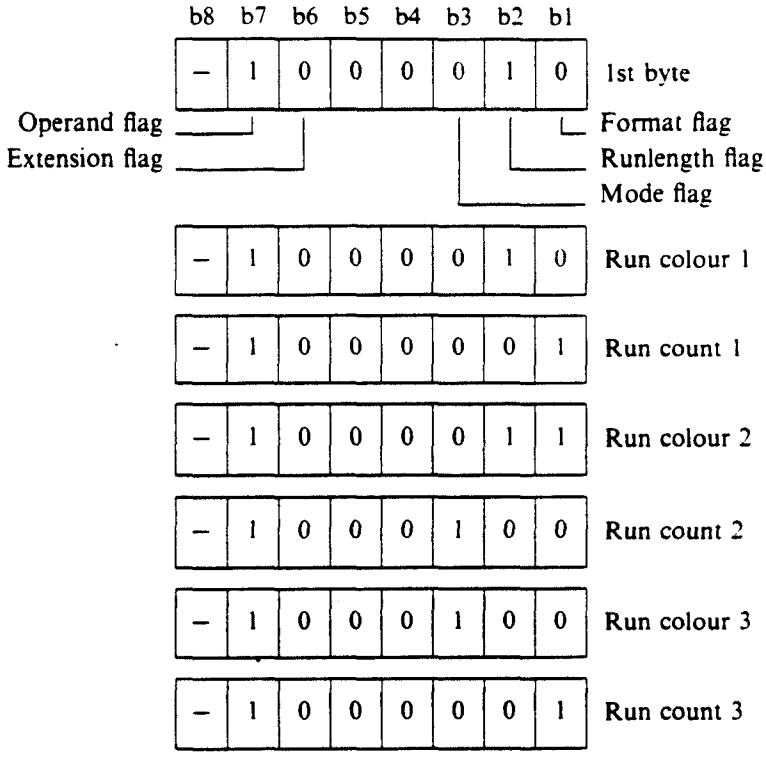
Parameter: Pattern cell colour index list
Datatype: Colour index list
Operand value: 2, 3, 3, 3, 3, 4

Figure A2-37 (T/TE 06-02). 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 the 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 A2-38 (T.TE 06-02).



Parameter: Pattern cell colour index list
Datatype: Colour index list
Operand value: 2, 3, 3, 3, 3, 4

Figure A2-38 (T/TE 06-02). 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 alway 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 A2-39 (T/TE 06-02) and A2-40 (T/TE 06-02).

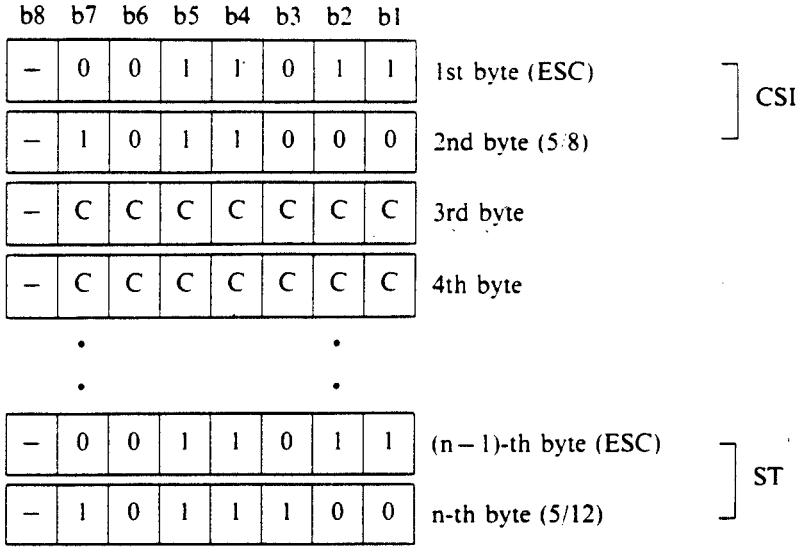


Figure A2-39 (T/TE 06-02). String format structure in a 7-bit environment. Databits are marked C.

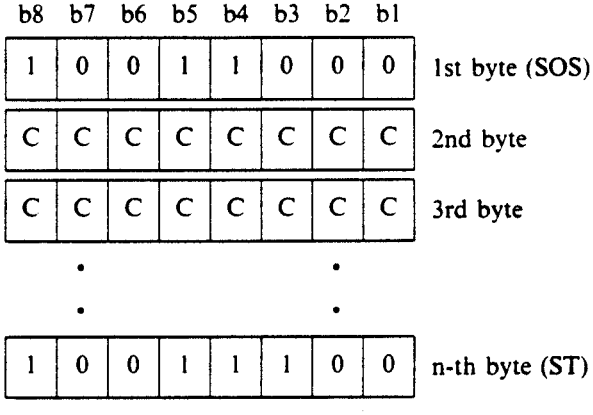


Figure A2-40 (T/TE 06-02). String format structure in an 8-bit environment. Databits are marked C.

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 an 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 the 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 opcode for each primitive.

Primitives	Opcode Byte 1	Opcode Byte 2
WORKSTATION MANAGEMENT PRIMITIVES		
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
OUTPUT DRAWING PRIMITIVES		
POLYLINE	2/0	—
POLYMARKER	2/2	—
FILL AREA	2/1	—
TEXT	2/4	—
CELL ARRAY	2/3	—
GDP	2/5	—
OUTPUT PRIMITIVES RELATED TO DISPLAY ELEMENT 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
TRANSFORMATION PRIMITIVES		
SET WORKSTATION WINDOW	3/2	2/1
SET WORKSTATION VIEWPORT	3/2	2/2

Primitives	Opcode Octet 1	Opcode Octet 2
CONTROL PRIMITIVES		
UPDATE WORKSTATION	3 2	3 4
SET DEFERRAL STATE	3 2	3 5
EMERGENCY CLOSE	3 2	2 6
SEGMENT RELATED PRIMITIVES		
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
ASSOCIATE SEGMENT WITH WORKSTATION	3 3	2 10
COPY SEGMENT TO WORKSTATION	3 3	2 11
INSERT SEGMENT	3 3	2 4
PROTOCOL DESCRIPTOR PRIMITIVES		
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 occurrence
<symbols> (n)	= n or more occurrences (with n > = 1)
<symbols> (= n)	= n occurrences (with n > = 1)
<symbols> (0)	= optional, 0 or more occurrences
[comment]	= explanation of a production
<x : y>	= construction x with meaning y
<symbols> | <symbols>	= choice

