INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

ANNEX TO BLUE BOOK

FASCICLE VII.5 — ANNEX II

DATA SYNTAX II
FOR INTERNATIONAL INTERACTIVE
VIDEOTEX SERVICE

RECOMMENDATION T.101, ANNEX C

IXTH PLENARY ASSEMBLY
MELBOURNE 1988

Geneva 1990

INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

ANNEX TO BLUE BOOK

FASCICLE VII.5 — ANNEX II

DATA SYNTAX II FOR INTERNATIONAL INTERACTIVE VIDEOTEX SERVICE

RECOMMENDATION T.101, ANNEX C

IXTH PLENARY ASSEMBLY
MELBOURNE 1988

Geneva 1990

ANNEX C

(to Recommendation T.101)

Data Syntax II

Note: This data syntax generally corresponds to the 'CEPT T/CD 06-01) presentation layer data syntax adopted by some European countries.

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

VIDEOTEX PRESENTATION LAYER DATA SYNTAX

CONTENTS

PART 5 Define COLOUR

PART 6 Define FORMAT

PART 7 TRANSPARENT Data

PART 8 RESET

PART 9 TELESOFTWARE

(For further study)

PART 10 TERMINAL FACILITY IDENTIFIER (T.F.I.)

PART 11 TIMING CONTROL

CONTENTS

1 INTRODUCTION

VIDEOTEX systems are text communication systems with the capability of a given level of pictorial representation and a repertoire of display attributes. The text and the pictures obtained are intended to be displayed using the current television (TV) raster standards of the different countries.

Videotex services will be provided in different ways in different countries. The Videotex services may be a distributed network of independent computers or a hierarchy of computers with external databases or a mixture of both. It is probable that in all countries Videotex terminals will primarily access the Videotex services via the switched telephone network, over which data is transmitted to a terminal which generates displays. Three types of display have been identified and are described and defined in this recommendation:

1. Alpha-mosaic

2. Geometric

3. Photographic

Other types of display may be defined in the future. Each type of display may be used simultaneously, though data for each type of display is separated during transmission. The way in which data is used to generate a display may be modified by 'management data'. Management data may affect more than one type of display.

1.1 Coding Principles

1.1.1 Identification Of Data Types

Different types of display data and management data are separated into different 'Videotex Presentation Data Elements' (VPDEs) during transmission.

'Videotex Presentation Data Elements' (VPDEs) are made up of two parts: 'Videotex Presentation Control Element' (VPCE) which identifies the type of data and 'Videotex Service Control Element' (VSCEs) which contain the data.

```
<---------------- VPDE ---------------->
 ______________________________________________
|          |                          |
|   VPCE   |           VSCE           |   VPCE
|__________|__________________________|__________
```

VPCEs are coded in the form US X where X is a character from:

columns 4-7 for alphamosaic data
column 2 for management data
column 3 for other data

The following VPCEs have been provisionally assigned:

TERMINAL FACILITY IDENTIFIER	US 2/0 and US 2/1
Define DRCS:	US 2/3
Define COLOUR:	US 2/6
Define FORMAT:	US 2/13
TIMING CONTROL	US 2/14
RESET	US 2/15
ALPHAMOSAIC display data:	US <any character from column 4-7>
Reserved (see note 1)	US 3/0
GEOMETRIC display data (3D)	US 3/1
GEOMETRIC display data (2D)	US 3/2
PHOTOGRAPHIC pixel data	US 3/4
PHOTOGRAPHIC table data	US 3/5
SOUND	US 3/11
Reserved (see note 2)	US 3/12
TELESOFTWARE data	US 3/14
TRANSPARENT data	US 3/15

Note 1: US 3/0 is reserved for Geometric display data according to the Montpellier version of this recommendation.

Note 2: US 3/12 is reserved for private use.

US is the UNIT SEPARATOR control and is coded 1/15

1.1.2 Use of default values

Where data fields are used to describe parameters of the following data (eg. the DRCS header) default values for these fields have been assigned. If the data field is not transmitted then the terminal will apply the default value.

It is anticipated that some terminals will only be able to process data which conforms to these default values, to simplify their operation these terminals may ignore such data, unless the parameters describing that data are omitted (implying that the data conforms to the default).

It is therefore recommended that when a parameter is equal to the default value that field is not transmitted, if it is transmitted then the response of the terminal is not guaranteed.

1.2 Display Principles

1.2.1 Defined Display Area

The defined display area is a rectangular area of the screen within which the text and pictorial information is displayed.

The ratio of the width to the height (aspect ratio) of this area should be 4:3.

For the aphamosaic display this area is composed of a defined number of rows each with a defined number of character positions. The default is 24 rows of 40 character positions.

For the geometric display the bottom lefthand corner of the defined display area is addressed as (0,0) and the upper righthand corner is defined as (1,0.75).

The photographic display area also maps to the same area. The top lefthand pixel of the photographic display area is addressed as (1,1).

The possibility of defining different aspect ratios is for further study.

1.2.2 Display structure

The theoretical structure of the display consists of the following layers in order of precedence:

1. Alphamosaic character foreground and background layers (see Part 1 Section 1.2)

2. Geometric layer or layers (see Part 2)

3. Photographic layer (see Part 3)

4. Full screen background layer (see Part 1)

5. Any other video source

Changing the display structure is for further study.

2.0 REFERENCES

This document is intended to be compatible with the following recommendations and standards:

CCITT Recommendation T50	International alphabet No 5
CCITT Recommendation F300	Videotex service
CCITT Recommendation T100	International information exchange for interactive videotex
CEPT T/CD 01-01 (Rev 1987)	Videotex Presentation Layer Data Syntax
ISO Standard 2022 (Rev 79)	Code Extension Techniques for use with the ISO 7-bit and 8-bit coded character set
ISO Standard 6937	Information processing - coded character sets for text communication
ISO Standard 6429.2	Draft standard - additional control functions for character imaging devices

3.0 DEFINITIONS

BIT-COMBINATION
Bit-combination is an ordered set of bits that represents a character.

BORDER AREA
Border area is that part of the display screen (visible display) which is outside the defined display area. (See Note and Figure 1 below).

CONTROL CHARACTER
Control character is a control function, the coded representation of which consists of a single bit-combination.

CONTROL FUNCTION
Control function is an action that affects the recording, processing, transmission or interpretation of data. The coded representation of a control function consists of one or more bit-combinations.

DEFINED DISPLAY AREA
The defined display area is a rectangular area of the screen within which the text and pictorial information is displayed (see Figure 1 and section 1.2.1).

```
  ______________________________
 /                              \
|  .----------------------------.  |
|  :                            :  |
|  :                            :  |
|  :                            :  |<----- Full Visible Display
|  :                            :  |
|  :                            :  |
|  :                            : <------- Border Area
|  :                            :  |
|  :                            :  |
|  :                         <------------- Defined Display Area
|  :                            :  |
|  :                            :  |
|  '----------------------------'  |
 \______________________________/
```

FIGURE 1 Full Visible Display, Border Area and Defined Display Area

NOTE

The default format of defined display area for the alphamosaic option is given in Part 1 Section 1.1.2

GRAPHIC CODE EXTENSION
Graphic code extension is the method of encoding graphic characters in excess of those which may be represented by the code combinations of the basic code table. Alternative sets of 94 characters may be designated by means of shift functions.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1.0 DESCRIPTION

1.1 Introduction

Data sent to the terminal is used to generate alphamosaic displays in which text and graphic characters are displayed, usually in a fixed format of rows and columns.

1.1.1 Definitions

ACTIVE POSITION
Active position is the position on the screen from which subsequent actions would take place if they were activated.

BACKGROUND COLOUR
Background colour is the colour of that area of the character cell not occupied by the foreground colour.

CHARACTER
Character is a member of a set of elements that is used for organization, control or representation of data. A character repertoire contains two types of element: graphic characters and control functions.

CODED CHARACTER SET
Coded character set is a set of unambiguous rules that establishes a character set and their one-to-one relationship between the characters of the set and the bit-combinations.

CODE TABLE
Code table is a table showing the character corresponding to each bit-combination in a code. A code table is normally represented as a rectangular matrix of columns and rows.

FOREGROUND COLOUR
Foreground colour is the colour of the graphics shape that is being displayed in a character cell.

FORMAT EFFECTORS
Format effectors are control functions that influence the positioning of text and pictorial images, within the defined display area on a presentation device.

GRAPHIC CHARACTER
Graphic character is a character, other than a control function, that has a visual representation normally printed or displayed.

HOME POSITION
Home position is the first characacter position of the first row of the defined display area.

MARKERS
Markers are flags in a memory to show where attribute controls have been set; they are associated with the leading edge of the character position.

PARALLEL ATTRIBUTES
Parallel attributes are the property of the active position and move with it under the action of format effectors or spacing display characters (including space). They apply to the displayed characters subsequently received until the attributes are changed by relevant controls including certain format effectors (CS,APA,APH). They also apply to spacing display characters (including space) inserted by control commands.

SERIAL ATTRIBUTES
Serial attributes are set between markers on a row. They apply from the position of the active position at the time they are received to the end of the row or until a contradictory marker is reached.

WRAPAROUND CONTROLS
Wraparound controls comprise a set of rules which govern what happens when the active position attempts to move off the defined display area.

1.1.2 Format

The default format is 24 rows of 40 columns with automatic wraparound on rows and columns. The format and wraparound may be changed by the 'Define FORMAT' VPDE.

1.1.3 Characters

Alphanumeric, block mosaic, smoothed mosaic and line drawing characters are defined. Accented characters are coded using the composition method of coding. The fixed repertoire of characters may be extended with dynamically redefinable characters loaded via the 'Define DRCS' VPDE.

1.1.4 Format Effectors

Characters may be positioned within the defined display area by means of format effector controls which move the active position, usually in units of one character position.

1.1.5 Attributes

The presentation of characters on the screen may be modified by the application of display attributes. Attributes may be applied to the full screen, full row, part of a row (serial) or to subsequently printed characters (parallel).

1.1.6 Device Control Functions

The action of scrolling, the display of the cursor and similar functions may be controlled by codes transmitted to the terminal.

1.2 Theoretical Terminal Model

The videotex service, alphamosaic option, may be described in the form of an ideally perfect theoretical terminal. This model is detailed hereafter.

1.2.1 Description

The theoretical terminal model is based on a separation between the visual content of the page and its structure. It can be described as if it were composed of three memories.

1. One character memory where one character address from the character generator is stored at every character locati-

2. One attribute memory where all the attributes ..e set in parallel at every location of the screen plus registers for full screen and full row background. The number of registers in this memory is equal to the number of rows plus two. The last two registers refer to the top segment (above the defined display area) and the bottom segment (below the defined display area) of the full screen background.

3. One marker memory where every attribute or group of attributes or display functions may be flagged at any character location. When an attribute or function is modified according to the serial mode, this modification occurs between the current character location and the next flag related to this attribute or function (or up to the end of the row).

1.2.2 Operation of 'Parallel' and 'Serial' Mode Controls

Both the 'parallel' and 'serial' modes set only serial attributes in the terminal memory (which means that all attributes set, by either mode, are active between markers or up to the end of the row).

Parallel mode controls only apply attributes to the character locations where the cursor prints a character (including space), and remain with the cursor when it moves between rows except when the control codes CS, APA or APH are received. An attribute is copied into the attribute memory and markers are set wherever an attribute is changed. Whenever a continuous string of graphic characters, including SPACE, is written on a row under the parallel mode, then, if there is a change of attribute(s) between adjacent character locations, a marker(s) is created or moved. In addition, any existing markers within the overwritten part of the row are deleted.

Serial mode control codes insert or modify a marker into the marker memory and cause an attribute to be copied immediately into the attribute memory until a contradictory or complementary marker is encountered in the marker memory, or until the end of the row. When in the serial mode, the writing of a graphic character does not modify by itself the attribute in the attribute memory.

Parallel and Serial mode control codes are taken from different control sets and therefore may be unambiguously recognised by the terminal. This is achieved by invoking the appropriate Parallel or Serial C1 set.

The invocation of a Parallel or Serial C1 set will cause the mode of operation of the terminal to switch. Thus in the Serial mode any parallel attributes locked to the cursor will have no effect. Their effect will be restored when the Parallel mode is re-invoked.

Interaction of Serial and Parallel mode control codes: a subsequent (in time) Parallel mode control code will apply to all characters which the cursor writes while in the Parallel mode irrespective of how their attributes had been previously set.

A subsequent (in time) Serial mode control will propagate to the right of the cursor position at which it is received until it meets a contradictory marker.

A full row attribute (other than the background colour) has the effect of overwriting the defined attributes on all the positions of the row and has the effect of deleting all contradictory or complementary attribute markers. The full screen attribute has the same effect but written to all rows it does not delete markers.

1.2.3 Layered Structure

The alphamosaic display area acts as if it were composed of 2 independent layers.

- A full screen background layer which may be partitioned into rows (with time-dependent precedence).

- A defined display area character layer. The colour of this layer is either BACKGROUND COLOUR or FOREGROUND COLOUR.

As indicated in Part 0 of this document geometric and photographic layers may exist between the full screen background layer and the defined display area character layer.

1.2.4 Action of Attributes on Layers

The transparent colour in the defined display area character layer (foreground or background) allows see-through to the underlying full screen background layer or the geometric or photographic layers if present.

The character BACKGROUND COLOUR attribute, including the transparent value, applies only to the defined display area character layer.

The full screen and full row BACKGROUND COLOUR attribute affects only the full screen background layer. Its transparent value refers to the video picture.

The full screen, full row and parallel INVERT attribute-controls affect simultaneously and symmetrically the FOREGROUND and the BACKGROUND in the defined display area character layer.

All other full screen or defined display area attributes apply only to the foreground of the defined display area character layer (except the SIZE attribute which also affects the background of this layer). The actions of the combined effect of INVERT and the transparent colour are to be seen in Table 1.

TABLE 1 TRUTH TABLE FOR ATTRIBUTE SETTINGS

ATTRIBUTE SETTINGS			Colour of resultant display	
FOREGROUND/BACKGROUND	FOREGROUND	BACKGROUND	FOREGROUND and underline	BACKGROUND colour
NORMAL	Normal (c)	Normal (b)	c	b
		Transp. (b)	c	a
	Transp. (c)	Normal (b)	a	b
		Transp. (b)	a	a
INVERT	Normal (c)	Normal (b)	b	c
		Transp. (b)	a	c
	Transp. (c)	Normal (b)	b	a
		Transp. (c)	a	a

Foreground of defined display area character layer (c)
Background of the defined display area character layer (b)
Full screen background layer (a)

1.3 Defined Attributes and Qualified Areas

1.3.1 FOREGROUND COLOUR

This is the colour of the graphics shape being displayed. The colour may be any colour from the available colour tables including 'transparent' in which case the full screen background colour (or the geometric or photographic layers if present) is seen.

1.3.2 BACKGROUND COLOUR

CHARACTER BACKGROUND COLOUR
This is the colour of the remaining area of the character cell. The colour may be any colour from the available colour tables or be transparent in which case the full screen background colour (or the geometric or photographic leyers if present) is seen.

FULL SCREEN or FULL ROW BACKGROUND COLOUR
This is the colour of layer (a) of the theoretical model, see section 1.2.3.

1.3.3 LINED

Alphanumeric characters are displayed with an underline in which the underline is considered to be part of the shape of the graphics character. Mosaic characters and line drawing characters are displayed in separated font, see section 2.1.2.

1.3.4 SIZE

There are four states of character size:

NORMAL-SIZE
The extent of characters occupies the active position.

DOUBLE-HEIGHT
The extent of characters occupies both the active position and the corresponding position of the adjacent row.

DOUBLE-WIDTH
The extent of characters occupies both the active position and the next position of the same row.

DOUBLE-SIZE
The extent of characters occupies the active position, the next position on the same row and the corresponding two positions on the adjacent row.

See section 1.4 for rules for the application of the SIZE attribute.

1.3.5 FLASH

The following attribute states are defined:

STEADY
The characters are displayed normally.

FLASHING

NORMAL FLASH
The characters are displayed alternately in the prevailing foreground colour and in the prevailing background colour.

INVERTED FLASH
This is as for FLASH but on the inverted phase of the flashing clock.

REDUCED INTENSITY FLASH (flash between colour tables)
The characters are displayed alternately in the prevailing foreground colour and in the equivalent colour of another colour table. Table 1 colours adopt table 2 colours, table 2 colours adopt table 1 colours, table 3 colours adopt table 4 colours, and table 4 colours adopt table 3 colours. (See section 1.5.3).

STATES OF FLASHING

Each of the above states may be displayed at either of the following rates:

50% ON/OFF ratio at about 1Hz

33 % ON, 1st phase)
33 % ON, 2nd phase) at about 2Hz
33 % ON, 3rd phase)

1.3.6 CONCEAL

The characters are displayed as spaces until the user chooses to make them appear.

1.3.7 INVERT

The characters are displayed as if the foreground and background colours had been exchanged. If FLASH is applied the polarity of the flashing clock is also inverted.

1.3.8 WINDOW/BOX

The 'full screen background' of the character positions becomes transparent, ie the video picture is displayed.

1.3.9 MARKED

The characters are marked for further action at the terminal, eg to be transferred to an output device.

1.3.10 PROTECTED

The character positions are protected against alteration, manipulation or erasure. The protection is valid for attributes as well as characters.

Protected character positions may only be overwritten by the use of a specific code or by the action of the clear screen command (CS), which deletes both the characters and the protection.

Protected character positions may be scrolled and therefore may disappear from the screen, because the protection is always related to the particular information on the screen.

Protected characters must not be obscured by enlarged characters.

1.3.11 Scrolling Area

A scrolling area is an area within the defined display area, within which the characters and associated attributes move in increments of one character position under the action of format effectors or specific controls. The procedure of scrolling is defined by two processes:

1. The designation of the screen area inside which a scroll operation is to be executed;

2. The execution of the scrolling action.

The border of a scrolling area must not be crossed by an enlarged character. The action of a double-height command in the serial mode on the bottom row causes a scroll up, the writing of a double-height character in the parallel mode on the top row causes a scroll down.

The scrolling operation is applied to full rows; the scrolling of parts of rows is for further study.

The use of APA and APH allows the active positions to be moved across the borders of a scrolling area. The addressing of APA is relative to the defined display area and is independent of scrolling.

1.3.12 Colour Tables

Extension of the colour range is accomplished by providing a number of colour tables of 8 colours each. At a given instant only one table may be 'in use'; this table can be invoked using colour table controls.

The fixed repertoire of colours (plus transparent) may be extended with redefinable colours loaded via the 'Define COLOUR' VPDE.