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. GDS ESCAPE1

<GDS ESCAPE1 opcode: 3 2, 2 11>
<integer: GDS escape identifier>
<record: GDS data record>
with
<integer: escape identifier> =
<integer: non-negative> [reserved]
<integer: negative> [private escape]

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: number of cells in PQ-direction>
[value M]
<integer: number of cells in QR-direction>
[value N]
<colour index list: cell colour index list>

8.2.2.1.6. GDP

<GDP opcode: 2/5>
<integer: GDP identifier>
with
<integer: GDP identifier> =
<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]
|<integer: negative> [private GDP]
if <integer: GDP identifier> = 0 [rectangle]
then
<point: two points> (=2)
if <integer: GDP identifier> = 1 [circle]

then
<point: centre>
<size value: radius>
if <integer: GDP identifier> = 2 [circular arc 3 point]
then
<point: starting point>
<point: intermediate point>
<point: ending point>
if <integer: GDP identifier> = 3 [circular arc 3 point chord]
then
<point: starting point>
<point: intermediate point>
<point: ending point>
if <integer: GDP identifier> = 4 [circular arc 3 point pie]
then
<point: starting point>
<point: intermediate point>
<point: ending point>
if <integer: GDP identifier> = 5 [circular arc centre]
then
<point: centre>
<size value: radius>
<point: start vector>
<point: end vector>
if <integer: GDP identifier> = 6 [circular arc centre chord]
then
<point: centre>
<size value: radius>
<point: start vector>
<point: end vector>
if <integer: GDP identifier> = 7 [circular arc centre pie]
then
<point: centre>
<size value: radius>
<point: start vector>
<point: end vector>
if <integer: GDP identifier> = 8 [ellipse]
then
<point: centre point>
<point: endpoints> (= 2) [(X1, Y1) and (X2, Y2)]
if <integer: GDP identifier> = 9 [elliptic arc]
then
<point: centre point>
<point: endpoints> (= 2) [(X1, Y1) and (X2, Y2)]
<real: T-start, T-end> (= 2)
if <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)
if <integer: GDP identifier> = 11 [elliptic arc pie]
<then
<point: centre points>
<point: endpoints (= 2) [(X1, Y1) and (X2, Y2)]
<real: T-start, T-end (= 2)
if <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>
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]

8.2.2.2.4. 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>
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]

8.2.2.2.9. 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 POLYMARKER 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]
|<index: 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]
|<integer: fill area style index> =
[interior style HATCH]
<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>
with
<enumerated: 0> [HOLLOW]
|<enumerated: 1> [SOLID]
|<enumerated: 2> [PATTERN]
|<enumerated: 3> [HATCH]
|<enumerated: >3> [reserved]

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: 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]

8.2.2.2.16. 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 list: pattern cell colour index list>

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 0]
|<integer: >0> [reserved]
|<integer: negative value> [private font]

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>
<enumerated: text precision>
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 0]
|<integer: >0> [reserved]
|<integer: negative value> [private font]

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 path> =
<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 the 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: <attribute> ASF> =
<enumerated: 0> [INDIVIDUAL]
|<enumerated: 1> [BUNDLED]
|<enumerated: >1> [reserved]

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)

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]
|<enumerated: >1> [reserved]

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>

8.2.3. *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: 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> [NOTHIGHLIGHTED]
|<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>

8.2.3.2. 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*

A Videotex environment will contain no workstations of the type INPUT, so this section is not applicable for Videotex purposes.

8.2.5. *Inquire primitives*

A Videotex environment will support no inquire primitives, so this section is not applicable for Videotex purposes.

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>
<integer: basic Radius>
with
<enumerated: angular resolution factor> =
<enumerated: 0> [resolution 0]
|<enumerated: 1> [resolution 1]
|<enumerated: 2> [resolution 2]
|<enumerated: 3> [resolution 3]
|<enumerated: >3> [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>
with
<enumerated: 0> [ALLOWED]
|<enumerated: 1> [FORBIDDEN]

8.2.6.4. SET REAL PRECISION

<SET REAL PRECISION opcode: 3/2, 3/9>
<integer: magnitude code>
<integer: granularity code>
<integer: default exponent>
with
<enumerated: 0> [ALLOWED]
|<enumerated: 1> [FORBIDDEN]

8.2.6.5. SET COLOUR INDEX PRECISION

<SET COLOUR INDEX PRECISION opcode: 3,2, 2,10>
<integer: 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 able to process primitives even if some parameter values are not yet defined.

Parameter	*Value*
ASPECT SOURCE FLAGS (ALL)	INDIVIDUAL
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
DOMAIN RING	
Angular resolution factor	RESOLUTION_0
Basic Radius	2**(0, −8-current granularity code)
FILL AREA BUNDLE TABLE	The entry of the FILL AREA BUNDLE TABLE corresponding to FILL AREA INDEX 1 contains the default 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	HOLLOW
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 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 INDEX 1 contains the default values of LINE TYPE, LINE WIDTH SCALE FACTOR and POLYLINE COLOUR INDEX. Non defined entries use the attributes as specified by POLYLINE INDEX 1.
POLYLINE COLOUR INDEX	1
POLYLINE INDEX	1

Parameter	*Value*
POLYMARKER BUNDLE TABLE	The entry of the POLYMARKER BUNDLE TABLE corresponding to POLYMARKER INDEX 1 contains the default values of MARKER TYPE, MARKER SIZE SCALE FACTOR and POLYMARKER COLOUR INDEX. Non defined 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 defined 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	1.0 0.0 0.0 0.0 1.0 0.0
UNIT RESOLUTION	4
UPDATE REGENERATION FLAG	POSTPONE
VISIBILITY FLAG	VISIBLE
WORKSTATION VIEWPORT*	The largest square which fits into the display area (see Figure A2-9 (T/CD 06-01)).
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 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 5.10 of GKS DIS 7942).

To be in conformance to one level, for a device, means to offer at least all the functionalities pertaining to this level.

* The default WORKSTATION WINDOW and WORKSTATION VIEWPORT set by the primitive SET DEFAULTS shall affect only the requested entries in the workstation state lists of all open workstations.

Appendix A2A

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 primitive is indicated:
— the applicability to workstation categories,
— the level in which it appears,
— the state (mandatory or optional).