1.4 Rules for the Action of the SIZE Attribute

In the parallel mode the application of the double-height control causes characters to be printed so that they occupy the character positions on the current row and on the row immediately above. The origin of the characters for subsequent attribute modification is the upper character position. The double-height and double-size controls are inactive on the top row of the defined display area. The writing of double-height character in the parallel mode on the top row of a scrolling area causes a scroll down.

In the serial mode double-height characters extend downwards, the origin of the character is the upper character position. The double-height and the double-size controls are inactive on the bottom row of the defined display area. The action of a double-height command in the serial mode on the bottom row of a scrolling area causes a scroll up.

Double-width characters extend to the right, the origin of the character is the left-hand character position. Alternate characters on the row are displayed.

The whole of an enlarged character is displayed with the attributes that apply to the origin of the character.

Parts of enlarged characters are not displayed, the double-width and double-size controls are inactive in the last character positions of a row.

Attributes set at obscured character positions do not take effect if they would break any of the above rules.

The application of a double-width attribute or a double-size attribute causes the cursor to move two character positions forward in both the serial and parallel modes when a character is written. The action of cursor control functions such as APB, APF and spacing attribute controls is not affected.

The application of one SIZE attribute terminates the action of any other SIZE attribute.

NOTE

Attention is drawn to the fact that the retention of characters obscured by enlarged characters and the overwriting of parts of enlarged characters is for further study.

1.5 Defaults

1.5.1 Default Initiation

The occurrence of certain events causes the default settings to be set. Table 2 below shows the events leading to the setting of a certain default. This is independent of the current mode of operation of the terminal.

TABLE 2 DEFAULT INITIATION

Default set	Full screen back-ground and device controls	Defined display area back-ground	Defined display area fore-ground	Cursor parallel attributes	Markers	Colour table and Scrolling area
Session start	x	x	x 3	x	x	x
CS		x	x 3	x	x	x
APA				x		
APH				x		
Full Row Attribute controls					x 1	
CAN					x 2	

1 For the related attribute in the row.

2 All the markers on the right of the active position up to the end of the row.

3 Default graphic character is SPACE.

1.5.2 Default Setting of Attributes

Full screen attributes are used as default conditions for defined display area attributes.

TABLE 3 DEFAULT SETTING OF ATTRIBUTES

Full screen background	Defined display area background and cursor	Defined display area foreground and cursor	Markers	Colour table	Scrolling
Black	Transparent	Colour white Normal size Unboxed Not concealed Steady Non-lined Not inverted Non-protected Unmarked	Off	Colour Table 1	Implicit Scrolling active No defined scrolling area

1.5.3 Default Colour Look-up Tables

CLUT1	addresses	colours	0 - 7	of	the	colour map
CLUT2	"	"	8 - 15	"	"	"
CLUT3	"	"	16 - 23	"	"	"
CLUT4	"	"	24 - 31	"	"	"

1.5.4 Default Colour Map

See Table 4

1.5.5 Default Device Controls

Cursor - off
Recording Device - stop
Hard Copy Device - stop
Auxillary Device - off
Display Device - on

1.5.6 Default Graphic Sets

G0 set - the primary set of graphic characters
G1 set - the second supplementary set of mosaic characters
G2 set - the supplementary set of graphic characters
G3 set - the third supplementary set of mosaic characters
L set - the first supplementary set of mosaic characters

FIGURE 1 COLOUR CONE

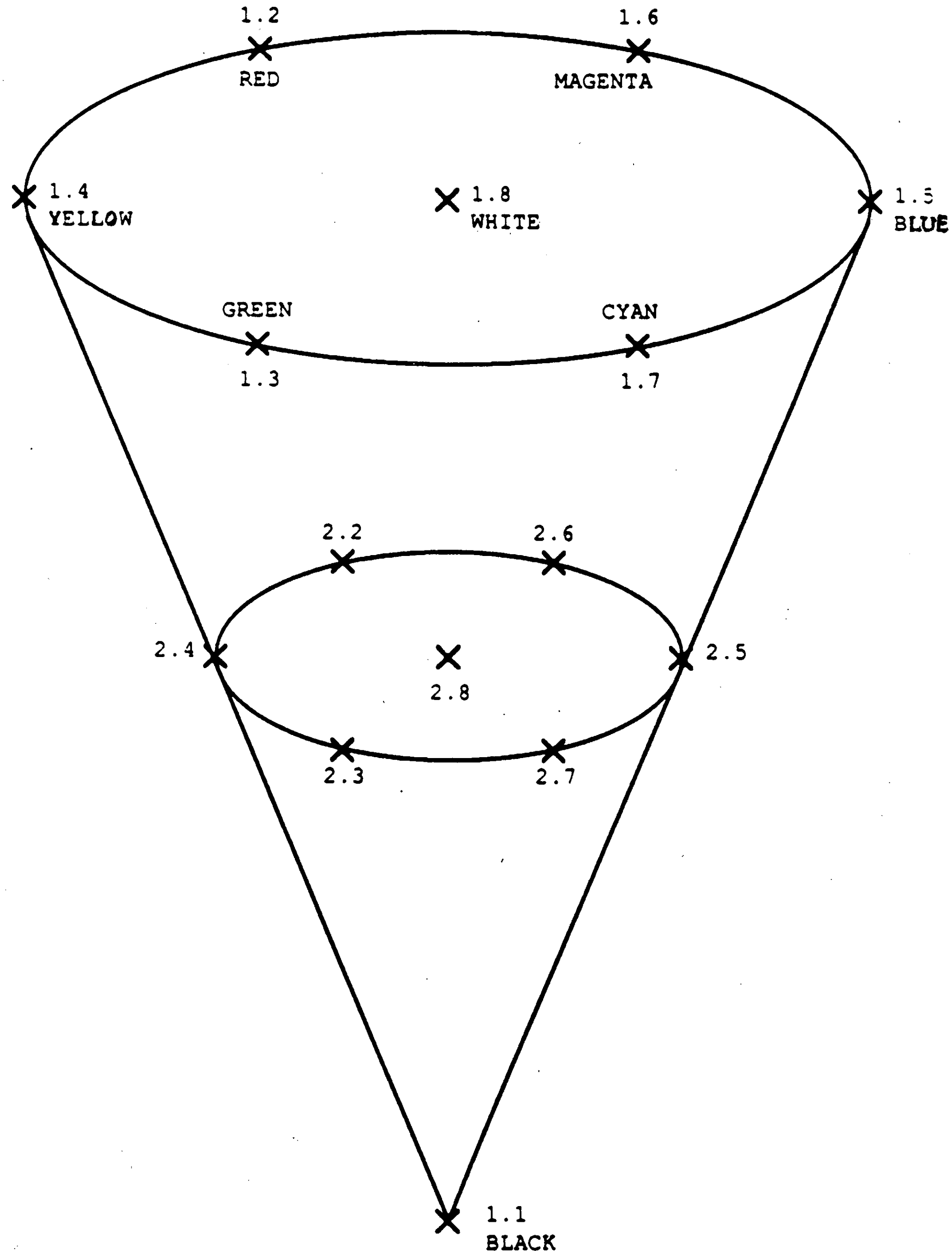


TABLE 4 RED GREEN AND BLUE COMPONENTS OF DEFAULT COLOUR MAP

Colour No		R	G	B	
0	BLACK	000000	000000	000000	
1	RED	111111	000000	000000	
2	GREEN	000000	111111	000000	
3	YELLOW	111111	111111	000000	
4	BLUE	000000	000000	111111	
5	MAGENTA	111111	000000	111111	
6	CYAN	000000	111111	111111	
7	WHITE	111111	111111	111111	
8	TRANSPARENT	--	--	--	*
9	REDUCED INTENSITY RED	011111	000000	000000	
10	" " GREEN	000000	011111	000000	
11	" " YELLOW	011111	011111	000000	
12	" " BLUE	000000	000000	011111	
13	" " MAGENTA	011111	000000	011111	
14	" " CYAN	000000	011111	011111	
15	GREY	011111	011111	011111	
16	BLACK	000000	000000	000000	
17	RED	111111	000000	000000	
18	GREEN	000000	111111	000000	
19	YELLOW	111111	111111	000000	
20	BLUE	000000	000000	111111	
21	MAGENTA	111111	000000	111111	
22	CYAN	000000	111111	111111	
23	WHITE	111111	111111	111111	
24	BLACK	000000	000000	000000	
25	RED	111111	000000	000000	
26	GREEN	000000	111111	000000	
27	YELLOW	111111	111111	000000	
28	BLUE	000000	000000	111111	
29	MAGENTA	111111	000000	111111	
30	CYAN	000000	111111	111111	
31	WHITE	111111	111111	111111	

NOTE

* If this entry (No 8) is defined as BLACK (as it is for default) it will be interpreted as TRANSPARENT.

2.0 REPERTORY

Alphamosaic presentation data is identified by the transmission of the ALPHAMOSAIC VPCE. The data following the VPCE may consist of any of the following repertoire of characters, format effectors, code extension controls, device controls or attribute controls.

2.1 Character Repertoire

The character repertoire consists of a fixed repertoire of alphanumeric characters, mosaic characters and line drawing characters. This fixed repertoire may be extended by the use of the DRCS option as described in part 4.

Characters of the fixed repertoire are identified according to the scheme described in Appendix 1.

2.1.1 Alphanumeric Characters

The alphanumeric repertoire consists of the fixed repertoire of 335 characters listed below.

Latin alphabetic characters

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	POS	SET	POS
LA01	a	small a	G0	6/1		
LA02	A	capital A	G0	4/1		
LA11	á	small a with acute accent	G2	4/2	G0	6/1
LA12	Á	capital A with acute accent	G2	4/2	G0	4/1
LA13	à	small a with grave accent	G2	4/1	G0	6/1
LA14	À	capital A with grave accent	G2	4/1	G0	4/1
LA15	â	small a with circumflex	G2	4/3	G0	6/1
LA16	Â	capital A with circumflex	G2	4/3	G0	4/1
LA17	ä	small a with diaeresis or umlaut	G2	4/8	G0	6/1
LA18	Ä	capital A with diaeresis or umlaut	G2	4/8	G0	4/1
LA19	ã	small a with tilde	G2	4/4	G0	6/1
LA20	Ã	capital A with tilde	G2	4/4	G0	4/1

ID	GRAPHIC	NAME OR DESCRIPTION	CODE			
			SET	POS	SET	POS
LA23	ă	small a with breve	G2	4/6	G0	6/1
LA24	Ă	capital A with breve	G2	4/6	G0	4/1
LA27	å	small a with ring	G2	4/10	G0	6/1
LA28	Å	capital A with ring	G2	4/10	G0	4/1
LA31	ā	small a with macron	G2	4/5	G0	6/1
LA32	Ā	capital A with macron	G2	4/5	G0	4/1
LA43	ą	small a with ogonek	G2	4/14	G0	6/1
LA44	Ą	capital A with ogonek	G2	4/14	G0	4/1
LA51	æ	small æ diphthong	G2	7/1		
LA52	Æ	capital Æ diphthong	G2	6/1		
LB01	b	small b	G0	6/2		
LB02	B	capital B	G0	4/2		
LC01	c	small c	G0	6/3		
LC02	C	capital C	G0	4/3		
LC11	ć	small c with acute accent	G2	4/2	G0	6/3
LC12	Ć	capital C with acute accent	G2	4/2	G0	4/3
LC15	ĉ	small c with circumflex	G2	4/3	G0	4/1
LC16	Ĉ	capital C with circumflex	G2	4/3	G0	4/1
LC21	č	small c with caron	G2	4/15	G0	6/3
LC22	Č	capital C with caron	G2	4/15	G0	4/3
LC29	ċ	small c with dot	G2	4/7	G0	6/3
LC30	Ċ	capital C with dot	G2	4/7	G0	4/3
LC41	ç	small c with cedilla	G2	4/11	G0	6/3
LC42	Ç	capital C with cedilla	G2	4/11	G0	4/3
LD01	d	small d	G0	6/4		
LD02	D	capital D	G0	4/4		

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS	CODE SET	CODE POS
LD21	ď or ď	small d with caron	G2	4/15	G0	6/4
LD22	Ď	capital D with caron	G2	4/15	G0	4/4
LD61	đ	small d with stroke	G2	7/2		
LD62	Đ	capital D with stroke, Icelandic eth	G2	6/2		
LD63	ð	small eth, Icelandic	G2	7/3		
LE01	e	small e	G0	6/5		
LE02	E	capital E	G0	4/5		
LE11	é	small e with acute accent	G2	4/2	G0	6/5
LE12	É	capital E with acute accent	G2	4/2	G0	4/5
LE13	è	small e with grave accent	G2	4/1	G0	6/5
LE14	È	capital E with grave accent	G2	4/1	G0	4/5
LE15	ê	small e with circumflex	G2	4/3	G0	6/5
LE16	Ê	capital E with circumflex	G2	4/3	G0	4/5
LE17	ë	small e with diaeresis or umlaut	G2	4/8	G0	6/5
LE18	Ë	capital E with diaeresis or umlaut	G2	4/8	G0	4/5
LE21	ě	small e with caron	G2	4/15	G0	6/5
LE22	Ě	capital E with caron	G2	4/15	G0	4/5
LE29	ė	small e with dot	G2	4/7	G0	6/5
LE30	Ė	capital E with dot	G2	4/7	G0	4/5
LE31	ē	small e with macron	G2	4/5	G0	6/5
LE32	Ē	capital E with macron	G2	4/5	G0	4/5
LE43	ę	small e with ogonek	G2	4/14	G0	6/5
LE44	Ę	capital E with ogonek	G2	4/14	G0	4/5
LF01	f	small f	G0	6/6		
LF02	F	capital F	G0	4/6		
LG01	g	small g	G0	6/7		

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS	CODE SET	CODE POS
LG02	G	capital G	G0	4/7		
LG11	ǵ	small g with acute accent	G2	4/2	G0	6/7
LG15	ĝ	small g with circumflex	G2	4/3	G0	6/7
LG16	Ĝ	capital G with circumflex	G2	4/3	G0	4/7
LG23	ğ	small g with breve	G2	4/6	G0	6/7
LG24	Ğ	capital G with breve	G2	4/6	G0	4/7
LG29	ġ	small g with dot	G2	4/7	G0	6/7
LG30	Ġ	capital G with dot	G2	4/7	G0	4/7
LG42	Ģ	capital G with cedilla	G2	4/11	G0	4/7
LH01	h	small h	G0	6/8		
LH02	H	capital H	G0	4/8		
LH15	ĥ	small h with circumflex	G2	4/3	G0	6/8
LH16	Ĥ	capital H with circumflex	G2	4/3	G0	4/8
LH61	ħ	small h with stroke	G2	7/4		
LH62	Ħ	capital H with stroke	G2	6/4		
LI01	i	small i	G0	6/9		
LI02	I	capital I	G0	4/9		
LI11	í	small i with acute accent	G2	4/2	G0	6/9
LI12	Í	capital I with acute accent	G2	4/2	G0	4/9
LI13	ì	small i with grave accent	G2	4/1	G0	6/9
LI14	Ì	capital I with grave accent	G2	4/1	G0	4/9
LI15	î	small i with circumflex	G2	4/3	G0	6/9
LI16	Î	capital I with circumflex	G2	4/3	G0	4/9
LI17	ï	small i with diaeresis or umlaut	G2	4/8	G0	6/9
LI18	Ï	capital I with diaeresis or umlaut	G2	4/8	G0	4/9
LI19	ĩ	small i with tilde	G2	4/4	G0	6/9

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	POS	SET	POS
LI20	Ĩ	capital I with tilde	G2	4/4	G0	4/9
LI30	I	capital I with dot	G2	4/7	G0	4/9
LI31	ī	small i with macron	G2	4/5	G0	6/9
LI32	Ī	capital I with macron	G2	4/5	G0	4/9
LI43	i	small i with ogonek	G2	4/14	G0	6/9
LI44	I	capital I with ogonek	G2	4/14	G0	4/9
LI51	ij	small ij ligature	G2	7/6		
LI52	IJ	capital IJ ligature	G2	6/6		
LI61	ı	small i without dot	G2	7/5		
LJ01	j	small j	G0	6/10		
LJ02	J	capital J	G0	4/10		
LJ15	ĵ	small j with circumflex	G2	4/3	G0	6/10
LJ16	Ĵ	capital J with circumflex	G2	4/3	G0	4/10
LK01	k	small k	G0	6/11		
LK02	K	capital K	G0	4/11		
LK41	ķ	small k with cedilla	G2	4/11	G0	6/11
LK42	Ķ	capital K with cedilla	G2	4/11	G0	4/11
LK61	ĸ	small k, Greenlandic	G2	7/0		
LL01	l	small l	G0	6/12		
LL02	L	capital L	G0	4/12		
LL11	ĺ	small l with acute accent	G2	4/2	G0	6/12
LL12	Ĺ	capital L with acute accent	G2	4/2	G0	4/12
LL21	ľ or ľ	small l with caron	G2	4/15	G0	6/12
LL22	Ľ or Ľ	capital L with caron	G2	4/15	G0	4/12
LL41	ļ	small l with cedilla	G2	4/11	G0	6/12
LL42	Ļ	capital L with cedilla	G2	4/11	G0	4/12

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS	CODE SET	CODE POS
LL61	ł	small l with stroke	G2	7/8		
LL62	Ł	capital L with stroke	G2	6/8		
LL63	ŀ	small l with middle dot	G2	7/7		
LL64	Ŀ	capital L with middle dot	G2	6/7		
LM01	m	small m	G0	6/13		
LM02	M	capital M	G0	4/13		
LN01	n	small n	G0	6/14		
LN02	N	capital N	G0	4/14		
LN11	ń	small n with acute accent	G2	4/2	G0	6/14
LN12	Ń	capital N with acute accent	G2	4/2	G0	4/14
LN19	ñ	small n with tilde	G2	4/4	G0	6/14
LN20	Ñ	capital N with tilde	G2	4/4	G0	4/14
LN21	ň	small n with caron	G2	4/15	G0	6/14
LN22	Ň	capital N with caron	G2	4/15	G0	4/14
LN41	ņ	small n with cedilla	G2	4/11	G0	6/14
LN42	Ņ	capital N with cedilla	G2	4/11	G0	6/14
LN61	ŋ	small eng, Lapp	G2	7/14		
LN62	Ŋ	capital eng, Lapp	G2	6/14		
LN63	ŉ	small n with apostrophe	G2	6/15		
LO01	o	small o	G0	6/15		
LO02	O	capital O	G0	4/15		
LO11	ó	small o with acute accent	G2	4/2	G0	6/15
LO12	Ó	capital O with acute accent	G2	4/2	G0	4/15
LO13	ò	small o with grave accent	G2	4/1	G0	6/15
LO14	Ò	capital O with grave accent	G2	4/1	G0	4/15
LO15	ô	small o with circumflex	G2	4/3	G0	6/15