The following notational conventions are used:
— SS: workstation independent segment storage,
— O: workstation of category OUTPUT,
— 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	M
CLOSE WORKSTATION	L0a	SS	O	M
ACTIVATE WORKSTATION	L0a	SS	O	M
DEACTIVATE WORKSTATION	L0a	SS	O	M
CLEAR WORKSTATION	L0a	SS	O	M
SET DEFAULTS	L0a	SS	O	M
GDS ESCAPE 1	L0a	SS	O	M
POLYLINE	L0a	SS	O	M
POLYMARKER	L0a	SS	O	M
TEXT	L0a	SS	O	M
FILL AREA	L0a	SS	O	M
CELL ARRAY	L0a	SS	O	M
GENERALIZED DRAWING PRIMITIVE:	L0a	SS	O	M
RECTANGLE	L0a	SS	O	N
CIRCLE	L0a	SS	O	N
CIRCULAR ARC 3 POINT	L0a	SS	O	N
CIRCULAR ARC 3 POINT CHORD	L0a	SS	O	N
CIRCULAR ARC 3 POINT PIE	L0a	SS	O	N
CIRCULAR ARC CENTRE	L0a	SS	O	N
CIRCULAR ARC CENTRE CHORD	L0a	SS	O	N
CIRCULAR ARC CENTRE PIE	L0a	SS	O	N
ELLIPSE	L0a	SS	O	N
ELLIPTIC ARC	L0a	SS	O	N
ELLIPTIC ARC CHORD	L0a	SS	O	N
ELLIPTIC ARC PIE	L0a	SS	O	N
SPLINE	L0a	SS	O	N
SET POLYLINE INDEX	L0a	SS	O	M
SET POLYLINE REPRESENTATION	L1a		O	M
SET LINE TYPE	L0a	SS	O	M
SET LINE WIDTH SCALE FACTOR	L0a	SS	O	M
SET POLYLINE COLOUR INDEX	L0a	SS	O	M
SET POLYMARKER INDEX	L0a	SS	O	M
SET POLYMARKER REPRESENTATION	L1a		O	M
SET MARKER TYPE	L0a	SS	O	M
SET MARKER SIZE SCALE FACTOR	L0a	SS	O	M
SET POLYMARKER COLOUR INDEX	L0a	SS	O	M
SET FILL AREA INDEX	L0a	SS	O	M
SET FILL AREA REPRESENTATION	L1a		O	M

PRIMITIVE	LEVEL	APPLIES	TO	OPTION
SET FILL AREA INTERIOR STYLE	L0a	SS	O	M
SET FILL AREA STYLE INDEX	L0a	SS	O	M
SET FILL AREA COLOUR INDEX	L0a	SS	O	M
SET PATTERN REPRESENTATION	L1a		O	N
SET PATTERN REFERENCE POINT	L0a	SS	O	N
SET PATTERN VECTORS	L0a	SS	O	N
SET TEXT INDEX	L0a	SS	O	M
SET TEXT REPRESENTATION	L1a		O	M
SET TEXT FONT AND PRECISION	L0a	SS	O	M
SET CHARACTER EXPANSION FACTOR	L0a	SS	O	M

Appendix A2B

B-SPLINE CURVES AND ELLIPSES

A2B.1. SHORT NOTE ON B-SPLINE CURVES

A spline is a piecewise polynomial function passing through a set of points called knots (see Figure A2B-1 (T TE 06-02).

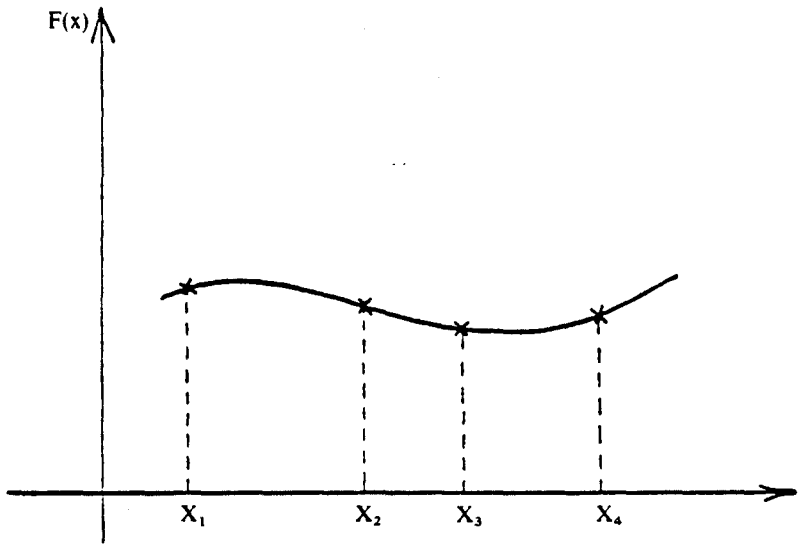


Figure A2B-1 (T/TE 06-02). A spline passing through 4 knots.

The values X1, X2, ..., Xn 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 (Xi, Xi + 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:

$$P(t) = \sum_{i=1}^{n} P_i N_i m(t)$$

where P(t) is a point on the curve, points Pi are called guiding points and $N_i m$ is a m-degree B-spline.

For a uniform quadratic B-spline, between two knots, we have:

$$P(t) = \left(\frac{P_{i+2} + P_i}{2} - P_{i+1}\right)t^2 + \left(P_{i+1} - P_i\right)t + \frac{1}{2}\left(P_{i+1} + P_i\right)$$

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 A2B-2 (T/TE 06-02). Knots are located on the middle of the segments joining the breakpoints and the curve is tangent to the segment at this point.

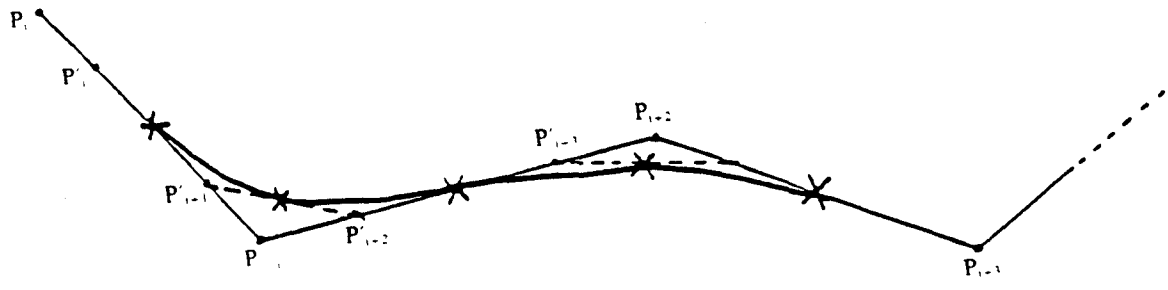


Figure A2B-2 (T TE 06-02). Uniform quadratic B-spline curve.