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	POS	SET	POS
LO16	Ô	capital O with circumflex	G2	4/3	G0	4/15
LO17	ö	small o with diaeresis or umlaut	G2	4/8	G0	6/15
LO18	Ö	capital O with diaeresis or umlaut	G2	4/8	G0	4/15
LO19	õ	small o with tilde	G2	4/4	G0	6/15
LO20	Õ	capital O with tilde	G2	4/4	G0	4/15
LO25	ő	small o with double acute accent	G2	4/13	G0	6/15
LO26	Ő	capital O with double acute accent	G2	4/13	G0	4/15
LO31	ō	small o with macron	G2	4/5	G0	6/15
LO32	Ō	capital O with macron	G2	4/5	G0	4/15
LO51	œ	small œ ligature	G2	7/10		
LO52	Œ	capital Œ ligature	G2	6/10		
LO61	ø	small o with slash	G2	7/9		
LO62	Ø	capital O with slash	G2	6/9		
LP01	p	small p	G0	7/0		
LP02	P	capital P	G0	5/0		
LQ01	q	small q	G0	7/1		
LQ02	Q	capital Q	G0	5/1		
LR01	r	small r	G0	7/2		
LR02	R	capital R	G0	5/2		
LR11	ŕ	small r with acute accent	G2	4/2	G0	7/2
LR12	Ŕ	capital R with acute accent	G2	4/2	G0	5/2
LR21	ř	small r with caron	G2	4/15	G0	7/2
LR22	Ř	capital R with caron	G2	4/15	G0	5/2
LR41	ŗ	small r with cedilla	G2	4/11	G0	7/2
LR42	Ŗ	capital R with cedilla	G2	4/11	G0	5/2
LS01	s	small s	G0	7/3		

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	POS	SET	POS
LS02	S	capital S	G0	5/3		
LS11	ś	small s with acute accent	G2	4/2	G0	7/3
LS12	Ś	capital S with acute accent	G2	4/2	G0	5/3
LS15	ŝ	small s with circumflex	G2	4/3	G0	7/3
LS16	Ŝ	capital S with circumflex	G2	4/3	G0	5/3
LS21	š	small s with caron	G2	4/15	G0	7/3
LS22	Š	capital S with caron	G2	4/15	G0	5/3
LS41	ş	small s with cedilla	G2	4/11	G0	7/3
LS42	Ş	capital S with cedilla	G2	4/11	G0	5/3
LS61	ß	small sharp s, German	G2	7/11		
LT01	t	small t	G0	7/4		
LT02	T	capital T	G0	5/4		
LT21	ť	small t with caron	G2	4/15	G0	7/4
LT22	Ť	capital T with caron	G2	4/15	G0	5/4
LT41	ţ	small t with cedilla	G2	4/11	G0	7/4
LT42	Ţ	capital T with cedilla	G2	4/11	G0	5/4
LT61	ŧ	small t with stroke	G2	7/13		
LT62	Ŧ	capital T with stroke	G2	6/13		
LT63	þ	small thorn, Icelandic	G2	7/12		
LT64	Þ	capital thorn, Icelandic	G2	6/12		
LU01	u	small u	G0	7/5		
LU02	U	capital U	G0	5/5		
LU11	ú	small u with acute accent	G2	4/2	G0	7/5
LU12	Ú	capital U with acute accent	G2	4/2	G0	5/5
LU13	ù	small u with grave accent	G2	4/1	G0	7/5
LU14	Ù	capital U with grave accent	G2	4/1	G0	5/5

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS	CODE SET	CODE POS
LU15	û	small u with circumflex	G2	4/3	G0	7/5
LU16	Û	capital U with circumflex	G2	4/3	G0	5/5
LU17	ü	small u with diaeresis or umlaut	G2	4/8	G0	7/5
LU18	Ü	capital U with diaeresis or umlaut	G2	4/8	G0	5/5
LU19	ũ	small u with tilde	G2	4/4	G0	7/5
LU20	Ũ	capital U with tilde	G2	4/4	G0	5/5
LU23	ŭ	small u with breve	G2	4/6	G0	7/5
LU24	Ŭ	capital U with breve	G2	4/6	G0	5/5
LU25	ű	small u with double acute accent	G2	4/13	G0	7/5
LU26	Ű	capital U with double acute accent	G2	4/13	G0	5/5
LU27	ů	small u with ring	G2	4/10	G0	7/5
LU28	Ů	capital U with ring	G2	4/10	G0	5/5
LU31	ū	small u with macron	G2	4/5	G0	7/5
LU32	Ū	capital U with macron	G2	4/5	G0	5/5
LU43	ų	small u with ogonek	G2	4/14	G0	7/5
LU44	Ų	capital U with ogonek	G2	4/14	G0	5/5
LV01	v	small v	G0	7/6		
LV02	V	capital V	G0	5/6		
LW01	w	small w	G0	7/7		
LW02	W	capital W	G0	5/7		
LW15	ŵ	small w with circumflex	G2	4/3	G0	7/7
LW16	Ŵ	capital W with circumflex	G2	4/3	G0	5/7
LX01	x	small x	G0	7/8		
LX02	X	capital X	G0	5/8		
LY01	y	small y	G0	7/9		
LY02	Y	capital Y	G0	5/9		

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS	SET	POS
LY11	ý	small y with acute accent	G2	4/2	G0	7/9
LY12	Ý	capital Y with acute accent	G2	4/2	G0	5/9
LY15	ŷ	small y with circumflex	G2	4/3	G0	7/9
LY16	Ŷ	capital Y with circumflex	G2	4/3	G0	5/9
LY17	ÿ	small y with diaeresis or umlaut	G2	4/8	G0	7/9
LY18	Ÿ	capital Y with diaeresis or umlaut	G2	4/8	G0	5/9
LZ01	z	small z	G0	7/10		
LZ02	Z	capital Z	G0	5/10		
LZ11	ź	small z with acute accent	G2	4/2	G0	7/10
LZ12	Ź	capital Z with acute accent	G2	4/2	G0	5/10
LZ21	ž	small z with caron	G2	4/15	G0	7/10
LZ22	Ž	capital Z with caron	G2	4/15	G0	5/10
LZ29	ż	small z with dot	G2	4/7	G0	7/10
LZ30	Ż	capital Z with dot	G2	4/7	G0	5/10

Non-alphabetic characters

Decimal digits

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS
ND01	1	digit 1	G0	3/1
ND02	2	digit 2	G0	3/2
ND03	3	digit 3	G0	3/3
ND04	4	digit 4	G0	3/4
ND05	5	digit 5	G0	3/5
ND06	6	digit 6	G0	3/6
ND07	7	digit 7	G0	3/7
ND08	8	digit 8	G0	3/8
ND09	9	digit 9	G0	3/9
ND10	0	digit 0	G0	3/0

Currency signs

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS
SC01	¤	general currency sign	G2 (G0	2/3 2/4)
SC02	£	pound sign	G2	2/3
SC03	$	dollar sign	G2	2/4
SC04	¢	cent sign	G2	2/2
SC05	¥	yen sign	G2	2/5

Punctuation marks

ID	GRAPHIC	NAME OR DESCRIPTION	CODE	
			SET	POS
SP01		space	Gx	2/0
SP02	!	exclamation mark	G0	2/1
SP03	¡	inverted exclamation mark	G2	2/1
SP04	"	quotation mark	G0	2/2
SP05	'	apostrophe	G0	2/7
SP06	(	left parenthesis	G0	2/8
SP07	)	right parenthesis	G0	2/9
SP08	,	comma	G0	2/12
SP10	-	hyphen or minus sign	G0	2/13
SP11	.	full stop, period	G0	2/14
SP12	/	solidus	G0	2/15
SP13	:	colon	G0	3/10
SP14	;	semicolon	G0	3/11
SP15	?	question mark	G0	3/15
SP16	¿	inverted question mark	G2	3/15
SP17	<	angle quotation mark left	G2	2/11
SP18	>	angle quotation mark right	G2	3/11
SP19	‘	single quotation mark left	G2	2/9
SP20	’	single quotation mark right	G2	3/9
SP21	“	double quotation mark left	G2	2/10
SP22	”	double quotation mark right	G2	3/10

Note. In videotex 'quotation mark', 'apostrophe' and 'comma' are independent characters which cannot have the meaning of diacritical marks.

Arithmetic signs

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS
SA01	+	plus sign	G0	2/11
SA02	±	plus/minus sign	G2	3/1
SA03	<	less-than sign	G0	3/12
SA04	=	equals sign	G0	3/13
SA05	>	greater-than sign	G0	3/14
SA06	÷	divide sign	G2	3/8
SA07	×	multiply sign	G2	3/4

Subscripts and Superscripts

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS
NS01	^{1}	superscript 1	G2	5/1
NS02	^{2}	superscript 2	G2	3/2
NS03	^{3}	superscript 3	G2	3/3

Fractions

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS
NF01	½	fraction one half	G2	3/13
NF04	¼	fraction one quarter	G2	3/12
NF05	¾	fraction three quarters	G2	3/14

Miscellaneous symbols

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS
SM01	#	number sign	G0 G2	2/3 2/6
SM02	%	percent sign	G0	2/5
SM03	&	ampersand	G0	2/6
SM04	*	star	G0	2/10
SM05	@	commercial at	G0	4/0
SM06	[	left square bracket	G0	5/11
SM07	\	reverse solidus	G0	5/12
SM08	]	right square bracket	G0	5/13
SM11	{	left curly bracket	G0	7/11
SM12	―	central horizontal bar jointive	G2	5/0
SM13	|	central vertical bar jointive	G0	7/12
SM14	}	right curly bracket	G0	7/13
SM17	µ	micro sign	G2	3/5
SM18	Ω	ohm sign	G2	6/0
SM19	°	degree sign	G2	3/0
SM20	º	ordinal indicator, masculine	G2	6/11
SM21	ª	ordinal indicator, feminine	G2	6/3
SM24	§	section sign	G2	2/7
SM25	¶	paragraph sign, pilcrow	G2	3/6
SM26	·	middle dot	G2	3/7
SM30	←	leftward arrow	G2	2/12
SM31	→	rightward arrow	G2	2/14
SM32	↑	upward arrow	G2	2/13
SM33	↓	downward arrow	G2	2/15
SM34	■	delete	Gx	7/15
SM35	®	registered mark symbol	G2	5/2

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS
SM36	©	copyright symbol	G2	5/3
SM37	™	trade mark symbol	G2	5/4
SM38	♪	musical symbol	G2	5/5
SM39	⅛	one eighth	G2	5/12
SM40	⅜	three eighths	G2	5/13
SM41	⅝	five eighths	G2	5/14
SM42	⅞	seven eighths	G2	5/15
SM43	^	arrowhead upwards	G0	5/14
SM44	`	upper reverse solidus	G0	6/0
SM45	|	left vertical bar jointive	G1	4/14
SM46	|	right vertical bar jointive	G1	5/14
SM47	‾	upper bar	G0	7/14
SM48	_	lower bar	G0	5/15

Diacritical marks (as displayed when used in conjunction with SPACE)

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS
SD11	´	acute accent	G2	4/2
SD13	`	grave accent	G2	4/1
SD15	^	circumflex	G2	4/3
SD17	¨	umlaut or diaeresis	G2	4/8
SD19	~	tilde	G2	4/4
SD21	ˇ	caron	G2	4/15
SD23	˘	breve	G2	4/6
SD25	˝	double acute accent	G2	4/13
SD27	˚	ring	G2	4/10
SD29	˙	dot	G2	4/7
SD31	¯	macron	G2	4/5
SD41	¸	cedilla	G2	4/11
SD43	˛	ogonek	G2	4/14

2.1.2 Mosaic Graphics

In addition to the alphanumeric repertoire it is possible to make simple pictures using characters from the mosaic graphic repertoire defined below. Each mosaic character completely fills the area of a character cell on the screen.

The repertoire consists of:

63 graphics (block mosaic characters) consisting of a combination of six rectangular elements;

48 graphics (smoothed mosaic characters) where the shapes are bounded by lines between corners of six rectangular elements;

8 graphics (smoothed mosaic characters) where the a shapes are bounded by lines between the corners of the character cell and the centre of the character cell;

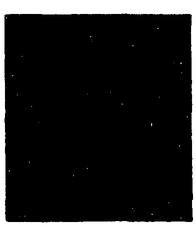
24 line drawing graphics;

4 jointive arrows;

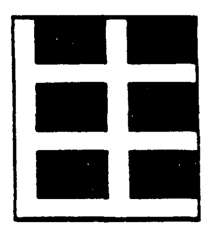
4 miscellaneous drawing graphics including one graphic with a dot-pattern where approximately 40% of the character cell area has the foreground colour and the remaining area has the background colour.

The shaded areas in the representations of the mosaic character are to be displayed in the defined foreground colour and the unshaded areas are to be displayed in the defined background colour.

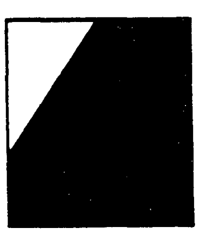
The mosaic graphic character may be displayed in two fonts, 'contiguous' and 'seperated' as shown in the examples below. For the 'seperated' font the 'seperation space' is on the left and lower edge of the blocks. The actual dimension of the space is not defined.



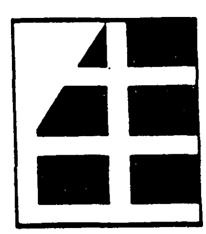
Contiguous Block Mosaic



Seperated Block Mosaic

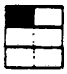
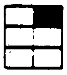
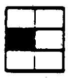
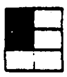
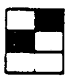
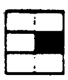
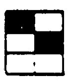
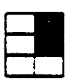
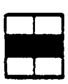


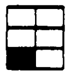
Contiguos Smoothed Mosaic

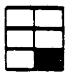
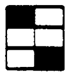
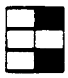
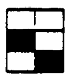
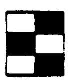
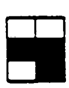
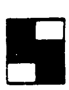


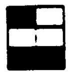
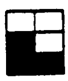
Seperated Smoothed Mosaic

Block Mosaic Graphics

ID	GRAPHIC	CODE SET	CODE POS
MG01		G1	2/1
MG02		G1	2/2
MG03		G1	2/3
MG04		G1	2/4
MG05		G1	2/5
MG06		G1	2/6
MG07		G1	2/7
MG08		G1	2/8
MG09		G1	2/9
MG10		G1	2/10
MG11		G1	2/11
MG12		G1	2/12
MG13		G1	2/13
MG14		G1	2/14
MG15		G1	2/15

ID	GRAPHIC	CODE SET	CODE POS
MG16		G1	3/0
MG17		G1	3/1
MG18		G1	3/2
MG19		G1	3/3
MG20		G1	3/4
MG21		G1	3/5
MG22		G1	3/6
MG23		G1	3/7
MG24		G1	3/8
MG25		G1	3/9
MG26		G1	3/10
MG27		G1	3/11
MG28		G1	3/12
MG29		G1	3/13
MG30		G1	3/14
MG31		G1	3/15

ID	GRAPHIC	CODE SET	CODE POS
MG32		G1	6/0
MG33		G1	6/1
MG34		G1	6/2
MG35		G1	6/3
MG36		G1	6/4
MG37		G1	6/5
MG38		G1	6/6
MG39		G1	6/7
MG40		G1	6/8
MG41		G1	6/9
MG42		G1	6/10
MG43		G1	6/11
MG44		G1	6/12
MG45		G1	6/13
MG46		G1	6/14
MG47		G1	6/15

ID	GRAPHIC	CODE SET	POS
MG48		G1	7/0
MG49		G1	7/1
MG50		G1	7/2
MG51		G1	7/3
MG52		G1	7/4
MG53		G1	7/5
MG54		G1	7/6
MG55		G1	7/7
MG56		G1	7/8
MG57		G1	7/9
MG58		G1	7/10
MG59		G1	7/11
MG60		G1	7/12
MG61		G1	7/13
MG62		G1	7/14
MG63		G1	5/15

Smoothed Mosaic Graphics

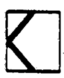
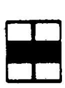
ID	GRAPHIC	CODE SET	CODE POS
SG01		G1	4/0
SG02		G1	4/1
SG03		G1	4/2
SG04		G1	4/3
SG05		G1	4/4
SG06		G1	4/5
SG07		G1	4/6
SG08		G1	4/7
SG09		G1	4/8
SG10		G1	4/9
SG11		G1	4/10
SG12		G1	4/11
SG13		G1	4/12
SG14		G1	4/13

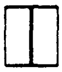
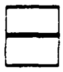
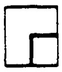
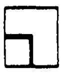
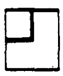
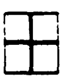
ID	GRAPHIC	CODE SET	CODE POS
SG15		G1	5/13
SG16		G1	5/12
SG17		G1	5/11
SG18		G1	5/10
SG19		G1	5/9
SG20		G1	5/8
SG21		G1	5/7
SG22		G1	5/6
SG23		G1	5/5
SG24		G1	5/4
SG25		G1	5/3
SG26		G1	5/2
SG27		G1	5/1
SG28		G1	5/0

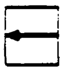
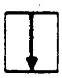
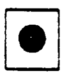
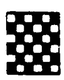
ID	GRAPHIC	CODE SET	CODE POS
SG29		G3	6/0
SG30		G3	6/1
SG31		G3	6/2
SG32		G3	6/3
SG33		G3	6/4
SG34		G3	6/5
SG35		G3	6/6
SG36		G3	6/7
SG37		G3	6/8
SG38		G3	6/9
SG39		G3	6/10
SG40		G3	6/11
SG41		G3	6/12
SG42		G3	6/13

ID	GRAPHIC	CODE SET	CODE POS
SG43		G3	7/0
SG44		G3	7/1
SG45		G3	7/2
SG46		G3	7/3
SG47		G3	7/4
SG48		G3	7/5
SG49		G3	7/6
SG50		G3	7/7
SG51		G3	7/8
SG52		G3	7/9
SG53		G3	7/10
SG54		G3	7/11
SG55		G3	7/12
SG56		G3	7/13

Line Drawing Graphics

ID	GRAPHIC	CODE SET	CODE POS
DG01		G3	4/0
DG02		G3	4/1
DG03		G3	4/2
DG04		G3	4/3
DG05		G3	4/4
DG06		G3	4/5
DG07		G3	4/6
DG08		G3	4/7
DG09		G3	4/8
DG10		G3	4/9
DG11		G3	4/10
DG12		G3	4/11
DG13		G3	4/12

ID	GRAPHIC	CODE SET	CODE POS
DG14		G3	5/0
DG15		G3	5/1
DG16		G3	5/2
DG17		G3	5/3
DG18		G3	5/4
DG19		G3	5/5
DG20		G3	5/6
DG21		G3	5/7
DG22		G3	5/8
DG23		G3	5/9
DG24		G3	5/10

ID	GRAPHIC	CODE SET	CODE POS
DG25		G3	5/11
DG26		G3	5/12
DG27		G3	5/13
DG28		G3	5/14
DG29		G3	4/13
DG30		G3	4/14
DG31		G3	4/15
DG32		G1	4/15

2.2 Format Effector Repertoire

Abbreviation	Name and Definition
APA	ACTIVE POSITION ADDRESSING A format effector which causes the active position to move to a defined position on the screen in accordance with parameters following.
APB	ACTIVE POSITION BACK A format effector which causes the active position to move backwards one character position on the same row. APB on the first character position on the row moves the active position to the last character position of the preceding row. APB on the first character position of the first row moves the active position to the last character position of the last row in the defined display area.
APF	ACTIVE POSITION FORWARD A format effector which causes the active position to move forward to the next character position on the same row. At the last position on he row, this format effector moves the active position to the first character position on the following row. APF on the last character of the last row moves the active position to the first character position on the first row in the defined display area.
APD	ACTIVE POSITION DOWN A format effector which causes the active position to move to the equivalent character position on the following row. APD on the last row moves the active position to the equivalent character position on the first row in the defined display area.
APU	ACTIVE POSITION UP A format effector which causes the active position to move to the equivalent character position on the preceding row. APU on the first row moves the active position to the equivalent character position on the last row in the defined display area.
APR	ACTIVE POSITION RETURN A format effector which causes the active position to move to the first character position of the same row.
APH	ACTIVE POSITION HOME A format effector which causes the active position to be moved to the first character position of the first row of the defined display area.
CS	CLEAR SCREEN A format effector which causes the active position to be moved to the first character position of the first row in the defined display area and causes all character positions to be filled with spaces with all attributes set to the default conditions as decribed in Section 1.5.