Such spline curves can easily be generated using the sub-division or refinement properties of B-splines [1].

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:

$$P'_i = \frac{1}{2}\left(P_i + \frac{P_i + P_{i-1}}{2}\right) \qquad P'_{i+2} = \frac{1}{2}\left(P_{i+1} + \frac{P_{i+1} + P_{i+2}}{2}\right)$$

$$P'_{i+1} = \frac{1}{2}\left(P_{i-1} + \frac{P_{i+1} + P_i}{2}\right) \qquad P'_{i+3} = \frac{1}{2}\left(P_{i+2} + \frac{P_{i+1} + P_{i+2}}{2}\right)$$

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 A2B-2 (T/TE 06-02)).

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 [2]. Note that in this algorithm, the given endpoints 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.

[1] LANE J.M., RIESENFELD R.F., «A theoretical Development for the Computer Generation and Display of Piecewise Polynomial Surfaces», IEEE Trans. on PAMI — vol. PAMI-2, No. 1 (January 1980), pp. 35-46.

[2] CHAIKIN G.M., «An Algorithm for High-Speed Curve Generation», Computer Graphics and Image Processing, 1974 — vol. 3 — pp. 346-349.

A2B.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 A2B-3 (T/TE 06-02) below, these axes do not remain mutually orthogonal (do not remain axes) across a scaling transformation which does not preserve aspect ratio.

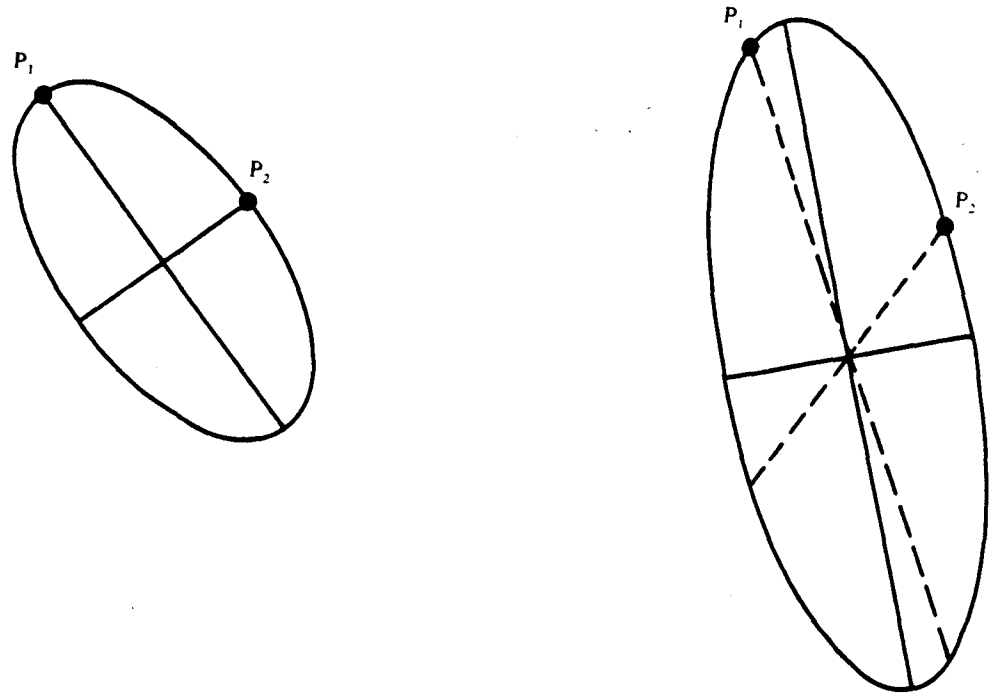


Figure A2B-3 (T/TE 06-02). The scaling of an ellipse and its axes such that $X' = X$ and $Y' = 2Y$.

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 A2B-4 (T/TE 06-02) in which the ellipse has been scaled by a factor of two in the Y-direction only.

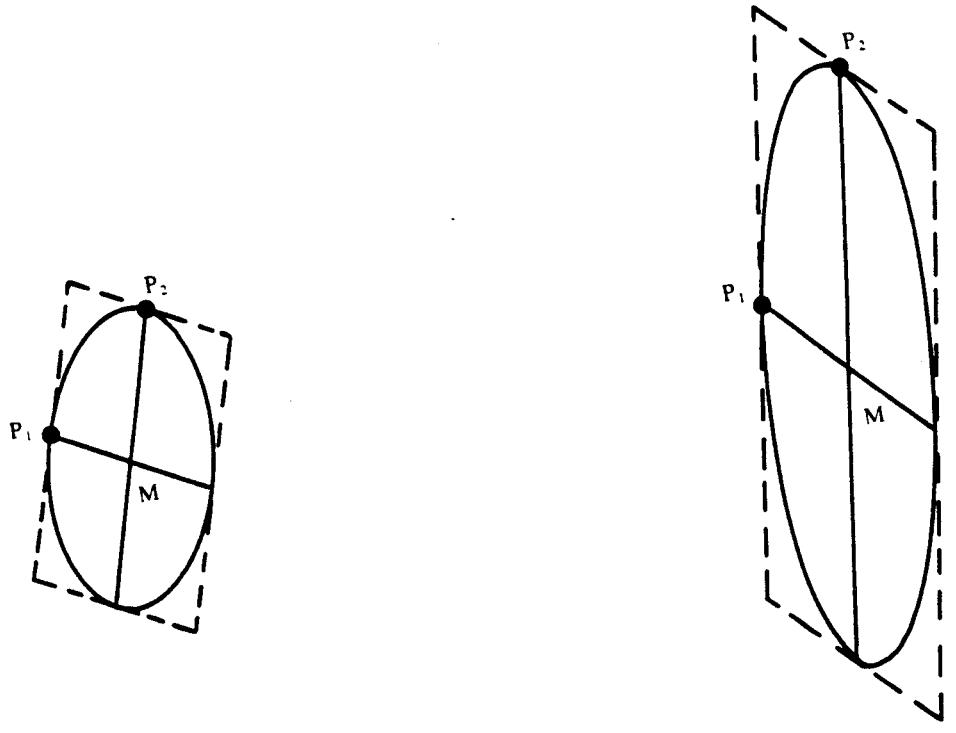


Figure A2B-4 (T/TE 06-02). Ellipses defined by a Conjugate Diameter Pair.