CAN CANCEL
A control function which fills all character positions from the active position to the end of the row inclusive with spaces. The active position is then returned to its previous location.

SP SPACE
A format effector which advances the active position one character-width forward on the same row. (It is also regarded as a graphic character with no foreground. In those systems which define an explicit background, the space copies the background colour into the active position and moves the active position one character-width forward. If used in conjunction with the inversion attribute it copies the foreground colour into the active position and moves the active position one character width forward.) SPACE on the last character position of a row moves the active position to the first position of the next row. SPACE on the last character position of a frame moves the active position to the first character position of the frame.

RPT REPEAT
A format effector which causes the immediatly preceding complete graphic character, including SPACE and DEL, to be displayed a number of times as defined by a parameter.

DEL DELETE
In the mosaic graphics mode the use of DEL moves the active position one space forward, with the vacated space obliterated with the foreground colour. Attributes (double-height, colour, etc.) remain in force.

In the alphanumeric mode the use of DEL moves the active position one space forward and displays the DELETE graphics character in the vacated position.

DEL on the last character position of a row moves the active position to the first position of the next row. DEL on the last character position of a frame moves the active position to the first character position of the frame.

HMS HOLD MOSAIC
When the mosaic graphics set is activated this function causes the last received mosaic graphic character to be displayed in its previously defined rendition when a serial attribute control function is transmitted.

RMS RELEASE MOSAIC
Causes the action of HOLD MOSAIC to be stopped.

2.3 Attribute Control Repertoire

An attribute control causes the desired display attribute to be applied to the display graphic characters referenced. Four types of attribute control are defined:

Full screen attribute controls -
These affect all the the character positions on the screen, except the full screen background colour control which affects the full screen background layer.

Full row attribute controls -
These affect all the character positions on the defined row, except the full row background colour control which affects the defined row of the full screen background layer.

Serial attribute controls
These apply between markers on a row. They apply from the location of the active position at the time they are received to the end of the row or until a contradictory marker is reached. Each of the control functions of this repertoire causes the active position to be advanced one character width forwards; the position thus vacated is to be generally displayed as a SPACE. The control HOLD MOSAICS may modify this display. Combinations of control functions may be applied at one character location.

Parallel attribute controls -
These are the property of the active position and move with it under the action of format effectors or spacing display characters (including space). They apply to the displayed characters subsequently received until the attributes are changed by relevant controls including certain format effectors (CS, APA, APH). They also apply to spacing display characters (including space) inserted by control commands.

2.3.1 FOREGROUND COLOUR Controls

(a) Full screen and Full row controls

The following controls are available as either full screen or full row controls.

The FOREGROUND COLOUR may be set to any one of the eight colours of the currently invoked colour table using the following controls.

Abbreviation	Name and Definition
BKF	BLACK FOREGROUND Invokes 1st colour of the colour table
RDF	RED FOREGROUND Invokes 2nd colour of the colour table
GRF	GREEN FOREGROUND Invokes 3rd colour of the colour table
YLF	YELLOW FOREGROUND Invokes 4th colour of the colour table
BLF	BLUE FOREGROUND Invokes 5th colour of the colour table
MGF	MAGENTA FOREGROUND Invokes 6th colour of the colour table
CNF	CYAN FOREGROUND Invokes 7th colour of the colour table
WHF	WHITE FOREGROUND Invokes 8th colour of the colour table

(b) Serial controls

The FOREGROUND COLOUR may be set to any one of the eight colours of the currently invoked colour table. The same controls are also used to shift into or out of the first mosaic set (the L set).

The following 'alpha' foreground colour controls cause the appropriate foreground colour to be applied and a locking shift from the first mosaic set (the L set) back to the previously invoked G set.

Abbreviation	Name and Definition
ABK	ALPHA BLACK Invokes 1st colour of the colour table
ANR	ALPHA RED Invokes 2nd colour of the colour table
ANG	ALPHA GREEN Invokes 3rd colour of the colour table
ANY	ALPHA YELLOW Invokes 4th colour of the colour table
ANB	ALPHA BLUE Invokes 5th colour of the colour table
ANM	ALPHA MAGENTA Invokes 6th colour of the colour table
ANC	ALPHA CYAN Invokes 7th colour of the colour table
ANW	ALPHA WHITE Invokes 8th colour of the colour table

The following 'mosaic' foreground colour controls cause the appropriate foreground colour to be applied and a locking shift to the first mosaic set (the L set). A shift back from the first mosaic set (the L set) to the previously invoked G set is implicit at the start of each row.

Abbreviation	Name and Definition
MBK	MOSAIC BLACK Invokes 1st colour of the colour table
MSR	MOSAIC RED Invokes 2nd colour of the colour table
MSG	MOSAIC GREEN Invokes 3rd colour of the colour table
MSY	MOSAIC YELLOW Invokes 4th colour of the colour table
MSB	MOSAIC BLUE Invokes 5th colour of the colour table
MSM	MOSAIC MAGENTA Invokes 6th colour of the colour table
MSC	MOSAIC CYAN Invokes 7th colour of the colour table
MSW	MOSAIC WHITE Invokes 8th colour of the colour table

(c) **Parallel controls**

The foreground colour may be set to any one of the eight colours of the currently invoked colour table using the following controls.

Abbreviation	Name and Definition
BKF	BLACK FOREGROUND Invokes 1st colour of the colour table
RDF	RED FOREGROUND Invokes 2nd colour of the colour table
GRF	GREEN FOREGROUND Invokes 3rd colour of the colour table
YLF	YELLOW FOREGROUND Invokes 4th colour of the colour table
BLF	BLUE FOREGROUND Invokes 5th colour of the colour table
MGF	MAGENTA FOREGROUND Invokes 6th colour of the colour table
CNF	CYAN FOREGROUND Invokes 7th colour of the colour table
WHF	WHITE FOREGROUND Invokes 8th colour of the colour table

2.3.2 BACKGROUND COLOUR Controls

(a) Full screen and Full row controls

The following controls are available as either full screen or full row controls. They cause the full screen background layer to adopt one of the eight colours of the currently invoked colour table or transparency.

Abbreviation	Name and Definition
BKB	BLACK BACKGROUND Invokes 1st colour of the colour table
RDB	RED BACKGROUND Invokes 2nd colour of the colour table
GRB	GREEN BACKGROUND Invokes 3rd colour of the colour table
YLB	YELLOW BACKGROUND Invokes 4th colour of the colour table
BLB	BLUE BACKGROUND Invokes 5th colour of the colour table
MGB	MAGENTA BACKGROUND Invokes 6th colour of the colour table
CNB	CYAN BACKGROUND Invokes 7th colour of the colour table
WHB	WHITE BACKGROUND Invokes 8th colour of the colour table
TRB	TRANSPARENT BACKGROUND Invokes transparent background (the underlying video picture)

(b) Serial controls

The following controls affect the character background.

Abbreviation	Name and Definition
NBD	NEW BACKGROUND Causes the BACKGROUND COLOUR to adopt the current foreground colour as defined by previous colour controls. The foreground colour is unchanged.
BBD	BLACK BACKGROUND Causes the BACKGROUND COLOUR to invoke the first colour of the colour table.

(c) Parallel controls

The following controls cause the character background layer (layer b) to adopt one of the eight colours of the currently invoked colour table or transparency.

Abbreviation	Name and Definition
BKB	BLACK BACKGROUND Invokes 1st colour of the colour table
RDB	RED BACKGROUND Invokes 2nd colour of the colour table
GRB	GREEN BACKGROUND Invokes 3rd colour of the colour table
YLB	YELLOW BACKGROUND Invokes 4th colour of the colour table
BLB	BLUE BACKGROUND Invokes 5th colour of the colour table
MGB	MAGENTA BACKGROUND Invokes 6th colour of the colour table
CNB	CYAN BACKGROUND Invokes 7th colour of the colour table
WHB	WHITE BACKGROUND Invokes 8th colour of the colour table
TRB	TRANSPARENT BACKGROUND Invokes Transparent background

2.3.3 LINING Controls

The following controls are available as full screen, full row, serial or parallel controls.

Abbreviation	Name and Definition
STL	START LINING Applies the LINED attribute
SPL	STOP LINING Stops the application of the LINED attribute

2.3.4 SIZE Controls

(a) Full screen and full row controls

The following control is available either as a full screen or full row control.

Abbreviation	Name and Definition
NSZ	NORMAL-SIZE Applies the NORMAL-SIZE attribute

(b) Serial and parallel controls

The following controls are available in both the serial and parallel modes.

Abbreviation	Name and Definition
NSZ	NORMAL-SIZE Applies the NORMAL-SIZE attribute
DBH	DOUBLE-HEIGHT Applies the DOUBLE-HEIGHT attribute
DBW	DOUBLE-WIDTH Applies the DOUBLE-WIDTH attribute
DBS	DOUBLE-SIZE Applies the DOUBLE-SIZE attribute

NOTE

As described in section 1.4, the action of the DOUBLE-HEIGHT control is different in the serial and parallel modes.

2.3.5 **FLASH Controls**

(a) **Full screen and full row controls**

The following controls are available as either full screen or full row controls.

Abbreviation	Name and Definition
FSH	FLASH Applies the normal (50%) FLASH attribute
STD	STEADY Cancels the application of any FLASH attribute

(b) Serial and parallel controls

The following controls are available in both the serial and parallel modes.

State controls:

Abbreviation	Name and Definition
FSH	FLASH Applies the normal flash state
STD	STEADY Cancels the application of any flash attribute
IVF	INVERTED FLASH Applies the inverted flash state
RIF	REDUCED INTENSITY FLASH (flash between colour tables) Applies the reduced intensity flash state

Rate controls:

Abbreviation	Name and Definition
FF1	FAST FLASH 1 Applies the 1st phase of three-phase flash
FF2	FAST FLASH 2 Applies the 2nd phase of three-phase flash
FF3	FAST FLASH 3 Applies the 3rd phase of three-phase flash

NOTE

The application of any of the state controls defaults to the normal 50% 1Hz rate.

Abbreviation	Name and Definition
ICF	INCREMENT FLASH Three-phase fast flash is applied to characters so that the phase is sequentially changed for every character (enlarged characters count as single characters) in a string of three adjacent characters to produce an apparent movement to the right.
DCF	DECREMENT FLASH Three-phase fast flash is applied to characters so that the phase is sequentially changed for every character (enlarged characters count as single characters) in a string of three adjacent characters to produce an apparent movement to the left.

2.3.6 CONCEAL Controls

(a) Full screen and full row attributes

The following controls are available as either full screen or full row controls.

Abbreviation	Name and Definition
CDY	CONCEAL DISPLAY Applies the CONCEAL attribute
STC	STOP CONCEAL Causes the concealed characters to be revealed

(b) Serial and parallel controls

The following controls are available in both the serial and parallel modes.

Abbreviation	Name and Definition
CDY	CONCEAL DISPLAY Applies the CONCEAL attribute
STC	STOP CONCEAL Stops the application of the CONCEAL attribute

At full screen and full row level there is no need for a 'non-concealed' control; the 'stop conceal' control is interpreted as 'reveal' and also resets the character positions addressed to the 'not concealed' state.

2.3.7 INVERT Controls

(a) Full screen, full row and parallel controls

The following controls are available as either full screen, full row or parallel controls.

Abbreviation	Name and Definition
IPO	INVERTED POLARITY Applies the INVERT attribute
NPO	NORMAL POLARITY Stops the application of the INVERT attribute

(b) Serial controls - none.

2.3.8 WINDOW/BOX Controls

The following controls are available as either full screen, full row, serial or parallel controls.

Full screen controls affect the whole of the defined display area.

Full row controls affect whole rows within the defined display area.

Serial and parallel controls affect parts of rows and individual characters respectively.

Abbreviation	Name and Definition
SBX	START BOX Applies the WINDOW/BOX attribute
EBX	END BOX Stops the application of the WINDOW/BOX attribute

2.3.9 MARKING Controls

The following controls are available as either full screen, full row, serial or parallel controls.

Full screen controls affect the whole of the defined display area.

Full row controls affect whole rows within the defined display area.

Serial and parallel controls affect parts of rows and individual characters respectively.

Abbreviation	Name and Definition
MMS	MARKED MODE START Applies the MARKED attribute
MMT	MARKED MODE STOP Stops the application of the MARKED attribute

2.3.10 PROTECTING Controls

(a) Full screen, full row, serial and parallel controls
The following controls are available as either full screen, full row, serial or parallel controls.

Full screen controls affect the whole of the defined display area.

Full row controls affect whole rows within the defined display area.

Serial and parallel controls affect parts of rows and individual characters respectively.

Abbreviation	Name and Definition
PMS	PROTECTED MODE START Applies the PROTECTED attribute
PMC	PROTECTED MODE CANCEL Cancels (removes) the PROTECTED attribute (allows overwriting)

(b) Additional serial and parallel controls
The following controls may be applied in either the serial or parallel mode.

Abbreviation	Name and Definition
PMI	PROTECTED MODE IDLE Stops the application of the PROTECTED attribute

2.3.11 Definition of Scrolling Area

Abbreviation	Name and Definition
CSA	CREATE SCROLLING AREA Creates a scrolling area
DSA	DELETE SCROLLING AREA Deletes all or part of a scrolling area

2.3.12 Execution of Scrolling

(a) Implicit scrolling

Scroll up
APF, or the printing of a character or spacing attribute control on the last character position, or APD in the lowest row of the selected part of the screen, copies the contents of row i to i-1. The contents of the uppermost row of the selected part of the screen will be discarded. The lowest row of the selected part of the screen is filled with spaces (2/0) but the off-screen row-defined attributes remain unchanged. Thus the lowest row will show spaces in the row-defined background colour.

Scroll down
APB on the first character position, or APU in the uppermost row of the selected part of the screen, copies the contents of row i to row i + 1. The contents of the lowest row of the selected part of the screen will be discarded. The uppermost row of the selected part of the screen is filled with spaces (2/0) but the off-screen row-defined attributes remain unchanged. Thus the uppermost row will show spaces in the row-defined background colour.

Abbreviation	Name and Definition
DIS	DEACTIVATE IMPLICIT SCROLLING This deactivates the implicit scrolling, allowing the active position in to move across the border of a scrolling area
AIS	ACTIVATE IMPLICIT SCROLLING This restores the implicit scrolling effect of format effectors

(b) Explicit scrolling

These controls affect the scrolling area.

Abbreviation	Name and Definition
SCU	SCROLL UP This causes a scrolling up of the designated scrolling area
SCD	SCROLL DOWN This causes a scroll down of the designated area. The active position does not move relative to the defined display area.

2.3.13 Colour Table Controls

The following controls invoke the selected colour table.

Abbreviation	Name and Definition
CT1	COLOUR TABLE 1 Invokes 1st colour table
CT2	COLOUR TABLE 2 Invokes 2nd colour table
CT3	COLOUR TABLE 3 Invokes 3rd colour table
CT4	COLOUR TABLE 4 Invokes 4th colour table

These controls are Locking controls and are reset by a contradictory control or clear screen (CS). Invoking a colour table has no effect on an attribute until that attribute is changed.

2.4 Device Control Function Repertoire

2.4.1 Cursor Controls

Abbreviation	Name and Definition
CON	CURSOR ON A device control function which causes the active position to be indicated
COF	CURSOR OFF A device control function which terminates the action of CON

2.4.2 Recording Device Controls

Abbreviation	Name and Definition
RDS	RECORDING DEVICE START Causes the associated recording device to start recording data subsequently received by the terminal
RDT	RECORDING DEVICE STOP Causes the associated recording device to stop
RDW	RECORDING DEVICE WAIT Causes the associated recording device to wait

2.4.3 Hard Copy Device Controls

Abbreviation	Name and Definition
HCS	HARD COPY START Causes the associated hard copy device to start copying data subsequently received by the terminal
HCT	HARD COPY STOP Causes the associated hard copy device to stop
HCW	HARD COPY WAIT Causes the associated hard copy device to wait

2.4.4 Display Device Controls

Abbreviation	Name and Definition
DDO	DISPLAY DEVICE ON Data subsequently received by the terminal is displayed
DDF	DISPLAY DEVICE OFF Data subsequently received by the terminal is not displayed

2.4.5 Auxiliary Device Controls

Abbreviation	Name and Definition
ADO	AUXILIARY DEVICE ON Data subsequently received by the terminal is passed to the auxiliary device
ADF	AUXILIARY DEVICE OFF Data subsequently received by the terminal is not passed to the auxiliary device

2.4.6 Miscellaneous Device Controls

Abbreviation	Name and Definition
EBU	EMPTY BUFFER Causes the contents of the terminal buffer to be transmitted to the line

3.0 CODING STRUCTURE

The coding structure defined allows for both 7- and 8-bit coding of presentation data.

Control functions are coded using primary and supplementary control sets and by using combinations of control codes and following parameters.

Characters are coded in five character-sets.
In the 7-bit environment only one of these character-sets may be invoked into the 'in use' code table.
In the 8-bit environment two of these character-sets may be invoked into the 'in use' code table.
In order to invoke the character-sets, locking shift functions are required for all sets (G0, G1, G2, G3 and L). To enable access to the sets not invoked, single shift functions are also incorporated.

The designation of the sets from a library to the G0, G1, G2 and G3 sets is, in accordance with ISO 2022, the same for both the 8-bit and 7-bit environment.

3.1 Code Extension and Invocation

3.1.1 Common Code Extension Control Functions

Abbreviation	Name and Definition
ESC	ESCAPE A control character that is used to provide additional control functions other than transmission control functions and that alters the meaning of a limited number of contiguously following bit combinations.
CSI	CONTROL SEQUENCE INTRODUCER A control character that is used to provide additional control functions other than transmission control functions and that alters the meaning of a limited number of contiguously following bit combinations.

3.1.2 Invocation Functions (7-bit Environment)

Abbreviation	Name and Definition
SO	SHIFT OUT Invokes the G1 set into columns 2-7 of the code table
SI	SHIFT IN Invokes the G0 set into columns 2-7 of the code table
LS2	LOCKING SHIFT 2 Invokes the G2 set into columns 2-7 of the code table
LS3	LOCKING SHIFT 3 Invokes the G3 set into columns 2-7 of the code table
SS2	SINGLE SHIFT 2 Invokes a single character from the G2 set
SS3	SINGLE SHIFT 3 Invokes a single character from the G3 set

NOTE

L-set activation is by serial C1-controls 5/0 to 5/7

L-set deactivation is by any one of the following:

serial C1-controls 4/0 to 4/7
invocation of parallel C1-set
entering new line
invocation of a G-set into columns 2 to 7 of the code table

3.1.3 Invocation Functions (8-bit Environment)

Abbreviation	Name and Definition
LS0	LOCKING SHIFT 0 Invokes the G0 set into columns 2-7 of the code table
LS1	LOCKING SHIFT 1 Invokes the G1 set into columns 2-7 of the code table
LS1R	LOCKING SHIFT 1 RIGHT Invokes the G1 set into columns 10-15 of the code table
LS2	LOCKING SHIFT 2 Invokes the G2 set into columns 2-7 of the code table
LS2R	LOCKING SHIFT 2 RIGHT Invokes the G2 set into columns 10-15 of the code table
LS3	LOCKING SHIFT 3 Invokes the G3 set into columns 2-7 of the code table
LS3R	LOCKING SHIFT 3 RIGHT Invokes the G3 set into columns 10-15 of the code table
SS2	SINGLE SHIFT 2 Invokes a single character from the G2 set
SS3	SINGLE SHIFT 3 Invokes a single character from the G3 set

NOTE

L-set activation is by serial C1-controls 5/0 to 5/7

L-set deactivation is by any one of the following:

serial C1-controls 4/0 to 4/7
invocation of parallel C1-set
entering new line
invocation of a G-set into columns 2 to 7 of the code table

3.1.4 Default Code Sets

The primary control function set is designated the CO set. Either of the supplementary control function sets may be designated as the default Cl set.

The primary set of characters is designated the GO set. The supplementary set of alphanumeric characters is designated the G2 set.

The first supplementary set of mosaic characters is designated the L set and is invoked by controls in the serial Cl set.

The second supplementary set of mosaic characters is designated the Gl set.

The third supplementary set of mosaic characters is designated the G3 set. In the 8-bit environment the GO set is invoked into columns 2-7 and the G2 set is invoked into columns 10-14 of the 'in use' code table.

3.2 The Primary Control Function Set - (Table 5)

This set contains two types of elements: those which consist of a single bit combination and those which are used in conjunction with following parameters (RPT and APA).

3.2.1 Parameters For Format Effectors

Repeat RPT (char)

The parameter (char) indicates the number of repetitions of the immediatly preceding graphic character. The representation is in binary form by the 6 least significant bits of the parameter which is taken from columns 4 to 7. The character itself is not included in the count. This function does not apply to control characters.

- Active Position Address APA (char) (char)

A control function with a two or four character parameter. All the characters are within the range 4/0 * to 7/14, and they represent respectively the row address and the column address in binary form, with 6 useful bits (bit 6 being the most significant bit) of the first character to be displayed.

The first character received shall be displayed on the designated character location of the addressed row.

The default address range of the defined display area is 1 to 24 vertically and 1 to 40 horizontally. The loacation addressed by APA, 4/1, 4/1 (or APA 4/0, 4/1, 4/0, 4/1 if the format exceeds ether 63 rows or 63 columns) is the top left-hand location of the defined display area.

If the format exceeds either 63 rows or 63 columns then the relevant parameter, ie the row or the column address, is coded as a two byte sequence with 12 useful bits, the first byte carrying the most significant bits.

* Addressing row 0 is for private use.

TABLE 5 THE PRIMARY CONTROL FUNCTION SET (DEFAULT CO SET)

				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	NUL							
0	0	0	1	1		CON						
0	0	1	0	2		RPT						
0	0	1	1	3								
0	1	0	0	4		COF						
0	1	0	1	5								
0	1	1	0	6								
0	1	1	1	7								
1	0	0	0	8	APB	CAN						
1	0	0	1	9	APF	SS2						
1	0	1	0	10	APD							
1	0	1	1	11	APU	ESC						
1	1	0	0	12	CS							
1	1	0	1	13	APR	SS3						
1	1	1	0	14	S0	APH						
1	1	1	1	15	SI	APA[1]						

(1) This code is also used for the Unit Separator (US) control.

(2) Empty positions in the table denote bit combinations reserved for future standardization and shall not be used.

(3) Shaded code positions are reserved for G sets and shall not be used for control characters.

(2604)

3.3 The Supplementary Control Function Sets

Two supplementary control function sets are defined; one for applying 'serial' attribute controls and one for applying 'parallel' attribute controls.

In the 7-bit environment individual characters of these sets are represented by two-bit combinations of the form ESC, Fe where Fe lies in the range 4/0 to 5/15.

In the 8-bit environment individual characters of these sets are represented by the combinations in the range 8/0 to 9/15.

3.3.1 THE SERIAL SUPPLEMENTARY CONTROL FUNCTION SET - (Table 6)

This set is designated by the sequence ESC 2/2 4/0.

3.3.2 THE PARALLEL SUPPLEMENTARY CONTROL FUNCTION SET - (Table 7)

This set is designated by the sequence ESC 2/2 4/1.

TABLE 6 THE SERIAL SUPPLEMENTARY CONTROL FUNCTION SET

				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					ABK	MBK		
0	0	0	1	1					ANR	MSR		
0	0	1	0	2					ANG	MSG		
0	0	1	1	3					ANY	MSY		
0	1	0	0	4					ANB	MSB		
0	1	0	1	5					ANM	MSM		
0	1	1	0	6					ANC	MSC		
0	1	1	1	7					ANW	MSW		
1	0	0	0	8					FSH	CDY		
1	0	0	1	9					STD	SPL		
1	0	1	0	10					EBX	STL		
1	0	1	1	11					SBX	CSI		
1	1	0	0	12					NSZ	BBO		
1	1	0	1	13					DBH	NBD		
1	1	1	0	14					DBW	HMS		
1	1	1	1	15					DBS	RMS		

TABLE 7 THE PARALLEL SUPPLEMENTARY CONTROL FUNCTION SET

b4	b3	b2	b1	b7 / b6 / b5	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
					0	1	2	3	4	5	6	7
0	0	0	0	0					BKF	BKB		
0	0	0	1	1					RDF	RDB		
0	0	1	0	2					GRF	GRB		
0	0	1	1	3					YLF	YLB		
0	1	0	0	4					BLF	BLB		
0	1	0	1	5					MGF	MGB		
0	1	1	0	6					CNF	CNB		
0	1	1	1	7					WHF	WHB		
1	0	0	0	8					FSH	CDY		
1	0	0	1	9					STD	SPL		
1	0	1	0	10					EBX	STL		
1	0	1	1	11					SBX	CSI		
1	1	0	0	12					NSZ	NPO		
1	1	0	1	13					DBH	IPO		
1	1	1	0	14					DBW	TRB		
1	1	1	1	15					DBS	STC		

3.4 The Coding of Graphic Characters

3.4.1 Code Sets

Five code sets are used to encode the graphic characters. These are:

1. The primary set of characters - Table 8
   This consists of the most frequently used alphanumeric characters and punctuation marks. The bit combination 2/0 is used for SPACE and 7/15 is used for DELETE.

2. The supplementary set of alphanumeric characters - Table 9
   This set contains three types of characters:

   - 4/0 to 4/15
     Diacritical marks which are used in combination with the letters of the basic Latin alphabet in the primary set to constitute the coded representations of characters with diacritical marks. Each of these characters acts as a modifier indicating that the immediately following letter is to be transformed.

   - 6/0 to 7/14
     Alphabetic characters which are used in addition to the basic Latin alphabet in the primary set and which are not composed by combining diacritical marks and basic letters.

   - 2/1 to 3/15
     Non-alphabetic characters which are used in addition to those in the primary set.

3. The first supplementary set of mosaic characters - Table 10
   This set consists of 63 block mosaic characters and 32 text characters, the representation of which is identical to that of the characters of columns 4 and 5 of the primary set of characters.

4. The second supplementary set of mosaic characters - Table 11
   This set consists of 63 block mosaic characters, 28 smoothed mosaic characters, two line vertical bars and one shading character.

5. The third supplementary set of mosaic characters - Table 12
   This set consists of 28 smoothed mosaic characters, 24 line drawing characters and 7 miscellaneous characters.

TABLE 8 THE PRIMARY SET OF GRAPHIC CHARACTERS (DEFAULT GO SET)

				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	@	P	`[2]	p
0	0	0	1	1			!	1	A	Q	a	q
0	0	1	0	2			"	2	B	R	b	r
0	0	1	1	3			#[2]	3	C	S	c	s
0	1	0	0	4			¤[2]	4	D	T	d	t
0	1	0	1	5			%	5	E	U	e	u
0	1	1	0	6			&	6	F	V	f	v
0	1	1	1	7			'	7	G	W	g	w
1	0	0	0	8			(	8	H	X	h	x
1	0	0	1	9			)	9	I	Y	i	y
1	0	1	0	10			*	:	J	Z	j	z
1	0	1	1	11			+	;	K	[[2]	k	{[2]
1	1	0	0	12			,	<	L	\[2]	l	|[2]
1	1	0	1	13			–	=	M	][2]	m	}[2]
1	1	1	0	14			.	>	N	^[2]	n	‾[2]
1	1	1	1	15			/	?	O	# / _[1]	o	

(1) The characters allocated to positions 5/15 may be displayed either as _ (LOWER BAR) or # (SQUARE) to represent the terminator function required by existing Videotex services.

(2) The representation of these characters is not guaranteed in international communication and may be replaced by national application oriented variants.

TABLE 9 THE SUPPLEMENTARY SET OF GRAPHIC CHARACTERS (DEFAULT G2 SET)

				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				°		—	Ω	ĸ
0	0	0	1	1			¡	±	`	¹	Æ	æ
0	0	1	0	2			¢	²	´	®	Đ	đ
0	0	1	1	3			£	³	^	©	ª	ð
0	1	0	0	4			$	×	~	™	Ħ	ħ
0	1	0	1	5			¥	µ	¯	♪		ı
0	1	1	0	6			#	¶	˘		Ĳ	ĳ
0	1	1	1	7			§	·	˙		Ŀ	ŀ
1	0	0	0	8			¤	÷	¨		Ł	ł
1	0	0	1	9			‘	’			Ø	ø
1	0	1	0	10			“	”	˚		Œ	œ
1	0	1	1	11			«	»	¸		º	ß
1	1	0	0	12			←	¼		⅛	Þ	þ
1	1	0	1	13			↑	½	˝	⅜	Ŧ	ŧ
1	1	1	0	14			→	¾	˛	⅝	Ŋ	ŋ
1	1	1	1	15			↓	¿	ˇ	⅞	ŉ	

(1) 4/8 is diaeresis and is used for compatibility with other text communication services which may need to distinguish between umlaut and diaeresis.

(2) Empty positions in the table denote bit combinations reserved for future standardization and shall not be used.

TABLE 10 THE FIRST SUPPLEMENTARY SET OF MOSAIC CHARACTERS (L SET)

b4	b3	b2	b1	b7 / b6 / b5	0 / 0 / 0	0 / 0 / 1	0 / 1 / 0	0 / 1 / 1	1 / 0 / 0	1 / 0 / 1	1 / 1 / 0	1 / 1 / 1
					0	1	2	3	4	5		
0	0	0	0	0					@	P		
0	0	0	1	1					A	Q		
0	0	1	0	2					B	R		
0	0	1	1	3					C	S		
0	1	0	0	4					D	T		
0	1	0	1	5					E	U		
0	1	1	0	6					F	V		
0	1	1	1	7					G	W		
1	0	0	0	8					H	X		
1	0	0	1	9					I	Y		
1	0	1	0	10					J	Z		
1	0	1	1	11					K	[(2)		
1	1	0	0	12					L	\ (2)		
1	1	0	1	13					M	] (2)		
1	1	1	0	14					N	^ (2)		
1	1	1	1	15					O	# / _ (1)		

(1) The characters allocated to positions 5/15 may be displayed either as _ (LOWER BAR) or # (SQUARE) to represent the terminator function required by existing Videotex services.

(2) The representation of these characters is not guaranteed in international communication and may be replaced by national application oriented variants.

TABLE 11 THE SECOND SUPPLEMENTARY SET OF MOSAIC CHARACTERS (DEFAULT G1 SET)

				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								

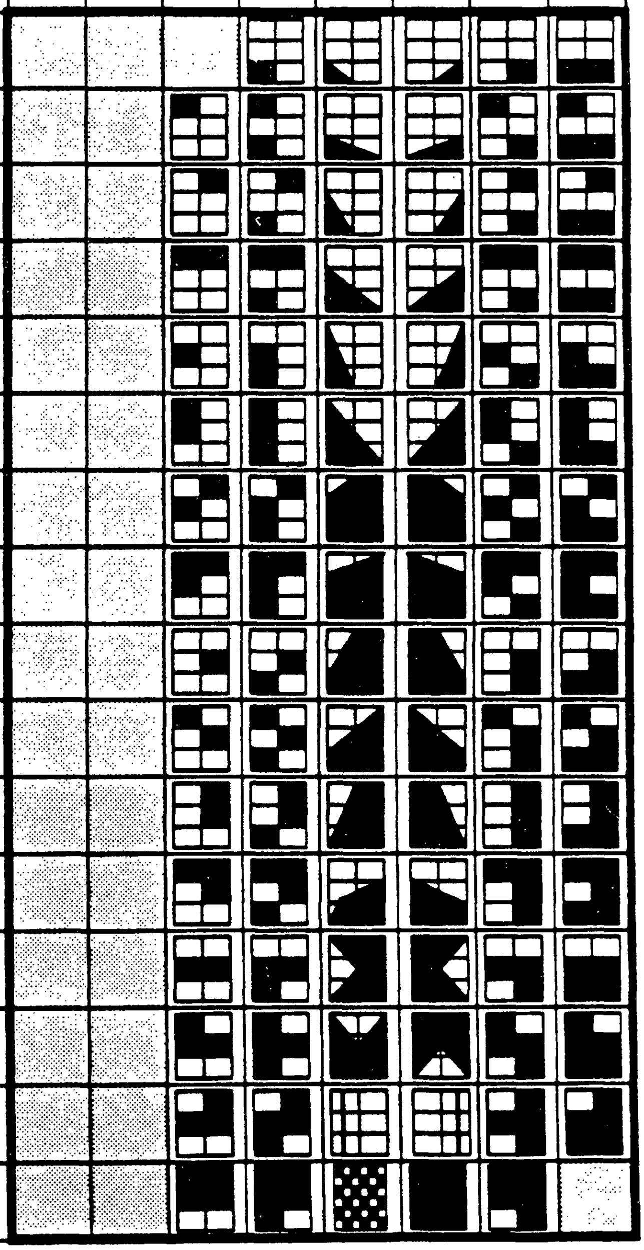
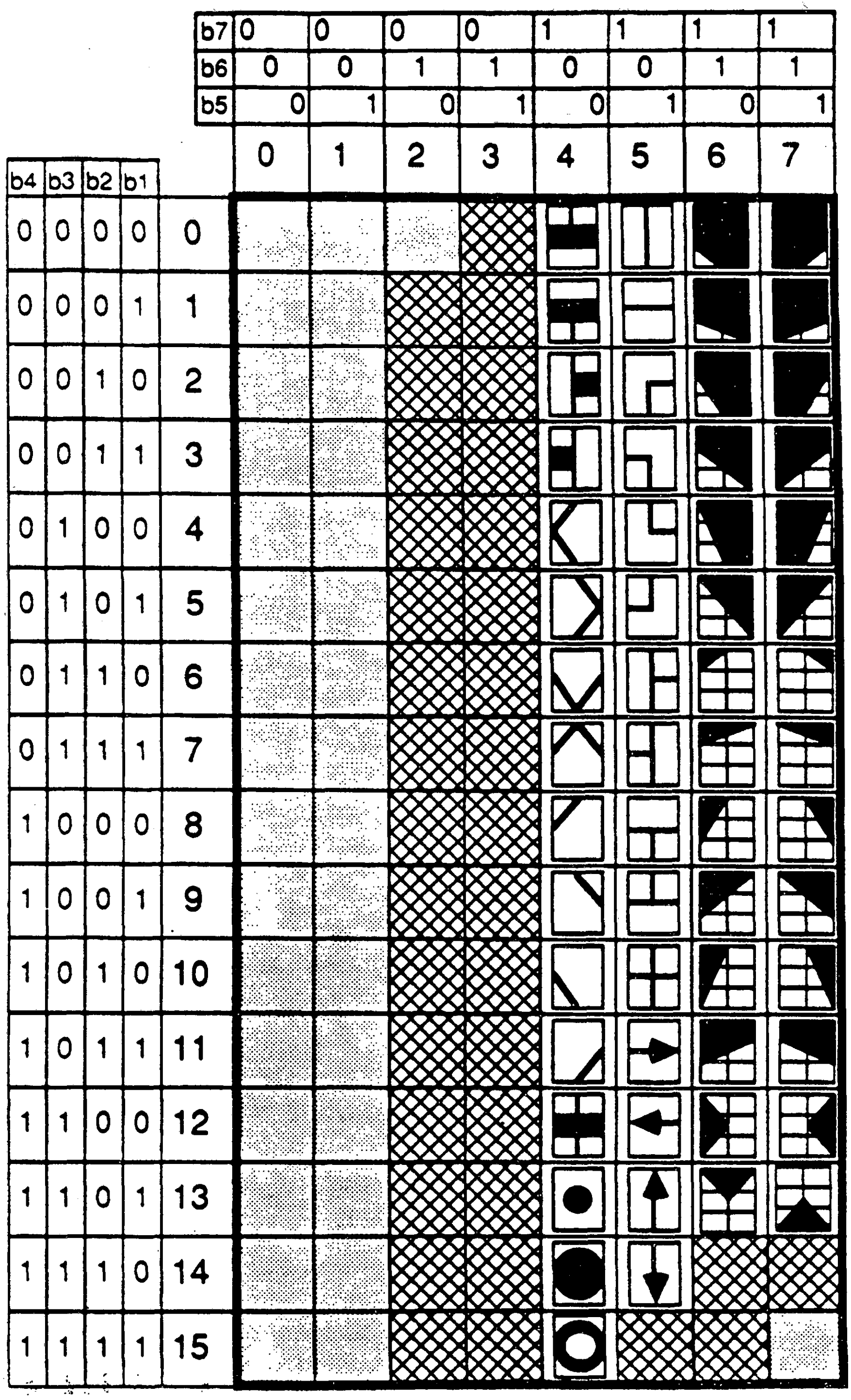


TABLE 12 THE THIRD SUPPLEMENTARY SET OF MOSAIC CHARACTERS (DEFAULT G3 SET)



(1) Empty positions in the table denote bit combinations reserved for future standardization and shall not be used.