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 are 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.

Appendix A2C

CROSS REFERENCES

A2C.1. FUNCTIONAL ORDER

Primitives	Chapter number	
	Description	Encoding
Workstation management primitives		
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 ESCAPE1	6.2.1.7.	8.2.1.7.
Output drawing primitives		
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.
GDP	6.2.2.1.6.	8.2.2.1.6.
Output primitives related to display element 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.
Transformation primitives		
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.

Primitives	Chapter number Description	Encoding
Clipping primitives		
SET CLIPPING RECTANGLE	6.2.2.4.1.	8.2.2.4.1.
Control primitives		
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.
Segment related primitives		
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.
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.
Protocol descriptor primitives		
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.

A2C.2. ALPHABETIC ORDER

Primitives	Chapter number	
	Description	Encoding
ACTIVATE WORKSTATION	6.2.1.3.	8.2.1.3.
ASSOCIATE SEGMENT WITH WORKSTATION	6.2.3.2.1.	8.2.3.2.1.
CELL ARRAY	6.2.2.1.5.	8.2.2.1.5.
CLEAR WORKSTATION	6.2.1.5.	8.2.1.5.
CLOSE SEGMENT	6.2.3.1.2.	8.2.3.1.2.
CLOSE WORKSTATION	6.2.1.2.	8.2.1.2.
COPY SEGMENT TO WORKSTATION	6.2.3.2.2.	8.2.3.2.2.
CREATE SEGMENT	6.2.3.1.1.	8.2.3.1.1.
DEACTIVATE WORKSTATION	6.2.1.4.	8.2.1.4.
DELETE SEGMENT	6.2.3.1.5.	8.2.3.1.5.
DELETE SEGMENT FROM WORKSTATION	6.2.3.1.4.	8.2.3.1.4.
EMERGENCY CLOSE	6.2.2.5.3.	8.2.2.5.3.
FILL AREA	6.2.2.1.3.	8.2.2.1.3.
GDP	6.2.2.1.6.	8.2.2.1.6.
GDS ESCAPE1	6.2.1.7.	8.2.1.7.
INSERT SEGMENT	6.2.3.2.3.	8.2.3.2.3.
OPEN WORKSTATION	6.2.1.1.	8.2.1.1.
POLYLINE	6.2.2.1.1.	8.2.2.1.1.
POLYMARKER	6.2.2.1.2.	8.2.2.1.2.
REDRAW ALL SEGMENTS ON WORKSTATION	6.2.3.1.6.	8.2.3.1.6.
RENAME SEGMENT	6.2.3.1.3.	8.2.3.1.3.
SET ASPECT SOURCE FLAGS	6.2.2.2.29.	8.2.2.2.29.
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 CHARACTER VECTORS	6.2.2.2.26.	8.2.2.2.26.
SET CLIPPING RECTANGLE	6.2.2.4.1.	8.2.2.4.1.
SET COLOUR HEADER	6.2.6.2.	8.2.6.2.
SET COLOUR REPRESENTATION	6.2.2.2.28.	8.2.2.2.28.
SET DEFAULTS	6.2.1.6.	8.2.1.6.
SET DEFERRAL STATE	6.2.2.5.2.	8.2.2.5.2.
SET DOMAIN RING	6.2.6.1.	8.2.6.1.
SET FILL AREA COLOUR INDEX	6.2.2.2.14.	8.2.2.2.14.
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 REPRESENTATION	6.2.2.2.11.	8.2.2.2.11.
SET FILL AREA STYLE INDEX	6.2.2.2.15.	8.2.2.2.15.
SET HIGHLIGHTING	6.2.3.1.7.	8.2.3.1.7.
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 MARKER SIZE SCALE FACTOR	6.2.2.2.9.	8.2.2.2.9.
SET MARKER TYPE	6.2.2.2.8.	8.2.2.2.8.
SET PATTERN REFERENCE POINT	6.2.2.2.17.	8.2.2.2.17.
SET PATTERN REPRESENTATION	6.2.2.2.16.	8.2.2.2.16.
SET PATTERN VECTORS	6.2.2.2.18.	8.2.2.2.18.
SET POLYLINE COLOUR INDEX	6.2.2.2.5.	8.2.2.2.5.
SET POLYLINE INDEX	6.2.2.2.2.	8.2.2.2.2.
SET POLYLINE REPRESENTATION	6.2.2.2.1.	8.2.2.2.1.
SET POLYMARKER COLOUR INDEX	6.2.2.2.10.	8.2.2.2.10.
SET POLYMARKER INDEX	6.2.2.2.7.	8.2.2.2.7.
SET POLYMARKER REPRESENTATION	6.2.2.2.6.	8.2.2.2.6.
SET SEGMENT PRIORITY	6.2.3.1.10.	8.2.3.1.10.
SET SEGMENT TRANSFORMATION	6.2.3.1.9.	8.2.3.1.9.
SET TEXT ALIGNMENT	6.2.2.2.27.	8.2.2.2.27.
SET TEXT COLOUR INDEX	6.2.2.2.24.	8.2.2.2.24.
SET TEXT FONT AND PRECISION	6.2.2.2.21.	8.2.2.2.21.
SET TEXT INDEX	6.2.2.2.20.	8.2.2.2.20.
SET TEXT PATH	6.2.2.2.25.	8.2.2.2.25.
SET TEXT REPRESENTATION	6.2.2.2.19.	8.2.2.2.19.
SET VISIBILITY	6.2.3.1.8.	8.2.3.1.8.
SET WORKSTATION VIEWPORT	6.2.2.3.2.	8.2.2.3.2.
SET WORKSTATION WINDOW	6.2.2.3.1.	8.2.2.3.1.
TEXT	6.2.2.1.4.	8.2.2.1.4.
UPDATE WORKSTATION	6.2.2.5.1.	8.2.2.5.1.

Appendix A2D

SELECTION OF THE GEOMETRIC DISPLAY

The geometric display is selected by means of the US sequence (VPCI, Videotex Presentation Control Information):

<US><3/2><y>

as described in part 0 of this Recommendation. US is the UNIT SEPARATOR control and is coded 1/15. The <y> indicates the highest level of primitives, which is embedded in the geometric data following the US sequence. The currently defined values of <y> are given in table A2D-1 (T/TE 06-02).