3.4.2 The Coding of Characters with Diacritical Marks

Each of these characters is represented by a sequence of two bit-combinations. The first part of this sequence consists of a bit-combination in the range 4/0 to 4/15 from the supplementary set representing a diacritical mark. The second part consists of a bit-combination in the range 4/1 to 5/10 or 6/1 to 7/10 from the primary set representing a basic Latin letter or space. The diacritical marks are shown in column 4 of Table 9 and the basic Latin letters are shown in Table 8.

NOTE

If a diacritical mark is used in combination with a basic character such that the resulting character is not within the repertoire the terminal will display at least the basic character.

3.4.3 Designation of Graphic Sets

ESC	2/8	4/0	Primary set of graphic characters (G0)	to G0
ESC	2/9	4/0	:	to G1
ESC	2/10	4/0	:	to G2
ESC	2/11	4/0	:	to G3
ESC	2/8	6/3	Secondary supplementary set of mosaic characters (G1)	to G0
ESC	2/9	6/3	:	to G1
ESC	2/10	6/3	:	to G2
ESC	2/11	6/3	:	to G3
ESC	2/8	6/2	Supplementary set of graphic characters (G2)	to G0
ESC	2/9	6/2	:	to G1
ESC	2/10	6/2	:	to G2
ESC	2/11	6/2	:	to G3
ESC	2/8	6/4	Third supplementary set of mosaic characters (G3)	to G0
ESC	2/9	6/4	:	to G1
ESC	2/10	6/4	:	to G2
ESC	2/11	6/4	:	to G3

Note: The default position is shown in brackets.

3.5 Supplementary Attribute and Qualified Area Controls

3.5.1 Serial Control STOP CONCEAL

Abbreviation	Name and Coding	
STC	STOP CONCEAL	CSI 4/2

3.5.2 Full Screen and Full Row Attributes

The attributes:

FOREGROUND COLOUR

BACKGROUND COLOUR

LINED

SIZE

FLASH

CONCEAL

INVERT

WINDOW/BOX

are coded as four-character escape sequences of the form:

ESC 2/3 2/0 (Fe) for full screen attributes;

ESC 2/3 2/1 (Fe) for full row attributes;

where Fe is the attribute control character from the parallel C1 set in the 7 bit environment

ie Fe is 4/1 for Red foreground

3.5.3 Marking Controls

Abbreviation	Name and Coding	
MMS	MARKED MODE START	
	Full screen control	CSI 3/0 5/3
	Full row control	CSI 3/1 5/3
	Serial or parallel control	CSI 3/2 5/3
MMT	MARKED MODE STOP	
	Full screen control	CSI 3/0 5/4
	Full row control	CSI 3/1 5/4
	Serial or parallel control	CSI 3/2 5/4

3.5.4 Protecting Controls

Abbreviation	Name and Coding	
PMS	PROTECTED MODE START	
	Full screen control	CSI 3/0 5/0
	Full row control	CSI 3/1 5/0
	Serial or parallel control	CSI 3/2 5/0
PMC	PROTECTED MODE CANCEL	
	Full screen control	CSI 3/0 5/1
	Full row control	CSI 3/1 5/1
	Serial or parallel control	CSI 3/2 5/1
PMI	PROTECTED MODE IDLE	
	Serial or parallel control	CSI 3/2 5/2

The currently invoked C1 set indicates whether the above controls for MARKED and PROTECTED should be interpreted as serial or parallel controls.

3.5.5 Definition of a Scrolling Area

Similar CSI sequences are used for CREATE SCROLLING AREA and DELETE SCROLLING AREA; only the final characters are different.

CSI <URH> <URT> <URU> 3/11 <LRH> <LRT> <LRU> <F>

URH	hundreds value of the upper row
URT	tens value of the upper row
URU	units value of the upper row
LRH	hundreds value of the lower row
LRT	tens value of the lower row
LRU	units value of the lower row

These values are coded from column 3 of the code table. Leading zeros may be omitted.

F : 5/5 for CREATE SCROLLING AREA
5/6 for DELETE SCROLLING AREA

The action of scrolling is initiated as described in sections 2.3.12 and 3.6 of Part 1.

3.5.6 Colour Table Controls

The coding of the colour table invocation controls is as follows:

Abbreviation	Name and Coding	
CT1	COLOUR TABLE 1	CSI 3/0 4/0
CT2	COLOUR TABLE 2	CSI 3/1 4/0
CT3	COLOUR TABLE 3	CSI 3/2 4/0
CT4	COLOUR TABLE 4	CSI 3/3 4/0

3.5.7 Additional FLASH Controls

The additional FLASH controls are coded as follows:

Abbreviation	Name and Coding	
IVF	INVERTED FLASH	CSI 3/0 4/1
RIF	REDUCED INTENSITY FLASH	CSI 3/1 4/1
FF1	FAST FLASH 1	CSI 3/2 4/1
FF2	FAST FLASH 2	CSI 3/3 4/1
FF3	FAST FLASH 3	CSI 3/4 4/1
ICF	INCREMENT FLASH	CSI 3/5 4/1
DCF	DECREMENT FLASH	CSI 3/6 4/1

3.6 Device Controls

3.6.1 Cursor Controls

See primary control function set, Part 1, Section 3.2.

3.6.2 Supplementary Device Controls

Abbreviation	Name and Coding			
RDW	RECORDING DEVICE WAIT	ESC	3/5	
RDS	RECORDING DEVICE START	ESC	3/6	
RDT	RECORDING DEVICE STOP	ESC	3/7	
HCW	HARD COPY WAIT	ESC	3/8	
HCS	HARD COPY START	ESC	3/9	
HCT	HARD COPY STOP	ESC	3/10	
DDO	DISPLAY DEVICE ON	ESC	3/12	
DDF	DISPLAY DEVICE OFF	ESC	3/13	
ADO	AUXILIARY DEVICE ON	ESC	3/14	
ADF	AUXILIARY DEVICE OFF	ESC	3/15	
SCU	SCROLL UP	CSI	3/0	6/0
SCD	SCROLL DOWN	CSI	3/1	6/0
AIS	ACTIVATE IMPLICIT SCROLLING	CSI	3/2	6/0
DIS	DEACTIVATE IMPLICIT SCROLLING	CSI	3/3	6/0
EBU	EMPTY BUFFER	ESC	3/11	

3.7 Designation And Invocation in the 7-Bit Environment (Figure 2)

3.7.1 General

For the 7-bit environment the bases of the coding structure for the Videotex service are the CCITT recommendation V3 (ISO 646), and International Standards ISO 2022 (Rev 79) and ISO 6937.

3.7.2 Coding of Code Extension Control Functions

Abbreviation	Name and Coding	
SI	SHIFT IN	0/15
SO	SHIFT OUT	0/14
LS2	LOCKING SHIFT 2	ESC 6/14
LS3	LOCKING SHIFT 3	ESC 6/15
SS2	SINGLE SHIFT 2	1/9
SS3	SINGLE SHIFT 3	1/13

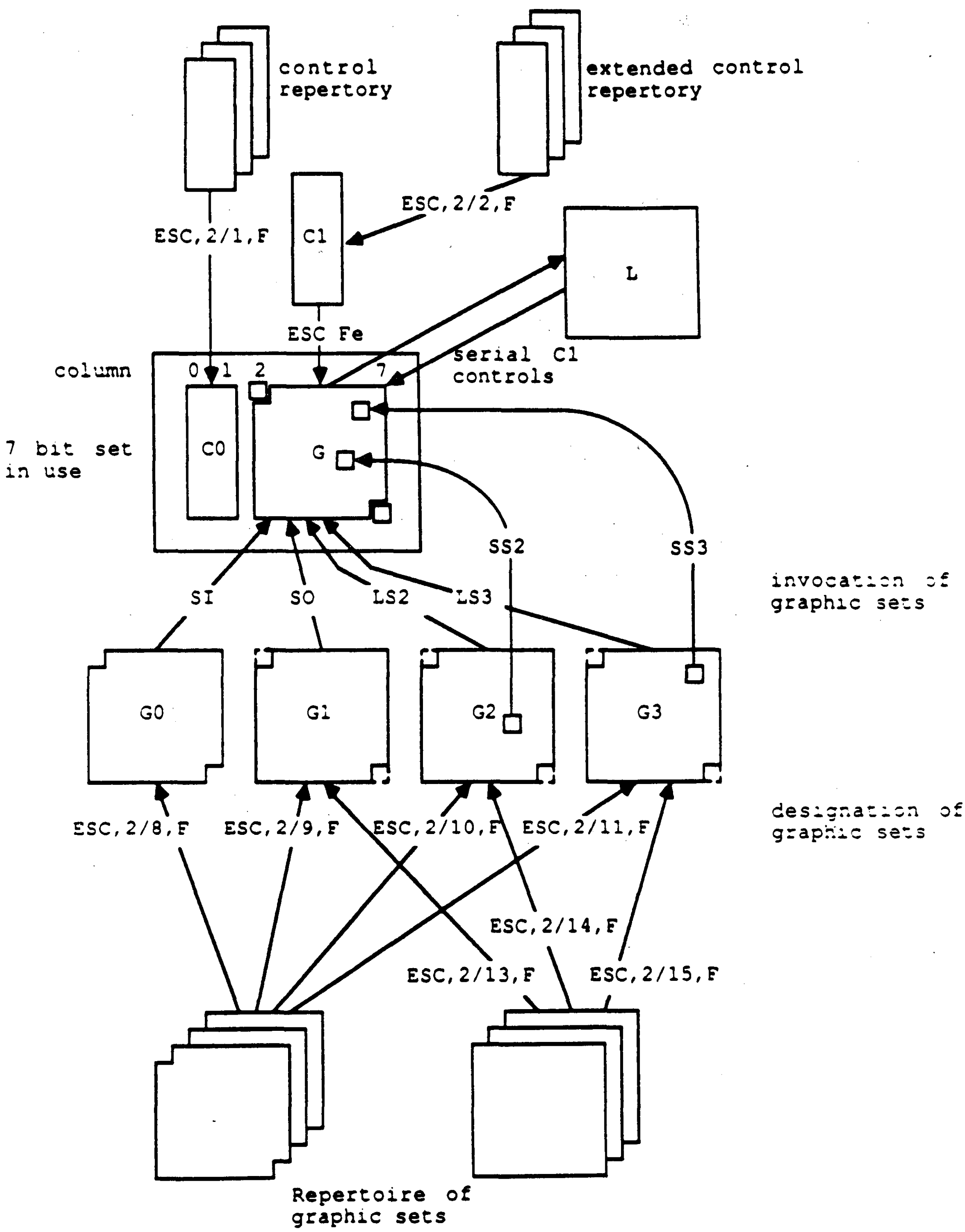


FIGURE 2 CODE EXTENSION IN A 7-BIT ENVIRONMENT

* See section 3.1.2.

(2604)

3.8 Designation and Invocation in the 8-Bit Environment (Figure 3)

3.8.1 General

The 8-bit code environment preserves the code extension structure of ISO 2022, ie the G0 set is invoked into the left-hand part (positions 2/1 to 7/14) and the G2 set into the right-hand part (positions 10/1 to 15/14) of the code table.

3.8.2 Coding of Code Extension Control Functions

Abbreviation	Name and Coding	
LS0	LOCKING SHIFT 0	0/15
LS1	LOCKING SHIFT 1	0/14
LS1R	LOCKING SHIFT 1 RIGHT	ESC 7/14
LS2	LOCKING SHIFT 2	ESC 6/14
LS2R	LOCKING SHIFT 2 RIGHT	ESC 7/13
LS3	LOCKING SHIFT 3	ESC 6/15
LS3R	LOCKING SHIFT 3 RIGHT	ESC 7/12
SS2	SINGLE SHIFT 2	1/9
SS3	SINGLE SHIFT 3	1/13

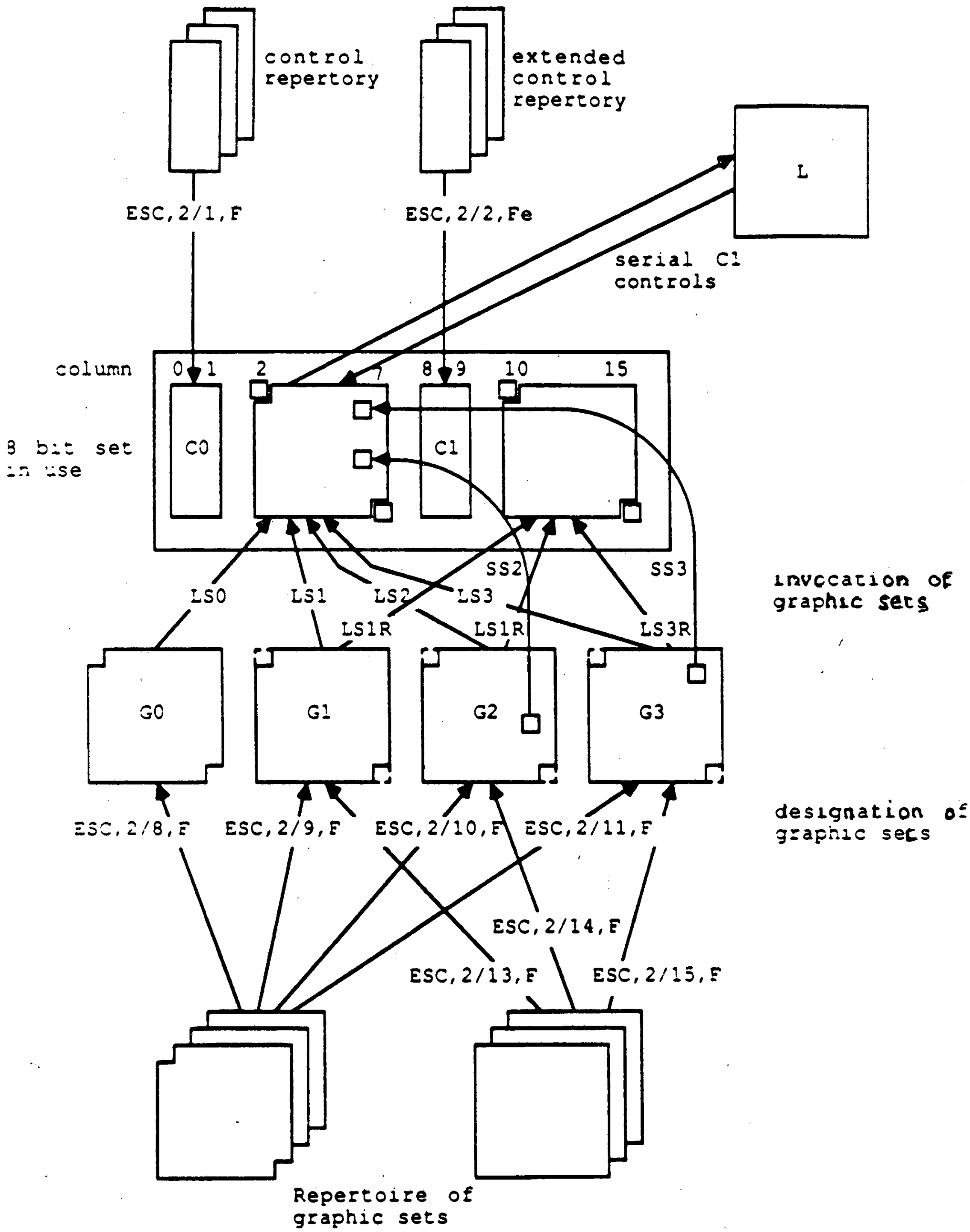


FIGURE 3 CODE EXTENSION IN AN 8-BIT ENVIRONMENT

* See section 3.1.3.

APPENDIX A

IDENTIFICATION SYSTEM

1. For the purpose of this Recommendation, a system has been developed that allows for the identification and description of each graphic character or control function. The system is shown in Table 13.

2. Each identifier consists of two letters and two digits.

3. The first letter indicates the alphabet, the language, etc.

4. The second letter indicates the letter of an alpabet or, in the case of a non-alphabetic graphic character or a control function, the group of characters or control functions.

5. The first digit indicates whether the letter in the second position is modified with a diacritical mark, the position of the diacritical mark, etc. It has no special meaning in the case of the first letter being a C, N or S.

6. The second digit indicates whether the letter is a capital or a small one (even or odd respectively). If the first letter is a C, N or S, this digit being even or odd has no significance.