<y>	Highest level
2/1	L0a
2/2	L1a
2/3	L2a

Table A2D-1 (T/TE 06-02). Relation between <y> code and levels.

After this US sequence (VPCI) all data is regarded as geometric data.

Appendix ANNA

SRM CONFORMANCE

ANNA 1. VIDEOTEX SRM CONFORMANCE FOR GEOMETRIC DISPLAY

The conformance section consists of two parts:

— Geometric display: (GDS level 0a, 1a or 2a)
— Combination rules: (pseudo-layered display or time dependent display)

For one geometric display level two profiles are proposed which give slightly different selections of functions from the conformance section as shown in the following diagram.

GEOMETRIC PRIMITIVES
LEVEL 0a geometric primitives encoding principles
LEVEL 1a for further study
LEVEL 2a for further study

PROFILE "X1"
Display Characteristics: 4-bit-map 16 colour CLUT 192v × 240h pixels minimum resolution no "dynamic effects" (scrolling, flashing, concealing)

COMBINATION RULES
SINGLE-PLANE DISPLAY time dependent display
pseudo-layered display
MULTI-PLANE DISPLAY for further study

PROFILE "X2"
Display Characteristics: 4-bits bit-map + attribute & colour memory 32 alphamosaic and 16 geometric colour CLUT 240v × 240h pixels minimum resolution no scrolling

ANNA 2. GEOMETRIC DISPLAY

ANNA 2.1. **Level 0a**

ANNA 2.1.1. *Geometric Primitives*

At least all primitives (mandatory or not mandatory) of level 0a are supported with the following exceptions:

— GDP ELLIPSE
— GDP ELLIPTIC ARC
— GDP ELLIPTIC ARC CHORD
— GDP ELLIPTIC ARC PIE
— SET PATTEN VECTORS
— GDS ESCAPE1
— SET PATTERN REFERENCE POINT

ANNA 2.1.1.1. Polyline

Line types: the terminal must be capable of displaying all four defined line types (SOLID, DASHED, DOTTED, DASHED-DOTTED).

Representation of polyline vertices: at each vertex the line should appear continuous with no unfilled sectors.

Line width: if the line width is smaller than the smallest width which is available, the line is presented with the smallest possible width.

Nominal line width: 0.75/minimum vertical resolution.

Maximum line width: smallest DDA edge.

Memory for storage of a minimum of 512 defined vertices is required.

ANNA 2.1.1.2. Polymarker

Polymarker types: the terminal must be capable of displaying all the five defined polymarkers (DOT, PLUS-SIGN, ASTERISK, CIRCLE, DIAGONAL-CROSS).

Polymarkers clipping: markers are exactly clipped at the clipping rectangle (pixel basis). A marker is visible if the marker position is within the clipping rectangle and the workstation window. Partially visible markers are clipped against the clipping rectangle and the workstation window on a pixel basis.

Workstation transformation: markers are transformed according to the transformation rules applied. Only the marker DOT is not transformed. The line width of the markers is also not transformed.

Nominal marker size: the width of the default character box.

Maximum marker size: smallest DDA edge.

Memory for storage of a minimum of 512 defined vertices is required.

ANNA 2.1.1.3. Fill Area

The terminal should be capable of processing at least 16 intersections on any horizontal or vertical line.

Memory for storage of a minimum of 128 defined vertices is required.

If incoming data exceeds these limitations the result may be implementation dependent.

The boundary for filled area is drawn only in case of HOLLOW.

Hatch is applied to the previously set background. The distance between the lines is not specified but left to terminal design.

Interior style: the terminal must be capable of processing the following interior styles: HOLLOW, SOLID and HATCH.

Interior hatch style: the terminal must be capable of displaying the 6 defined hatch styles (line width and spacing are implementation dependent): VERTICAL LINES, HORIZONTAL LINES, 45_DEGREE LINES, −45_DEGREE LINES, CROSSED LINES, VERTICAL AND HORIZONTAL, CROSSED LINES 45 AND −45 DEGREES.

PATTERN interior style is not supported: the fallback is HOLLOW.

Fill Area interior style hollow: the area to be filled is surrounded by a close boundary and the boundary is a part of the filling area. For the boundary representation, the terminal shall use the following aspects:

line type — solid
line width — nominal line width
line colour — fill area colour

ANNA 2.1.1.4. Text

Text and Font precision: the terminal must be capable of processing 1 font (character font is implementation dependent).

Text alignment: in certain cases of text alignment a character buffer is required. The minimum buffer size is 72 bytes.

Character sets: in the geometric mode only the primary set of graphic characters (default G0) and the supplementary set of graphic characters (default G2) are available (default state).

7-bit environment: the shift functions SI, LS2 and SS2 are available.

8-bit environment: invocation sequences and shift functions are ignored. G0 set in col 2 to 7, G2 set in col 10 to 15.

The ratio between character expansion factor and character line width should be retained. Character height/width corresponds to the dimension of the character box:

The height of the default character box equals 1/24 of the vertical DDA edge.

The width of the default character box equals 1/40 of the horizontal DDA edge.

The capline position is 9/10 of the character box height from the character box base.

The baseline position is 2/10 of the character box height from the character box base.

For calculating the character size, the terminal supports continuous range of character height and character expansion factor. The following limitations are required:

minimum character expansion factor: 0.1
maximum character expansion factor: 10.0

L-Set: the application of the L-Set is not considered in combination with geometric information.

ANNA 2.1.1.5. Cell Array

If the points P, Q and R are co-linear, a line has to be drawn. The colour(s) in which the line will be drawn is/are implementation dependent.

The whole of a cell array is to be occupied by the selected pattern. This implies that the subdivision of the cell array into cells does not lead to exactly but only approximately equally dimensioned cells because of the difference between NDC and DC. All four corners of the cell array shall be inside the square of [−7,7]★[−7,7]. Otherwise the primitive will be ignored.

ANNA 2.1.1.6. Arcs/Circles

In case of arcs of circles if the calculated centre is outside the co-ordinates addressed by the available range of magnitude/granularity codes or if the radius exceeds 7 NDC the representation of the primitive is not guaranteed.

ANNA 2.1.1.7. Arc Pie

The deviation of the calculated centre of an arc pie (if defined with 3 points) should be smaller than 1/40 of the screen width (in case of an opening angle of 10 degrees).