7. The numbering is used in a consistent manner so that each diacritical mark is always given the same number.

8. The numbering principle is shown in Table 1.

TABLE 13 NUMBERING PRINCIPLE FOR ALPHABETIC CHARACTERS

Item	Small	Capital
No diacritical mark	01	02
Acute accent	11	12
Grave accent	13	14
Circumflex	15	16
Diaeresis or umlaut	17	18
Tilde	19	20
Caron	21	22
Breve	23	24
Double acute accent	25	26
Ring	27	28
Dot	29	30
Macron	31	32
Cedilla	41	42
Ogonek	43	44
Diphthong or ligature	51	52
Special form	61,63,etc	62,64,etc

```
L   A   0   1
|   |   |   |
|   |   |   |_ For alphabetic characters:
|   |   |   |     odd digit - small letter;
|   |   |   |     even digit - capital letter.
|   |   |   |
|   |   |   |_ If C, N or S in first position:
|   |   |         no special meaning.
|   |   |
|   |   |_ For alphabetic characters:
|   |   |     0 - letter without diacritical mark;
|   |   |     1, 2 or 3 - letter with diacritical mark above it;
|   |   |     4 - letter with diacritical mark below it;
|   |   |     5 - diphthong or ligature;
|   |   |     6 - special form.
|   |   |
|   |   |_ If C, N or S in first position:
|   |         no special meaning.
|   |
|   |_ For alphabetic characters:
|   |     A to Z - the respective letter of the Latin alphabet, or the
|   |              Latin equivalent in the case of a non-Latin letter.
|   |
|   |_ If C in first position:
|   |     E - code extension control function;
|   |     F - format effector;
|   |     P - presentation control function;
|   |     M - other control function.
|   |
|   |_ If N in first position:
|   |  D - decimal digit;
|   |  F - fraction;
|   |  S - subscript or superscript
|   |
|   |_ If S in first position:
|      A - arithmetic sign;
|      C - currency sign;
|      D - diacritical mark;
|      P - punctuation mark;
|      M - other symbol.
|
|_ L - Latin alphabetic character;
|_ C - control function;
|_ N - non-alphabetic graphic character;
|_ S - special graphic character.
```

FIGURE 4   IDENTIFICATION SYSTEM

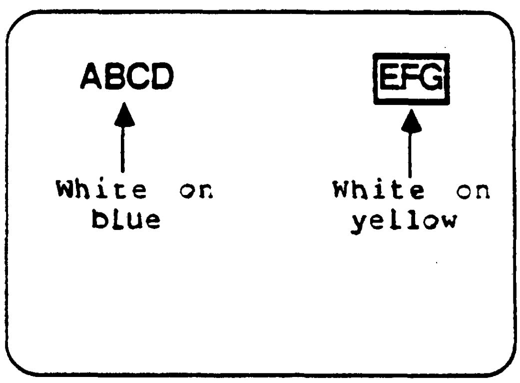
APPENDIX B

EXAMPLES OF TIME DEPENDENCY IN THE UNIFIED ALPHAMOSAIC MODEL

EXAMPLE 1

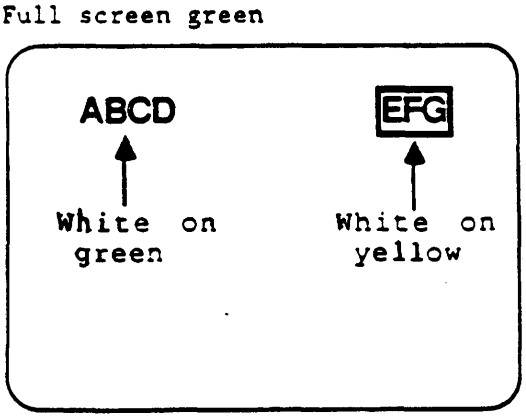
Codes:

Full screen blue



CS, full screen blue background, (transparent background), A, B, C, D, yellow background, E, F, G.

Full screen green



Full screen green background.

EXAMPLE 2

DISPLAY	SEQUENCE OF CODES CS
RED CHARACTERS · GREEN CHARACTERS A B C D E F G H I J K MARKERS	ESC,2/2,Fs ESC,4/1,APB A,B,C,D,E,F,G, ESC,4/2 H,I,J,K
RED · YELLOW · GREEN A B C L M N O P Q I J K	APR APF,APF,APF ESC,2/2,Fp ESC,4/3 L,M,N,O,P,Q
BLUE · YELLOW · GREEN A B C L M N O P Q I J K	APR ESC,2/2,Fs ESC,4/4
BLUE · YELLOW · GREEN A B R S T N O P Q I J K	APF, R,S,T
YELLOW · GREEN U V X Y Z N O P Q I J K	APR ESC,2/2,Fp U,V,X,Y,Z
RED U V X Y Z N O P Q I J K	Row attribute set to red foreground

February 1987 CEPT T/CD 6.1

CEPT

VIDEOTEX PRESENTATION LAYER DATA SYNTAX

GEOMETRIC DISPLAY

(T/CD 6.1 part 2)

Source: CEPT geometric experts

(2604)

TABLE of CONTENTS

8.2.1.1 OPEN WORKSTATION
8.2.1.2 CLOSE WORKSTATION
8.2.1.3 ACTIVATE WORKSTATION
8.2.1.4 DEACTIVATE WORKSTATION
8.2.1.5 CLEAR WORKSTATION
8.2.1.6 SET DEFAULTS
8.2.1.7 GDS ESCAPE1
8.2.2 Output workstation primitives
8.2.2.1 Output drawing primitives
8.2.2.1.1 POLYLINE
8.2.2.1.2 POLYMARKER
8.2.2.1.3 FILL AREA
8 2.2.1.4 TEXT
8.2.2.1.5 CELL ARRAY
8.2.2.1.6 GDP
8.2.2.2 Output primitives related to display element attributes
8.2.2.2.1 SET POLYLINE REPRESENTATION
8.2.2.2.2 SET POLYLINE INDEX
8.2.2.2.3 SET LINE TYPE
8.2.2.2.4 SET LINE WIDTH SCALE FACTOR
8.2.2.2.5 SET POLYLINE COLOUR INDEX
8.2.2.2.6 SET POLYMARKER REPRESENTATION
8.2.2.2.7 SET POLYMARKER INDEX
8.2.2.2.8 SET MARKER TYPE
8.2.2.2.9 SET MARKER SIZE SCALE FACTOR
8.2.2.2.10 SET POLYMARKER COLOUR INDEX
8.2.2.2.11 SET FILL AREA REPRESENTATION
8.2.2.2.12 SET FILL AREA INDEX
8.2.2.2.13 SET FILL AREA INTERIOR STYLE
8.2.2.2.14 SET FILL AREA COLOUR INDEX
8.2.2.2.15 SET FILL AREA STYLE INDEX
8.2.2.2.16 SET PATTERN REPRESENTATION
8.2.2.2.17 SET PATTERN REFERENCE POINT
8.2.2.2.18 SET PATTERN VECTORS
8.2.2.2.19 SET TEXT REPRESENTATION
8.2.2.2.20 SET TEXT INDEX
8.2.2.2.21 SET TEXT FONT AND PRECISION
8.2.2.2.22 SET CHARACTER EXPANSION FACTOR
8.2.2.2.23 SET CHARACTER SPACING
8.2.2.2.24 SET TEXT COLOUR INDEX
8.2.2.2.25 SET TEXT PATH
8.2.2.2.26 SET CHARACTER VECTORS
8.2.2.2.27 SET TEXT ALIGNMENT
8.2.2.2.28 SET COLOUR REPRESENTATION
8.2.2.2.29 SET ASPECT SOURCE FLAGS
8.2.2.3 Transformation primitives
8.2.2.3.1 SET WORKSTATION WINDOW
8.2.2.3.2 SET WORKSTATION VIEWPORT
8.2.2.4 Clipping primitives
8.2.2.4.1 SET CLIPPING RECTANGLE
8.2.2.5 Control primitives
8.2.2.5.1 UPDATE WORKSTATION
8.2.2.5.2 SET DEFERRAL STATE
8.2.2.5.3 EMERGENCY CLOSE
8.2.3 Segment related primitives
8.2.3.1 WDSS related primitives
8.2.3.1.1 CREATE SEGMENT

Figures

1. INTRODUCTION

ECMA-96 specifies a Graphics Data Syntax for a multiple workstation interface (GDS).

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 Rec. T.101 (Data syntax II).

This part (part 2 of T/CD 6.1, 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/CD 6.1.

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

A graphics application program, using GKS functions, communicates with the 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.

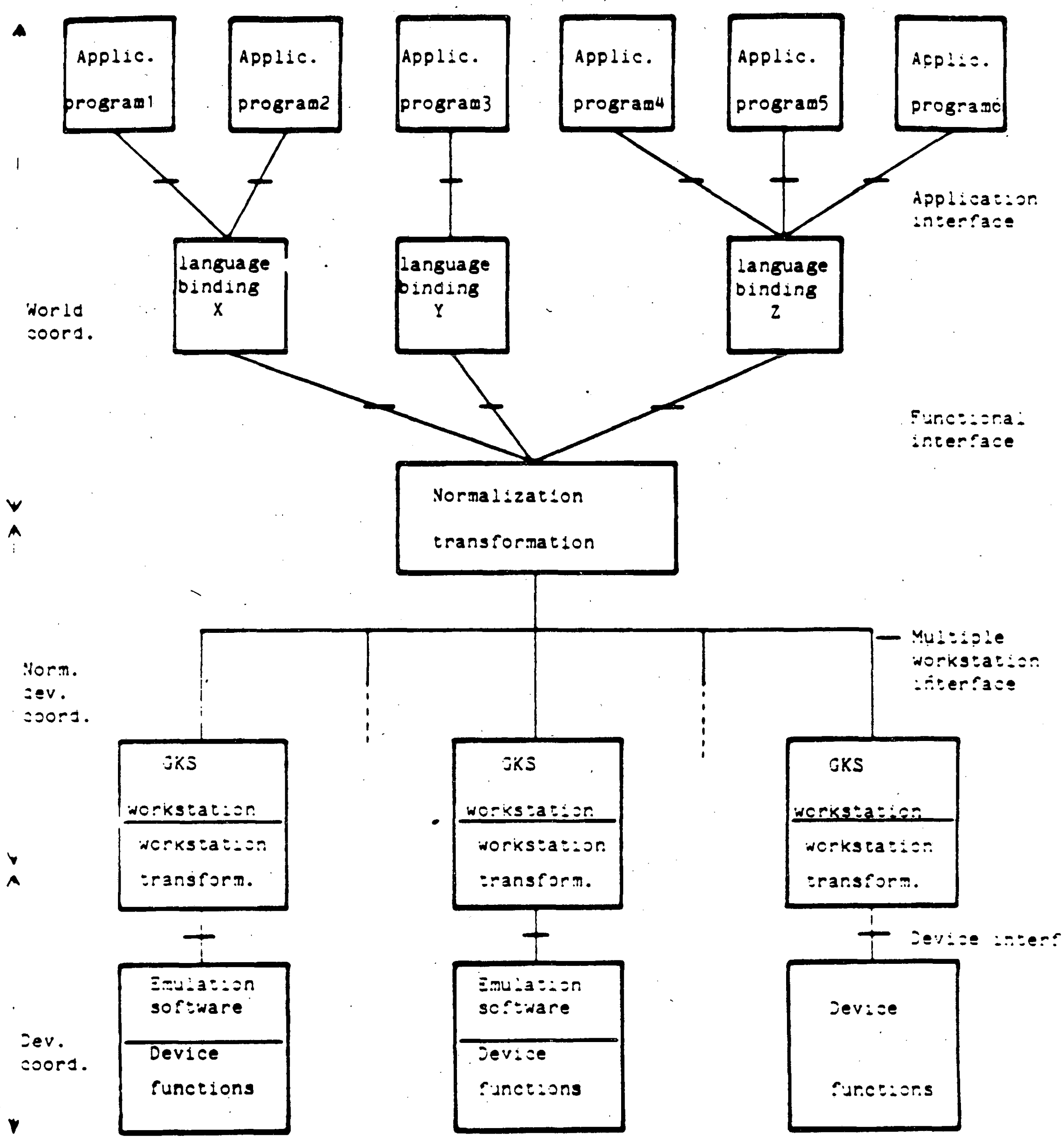


Figure 1 : Model describing the GKS environment and its interfaces.

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 - Rec. T.101 - International interworking for Videotex services.

- ECMA-96 - Syntax of graphical data for a multiple-workstation interface (GDS).

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, STROKE, VALUATOR, CHOICE, PICK and STRING.

LOCATOR DEVICE:
A logical input device providing a position in normalized device coordinates.

LOGICAL INPUT DEVICE:
A logical input device is an abstraction of one or more physical devices delivering a logical input value. Logical input devices can be of 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 ReGeneration
NDC	Normalized Device Coordinates
OUTIN	OUTput and INput workstation
OUTPUT	OUTPUT only workstation
WDSS	Workstation Dependent Segment Storage
WISS	Workstation Independent Segment Storage

5. GENERAL DESCRIPTION

This clause provides a description of all the concepts involved in 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 2.

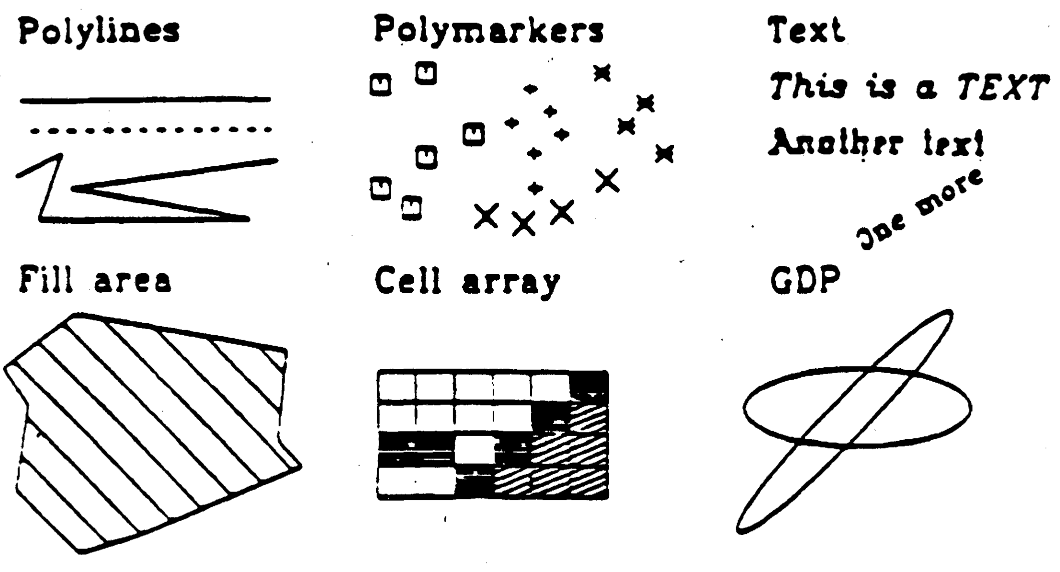


Figure 2 : Examples of display elements

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 ron-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 bundle tables with different values in a particular bundle for different workstations. For individual specification of aspects, there is a seperate 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 an 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.

The attributes which apply to each display element are:

a. POLYLINE
POLYLINE INDEX
LINE TYPE
LINE WIDTH SCALE FACTOR
POLYLINE COLOUR INDEX
LINE TYPE ASF
LINE WIDTH 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:

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

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

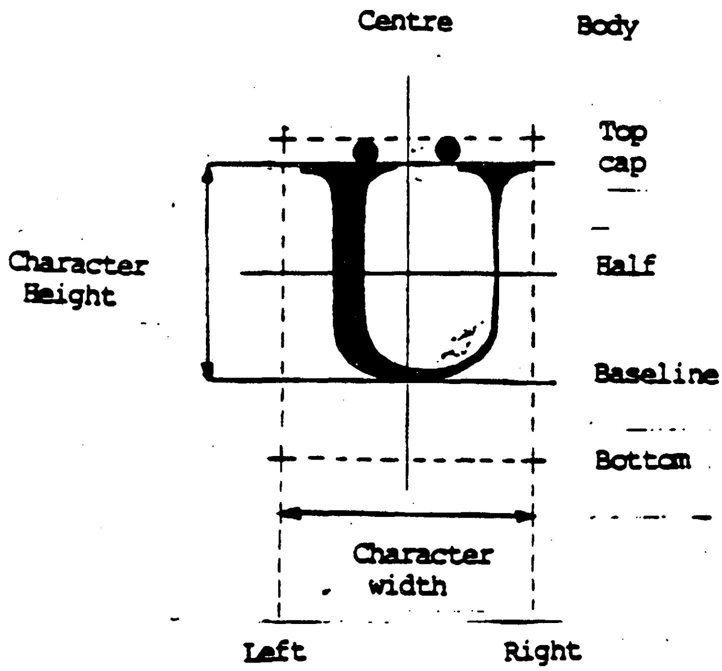


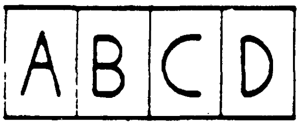
Figure 3 : Font description coordinate system

For monospaced fonts the character bodies of all characters have the same size. For proportionally spaced fonts, the width of the bodies may differ from character to character. The character body edges must be parallel to the character vectors. The font baseline and the capline must be parallel to the width vector and within the vertical extent of the body. The centre line is parallel to the height vector and bisects the body.

The height of a character in the font coordinate system is given by the height from the font base line 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.

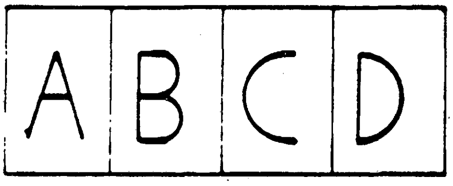
In general, the top limits of the bodies for a font will be identical with or very close to the typographical capline or ascender line and the bottom limit to the descender line. However, these and other details are purely for the use of the font designer. The intention is only that characters placed 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 4 to 8 are only inserted for clarification purposes. They show the effects of the different attributes on the display of the text "ABCD".