ANNA 2.1.1.8. Open Workstation (id = 0)

In case the WS was previously closed, Open WS performs implicitly the following actions:
SET WS WINDOW (0.0, 0.0, 1.0, 1.0) (Default values)
SET WS VIEWPORT (0.0, 0.75, 0.0, 0.75) (Default values)

ANNA 2.1.1.9. Close Workstation

No clear.

ANNA 2.1.1.10. Update Workstation

The effect of this primitive is a clear of the display.

In case of "perform", the primitive is to be processed. It performs an update of workstation window and workstation viewport.

In case of "postpone", the primitive is not to be ignored.

ANNA 2.1.1.11. Set Workstation Viewport

The sequential order of transmitted parameters is:
Xmin Xmax Ymin Ymax

(Xmin, Xmax) specifies the width of the WS Viewport as a fraction of the horizontal DDA edge.

(Ymin, Ymax) specifies the height of the WS Viewport (of the Default Viewport). The largest value in the Y direction (Ymax) equals 0.75.

This CD space (0.0, 2.0, 0.0, 0.75) is mapped entirely to the display surface with the aspect ratio of approximately 4:3.

ANNA 2.1.2. *Specification of a Mandatory Workstation*

ANNA 2.1.2.1. Workstations

The terminal is identified by the workstation number 0.

Information addressed to a different workstation is not displayed on workstation 0, but may be processed by other available workstations.

The default state of the workstation with ID = 0 is OPEN and ACTIVE. Open or Activate again does not affect this state.

The output is generated at all active output workstations. The workstation description table has its entry "DYNAMIC MODIFICATION ACCEPTED FOR WORKSTATION TRANSFORMATION" set to IRG: so the parameters of "workstation window" and "workstation viewport" have no immediate effect on the display.

ANNA 2.1.2.2. Bundle Tables

All bundle tables have only one entry with the default values of the individual attributes.

ANNA 2.1.2.3. Transformations

The expansion factor for window/viewport transformation (applied to any window size) is max. 7.

The reduction factor is not limited.

The line width is transformed (enlarged or reduced).

The line width of the polymarkers is not transformed.

ANNA 2.1.3. *Encoding Principles*

ANNA 2.1.3.1. Encoding of Co-ordinates

The facilities and functions are coded according to T/TE 06-01.

All values covered by the range of "SET CO-ORDINATE PRECISION" parameters of "SET REAL PRECISION" are to be processed.

Points within the NDC space $(-7,7)\bigstar(-7,7)$ shall be processed by the terminal.

Points which are outside this space may be ignored.

BASIC FORMAT: for values encoded in basic format, terminals must be capable of, at least, processing 15 bits plus a sign bit.

REAL FORMAT: the difference, magnitude_code − granularity_code, indicates the number of bits which are necessary to store points with full precision.

The parameters of SET CO-ORDINATE PRECISION satisfy the following conditions: for the difference, magnitude_code − granularity_code, terminals must be capable of processing at least values of 16. Violation of the magnitude code or of the granularity code by any transmitted value results in an error.

Changes of granularity immediately affect the processing of subsequently following data.

The complete INCREMENTAL MODE is supported.

ANNA 2.1.3.2. Encoding of Colour Index Lists

BASIC format, BITSTREAM format and RUN LENGTH format are all supported for colour index lists.

ANNA 2.2. **Level 1a**

For further study.

ANNA 2.3 **Level 2a**

For further study.

ANNA 3. COMBINATION RULES

ANNA 3.1. Architecture and Display Parameters

The architecture as well as the display design are basically freely selectable. It is assumed that the terminal is based:
— either on a ONE_DISPLAY_PLANE architecture containing alphamosaic and geometric information,
— or on a TWO_DISPLAY_PLANE architecture containing alphamosaic and geometric information in different planes.

In the case of a one_display_plane architecture there are two manners of implementing it:
— TIME DEPENDENT DISPLAY: the last received information (alphamosaic or geometric) overwrites the display,
— PSEUDO-LAYERED DISPLAY: alphamosaic information only overwrites geometric information.

The aspect ratio of the DC_space is approx. 4:3. The origin of the DC_space is mapped onto the lower left corner of the defined display area (DDA). Optimally the DC_space is congruent to the alphamosaic DDA. If required by the implementation constraints, the upper right corner may differ slightly.

The alphamosaic service row is located outside the DC_space (Definition and use of the service row is national dependent).

When displaying pictures with an aspect ratio of the DC_space different from 4:3 aspect ratio, parts of the display area are unused.

ANNA 3.2. Combination Rules

ANNA 3.2.1. *Default Conditions*

The alphamosaic mode represents the default state, i.e. in the default state, the display memory contains spaces with default attributes (default spaces) and the full screen background is black.

ANNA 3.2.2. *Attributes Applied to Space*

The "foreground colour" and the attributes "window", "protected area" and "marked area" of the space character have no meaning with regard to the composition rules.

Any code positions (2/0 and 10/0) representing a space may be used. Spaces with diacritical marks are no default spaces.

ANNA 3.2.3. *Application of Geometric Information*

ANNA 3.2.3.1. One Display Plane: "Time Dependent Display"

Geometric information is displayed without restrictions after receiving a geometric VPDE.

ANNA 3.2.3.2. One Display Plane: "Pseudo-Layered Display"

In the default state of alphamosaic display, geometric information is displayed without restrictions after receiving a geometric VPDE.

If the display contains alphamosaic information, geometric information is only displayed in those areas which do not contain alphamosaic information; i.e. alphamosaic characters have priority and the geometric information terminates at the borders of the alphamosaic character positions (except in the case of default spaces) even if the character background is transparent.

ANNA 3.2.3.3. Two Display Plane

For further study.

ANNA 3.2.4. *Application of Alphamosaic Information*

ANNA 3.2.4.1. One Display Plane: "Time Dependent Display"

Alphamosaic characters delete all geometric information at the relevant character positions of the display, the previous geometric information cannot be displayed again.

ANNA 3.2.4.2. One Display Plane: "Pseudo-Layered Display"

Alphamosaic characters delete all geometric information at the relevant character positions of the display, the previous geometric information cannot be displayed again. This does not apply to "clear screen" and "default spaces" which only affect the alphamosaic display.

If alphamosaic information is deleted by default spaces, these areas can subsequently be overwritten by geometric information.

ANNA 3.2.4.3. Two Display Plane

For further study.

ANNA 3.2.5. *Workstation Transformation*

The transformation from NDC to DC does not impact alphamosaic information.

ANNA 3.2.6. *Background Colour*

The full screen background colour is common to both geometric and alphamosaic modes.

ANNA 3.3. **Clear Screen**

Clear screen deletes all alphamosaic information contained in the display memory and functions as described in T/TE 06-01, Part 1.

Appendix ANNB

SRM PROFILES

The following profile options are recognised by CEPT for existing Videotex services which are mixing Alphamosaic and Geometric information.

ANNB 1. ONE DISPLAY PLANE

ANNB 1.1. Geometric Display Level 0a

This profile is in conformance with Geometric level 0a and one of the alphamosaic profiles.

It is based on a One Display Plane architecture containing Alphamosaic and Geometric information.

At least one 4-bit plane is implemented (16 colours).

ANNB 1.1.1. *Time Dependent Display (Profile X1)*

ANNB 1.1.1.1. Colours

The terminal has at least a resolution of 4 bits/pixel, so only 16 different colours may be used at the time on a screen through a colour lookup table.

When a pure alphamosaic frame is to be displayed, then it is possible to use the 16 colours available in the colour map through the mechanism described in T/TE 06-01, Part 1 (use of set C1) and Part 5 (define colour). However, for those terminals using only 4 bits/pixels display, it is clear that the colour map is limited to the first 16 entries of the colour map.

When a pure geometric frame is to be displayed, the same set of 16 colours is available and may be handled as defined by the common CEPT/ECMA geometric standard.

When a mixed mode frame is to be displayed, the same rules may apply. The colour map may be accessed either from the geometric part or from the alphamosaic part (T/TE 06-01, Part 5).

ANNB 1.1.1.2. Initialisation of the Colour Table

The proposed mechanism works correctly when assuming that all frames are produced by a single information provider: it is the responsibility of the host to use correctly the colour map throughout the whole application.

The Colour Reset Unit (T/TE 06-01, Part 5) is used to set the colour map to its default values.

ANNB 1.1.1.3. Minimum Display Resolution

A minimum resolution of 192v × 240h pixels is required in order to be compatible with Personal Computer display add-on boards.

ANNB 1.1.1.4. Minimum Number of Vertices for Fill Area

Although a minimum of 128 vertices is required for fill area (ref. Annex A, 2.1.1.3.), support for 512 vertices is recommended, taking the growing capabilities into account.

ANNB 1.1.1.5. Alphamosaic Limitations

Dynamic effects like scrolling, flashing and concealing are not supported after receiving a geometric VPDE. These limitations are not effective in a pure alphamosaic mode.

ANNB 1.1.1.6. Geometric Limitations

All primitives given in the conformance section are supported.

Text precisions STRING and CHARACTER are required.

Strings are clipped on a pixel basis.

Terminal support for at least 4 values of the direction of the character width vector is required: 0 deg, 90 deg, 180 deg, 270 deg.

Other values will be mapped to the nearest supported direction.

After mapping characters on the display surface, at least the following combinations of resulting vectors (in DC) are supported:

Height vector (fractions of vertical DDA edge)	Width vector (fractions of horizontal DDA edge)
(0, 1★ (1/24))	(1 ★ (1/40), 0)
(0, 2★ (1/24))	(1.5★ (1/40), 0)
(0, 3★ (1/24))	(2 ★ (1/40), 0)
(0, 4★ (1/24))	(2.5★ (1/40), 0)
(0, 5★ (1/24))	(3 ★ (1/40), 0)

The mapping of resulting character vectors on these combinations is implementation dependent.

Alphamosaic "Clear Display" command may affect the geometric display.

ANNB 1.1.2. *Pseudo-Layered Display (Profile X2).*

ANNB 1.1.2.1. Colours

One 4-bit plane is implemented (16 colours).

Two colour groups are available, one containing the first 16 colours, the second containing the second 16 colours of the colour map.

Switches between the colour groups occur implicitly by bit 5 of the colour index parameter (possibly spill-over of colour information from colour group 1 to colour group 2).

The operand colour index (the colour index parameter /4 bits) of the colour primitives corresponds to the ordinal entries in the geometric colour map.

In the geometric mode, colour group 1 represents the default state.

The application of 16-colour-DRCS in Geometric pictures leads to non-guaranteed representations because of the dependencies of colours in both modes.

Allocation of colour map entries to values of set colour precision (direct mapping of values to colour entries for the following colour index list specification of colour indices):

Precision	Value	Colour table entries
1 bit	0.1	colours 8 and 7
2 bits	0. .3	colours 16 . . . 19
3 bits	0. .7	colours 8 . . . 15
4 bits	0. .15	colours 0 . . . 15
5 bits	0. .31	colours 0 . . . 31

The default colour is white, i.e. the colour index 1 of the geometric part directly points to colour entry 7 of the colour map (only in case the colour index is not set explicitly).

ANNB 1.1.2.2. Minimum Display Resolution

A minimum resolution of 240v × 240h pixels is required.

ANNB 1.1.2.3. Alphamosaic Limitations

"Format": In the Geometric mode both formats (20v × 40h and 24v × 40h characters) are permitted, but the representation of a picture is not guaranteed when the format has been changed after the representation of geometric information.

"Scrolling": In Geometric pictures the application of scrolling (character scrolling) does not have a guaranteed representation.

ANNB 1.1.2.4. Geometric Limitations

All primitives given like in the conformance section are supported with the following exceptions:
CIRCULAR ARC CENTRE
CIRCULAR ARC CENTRE CHORD
CIRCULAR ARC CENTRE PIE
SET COLOUR REPRESENTATION
SET COLOUR HEADER

ANNB 1.1.2.5. Text

STRING, STROKE and CHARACTER precision are required.

Text precision—STRING: character height, character width and character expansion factor are used to represent the string. The string is clipped exactly at the clipping rectangle (pixel basis).

Text precision—CHARACTER: all text attributes are used. The string is clipped exactly at the clipping rectangle (pixel basis).

Changes in character size take place on a stroke-oriented basis.

ANNB 1.2. **Geometric Display Level 1a**

For further study.

ANNB 1.3. **Geometric Display Level 2a**

For further study.

ANNB 2. **TWO DISPLAY PLANE**

ANNB 2.1. **Geometric Display Level 0a**

For further study.

ANNB 2.2. **Geometric Display Level 1a**

For further study.

ANNB 2.3. **Geometric Display Level 2a**

For further study.

History

Document history	
November 1990	First Edition
January 1996	Converted into Adobe Acrobat Portable Document Format (PDF)