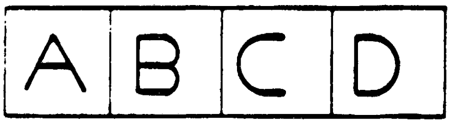
CHARACTER VECTORS = (0.0, 0.025) and (0.025, 0.0)

CHARACTER EXPANSION FACTOR = 1.0



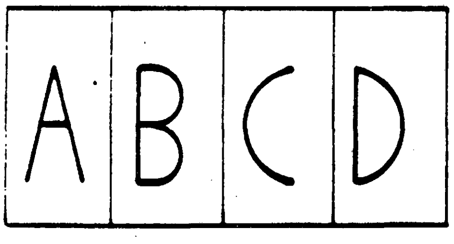
CHARACTER VECTORS = (0.0, 0.0375) and (0.0375, 0.0)

CHARACTER EXPANSION FACTOR = 1.0



CHARACTER VECTORS = (0.0, 0.025) and (0.025, 0.0)

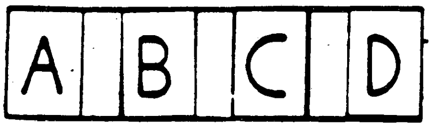
CHARACTER EXPANSION FACTOR = 1.5



CHARACTER VECTORS = (0.0, 0.050) and (0.050, 0.0)

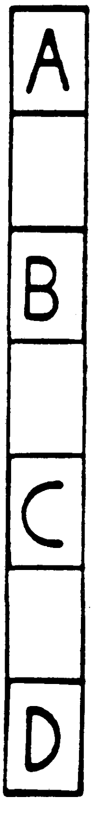
CHARACTER EXPANSION FACTOR = 0.75

Figure 4 : CHARACTER VECTORS and CHARACTER EXPANSION FACTOR



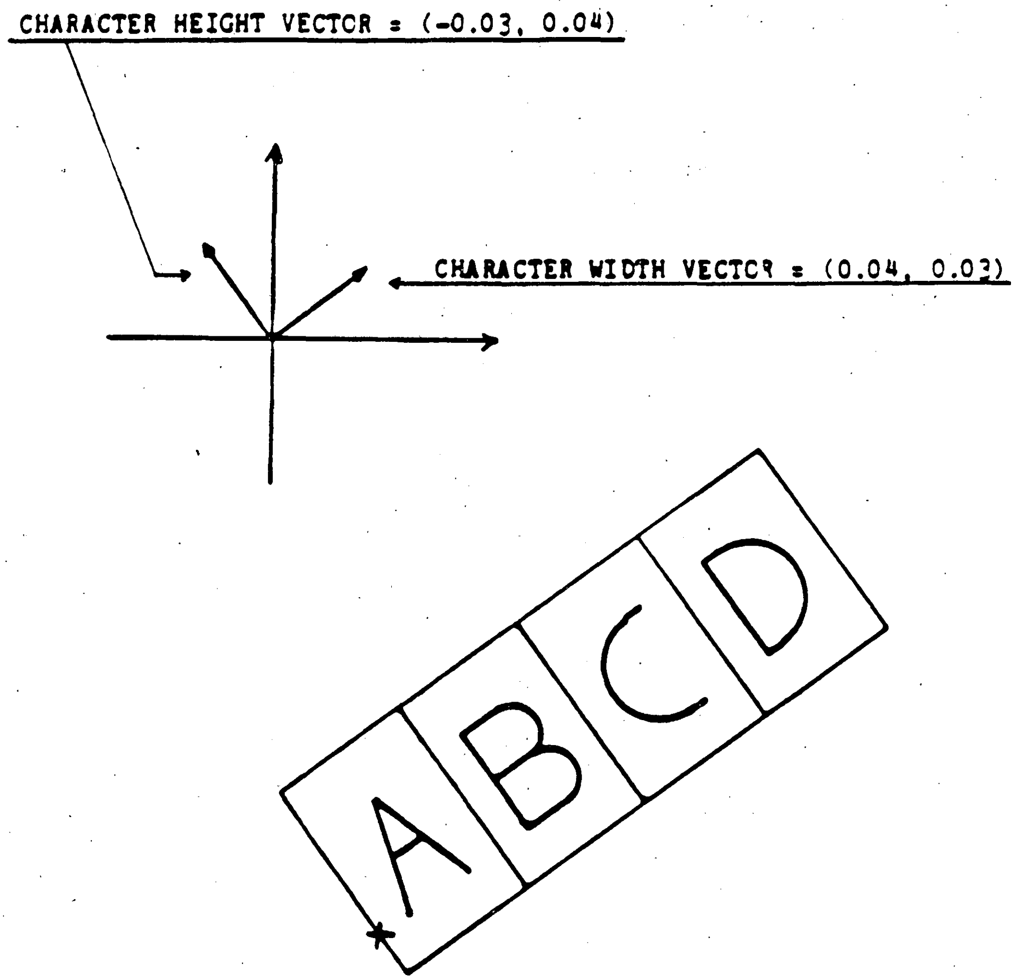
CHARACTER VECTORS = (0.0, 0.025) and (0.025, 0.0)
CHARACTER SPACING = 0.67
TEXT PATH = right

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



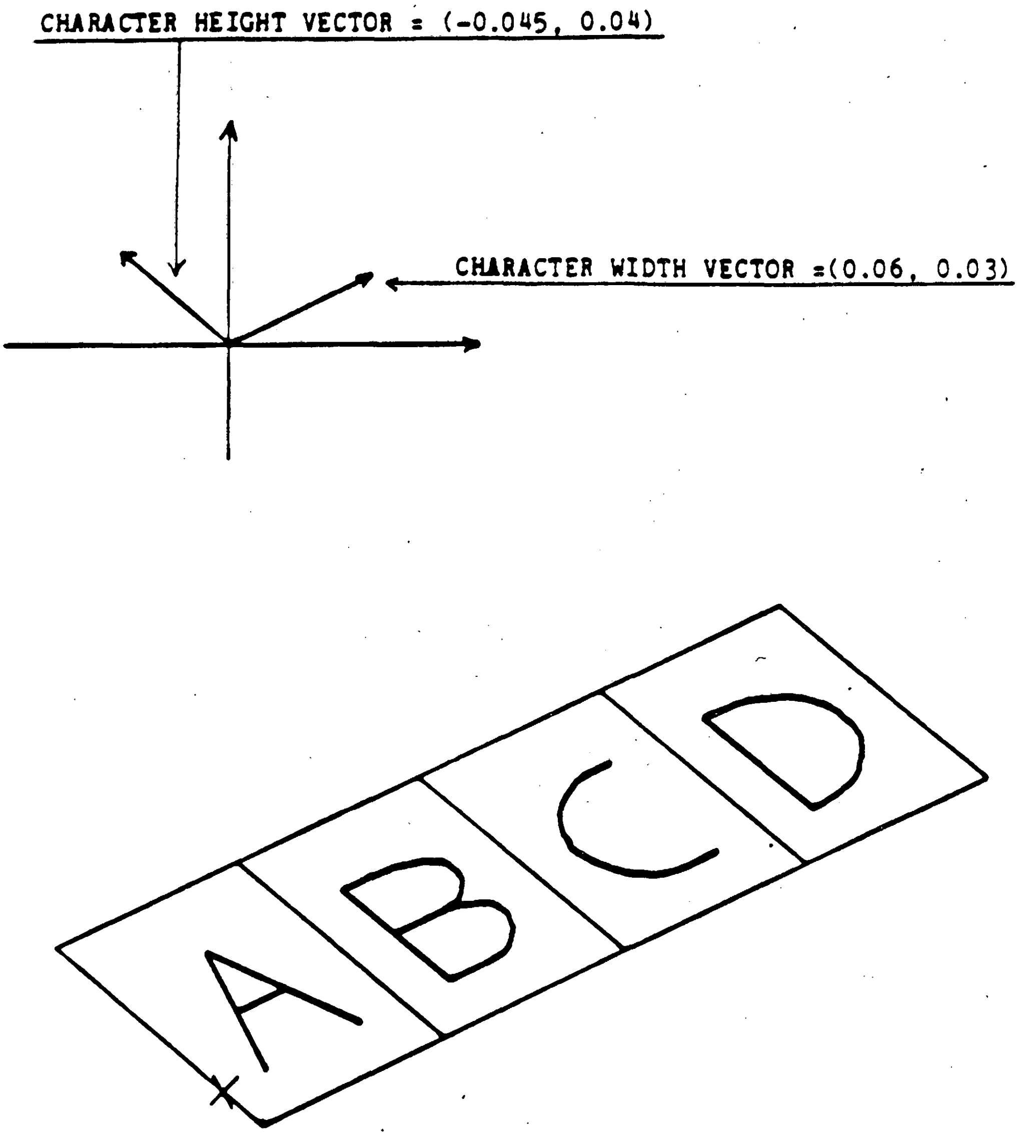
CHARACTER VECTORS = (0.0, 0.025) and (0.025, 0.0)
CHARACTER SPACING = 2.0
TEXT PATH = down

Figure 5 : CHARACTER SPACING and TEXT PATH



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

Figure 6 : CHARACTER VECTORS



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

Figure 7 : CHARACTER VECTORS after anisotropic transformation

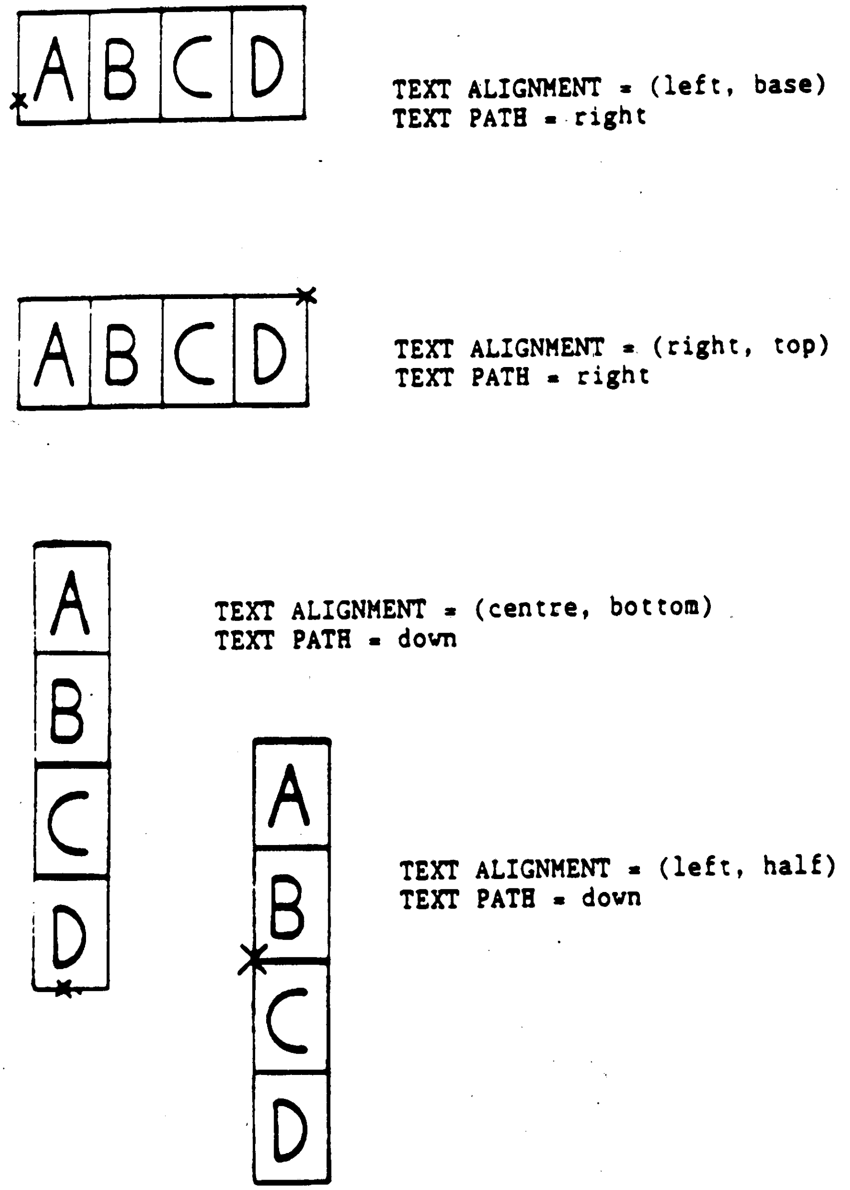


Figure 8 : TEXT ALIGNMENT and TEXT PATH

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 1 may be used for invocation purposes. 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 1

Permitted invocation sequences in a 7-bit environment

In an 8-bit environment the shift functions contained in Table 2 may be used for invocation purposes, see Table 2. The coded representation of these shift functions is defined in part 1 of this document.

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

Table 2

Permitted invocation sequences in an 8-bit environment

Characters in the text string not from the invoked G-set or not from table 1, in a 7-bit environment, will be ignored. In an 8-bit environment characters not from the invoked G-sets or not from table 2 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
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 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] x [0.0, 1.0] to be displayed somewhere on the workstation display surface. The workstation transformation is a uniform mapping from NDC onto DC and thus performs translation and equal scaling with a positive scale factor for the two axes.

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

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 10):

- 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 12.

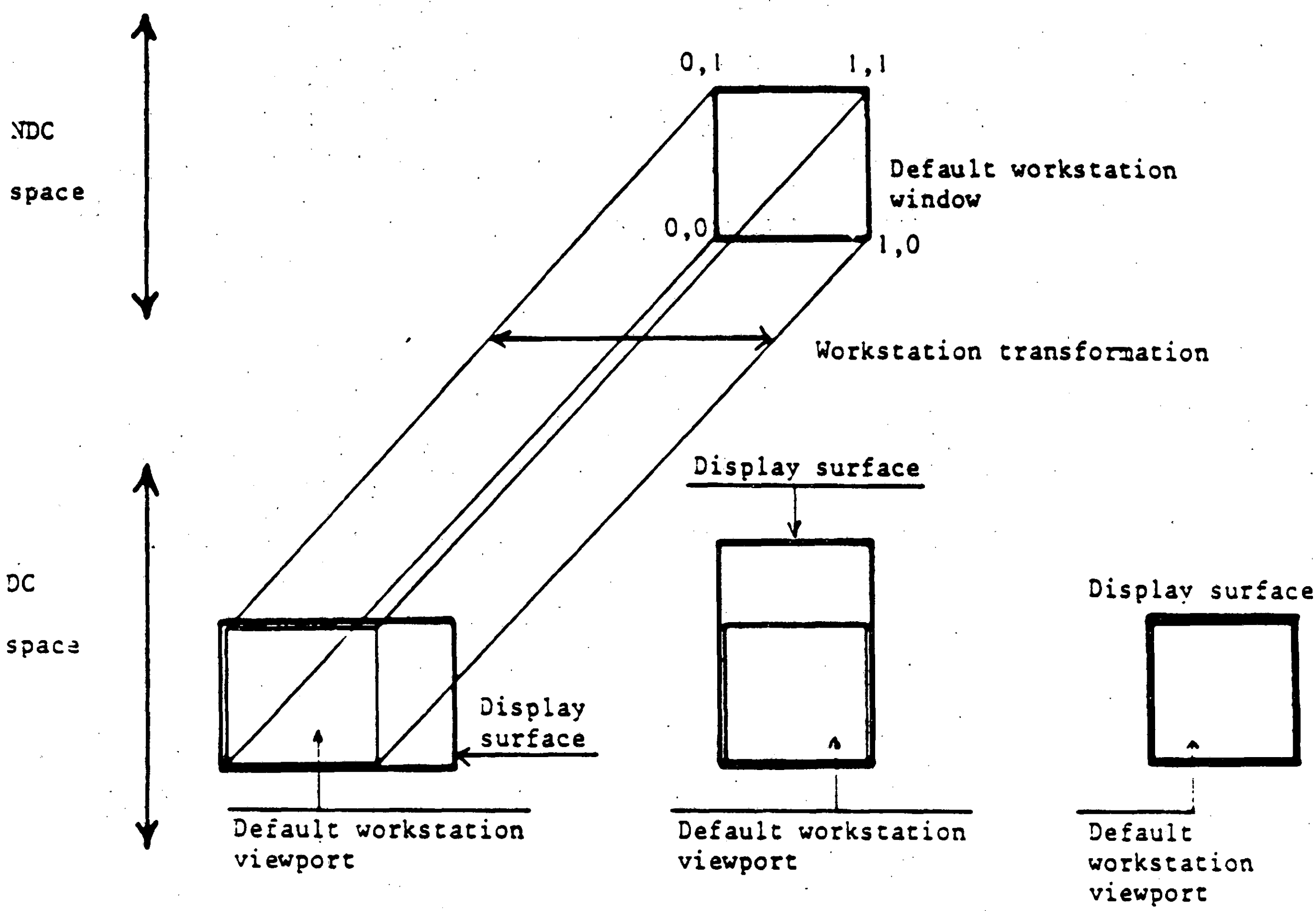


Figure 9 : Default workstation transformation

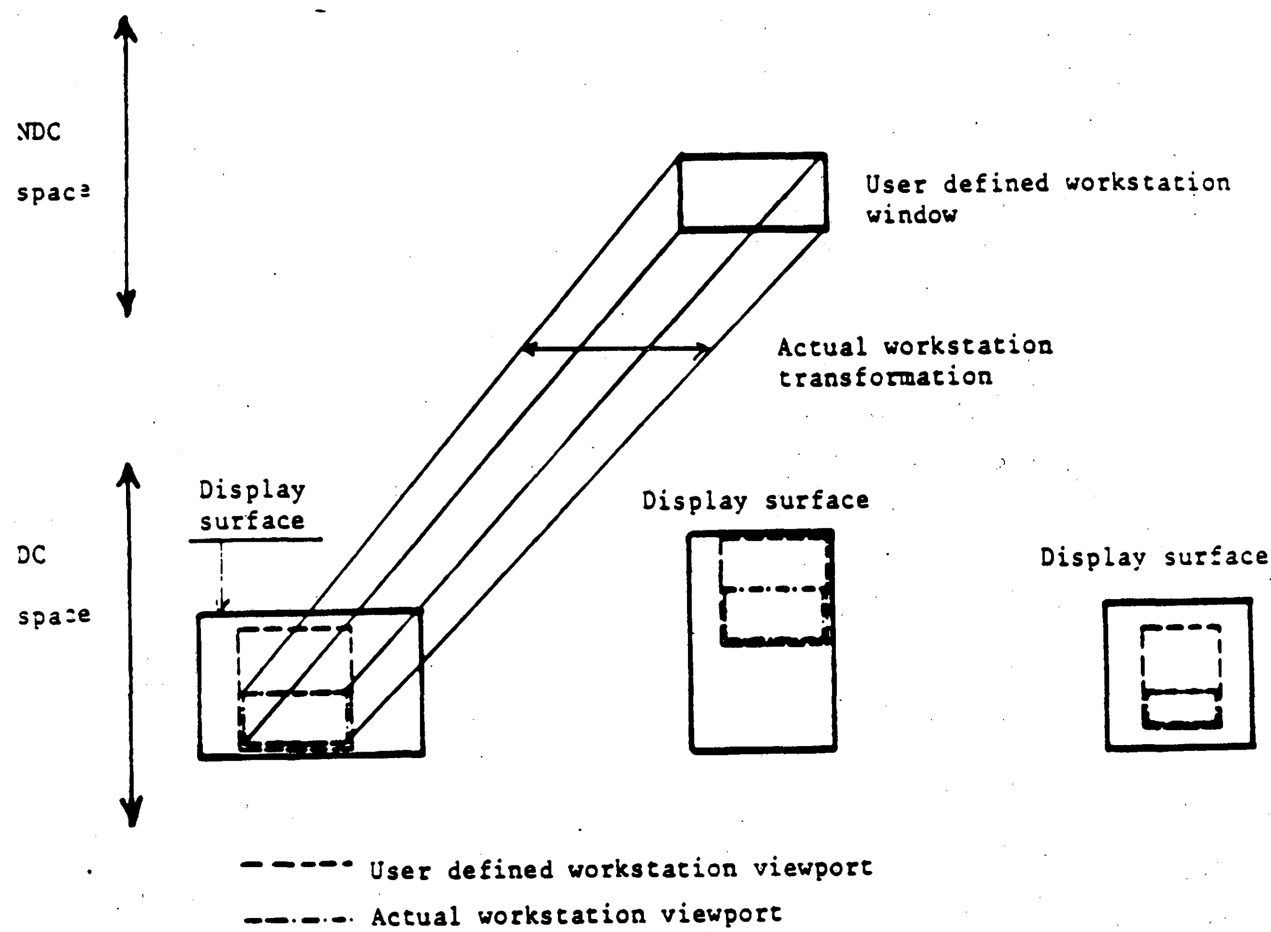


Figure 10 : Workstation transformation with anisotropic workstation window and workstation viewport

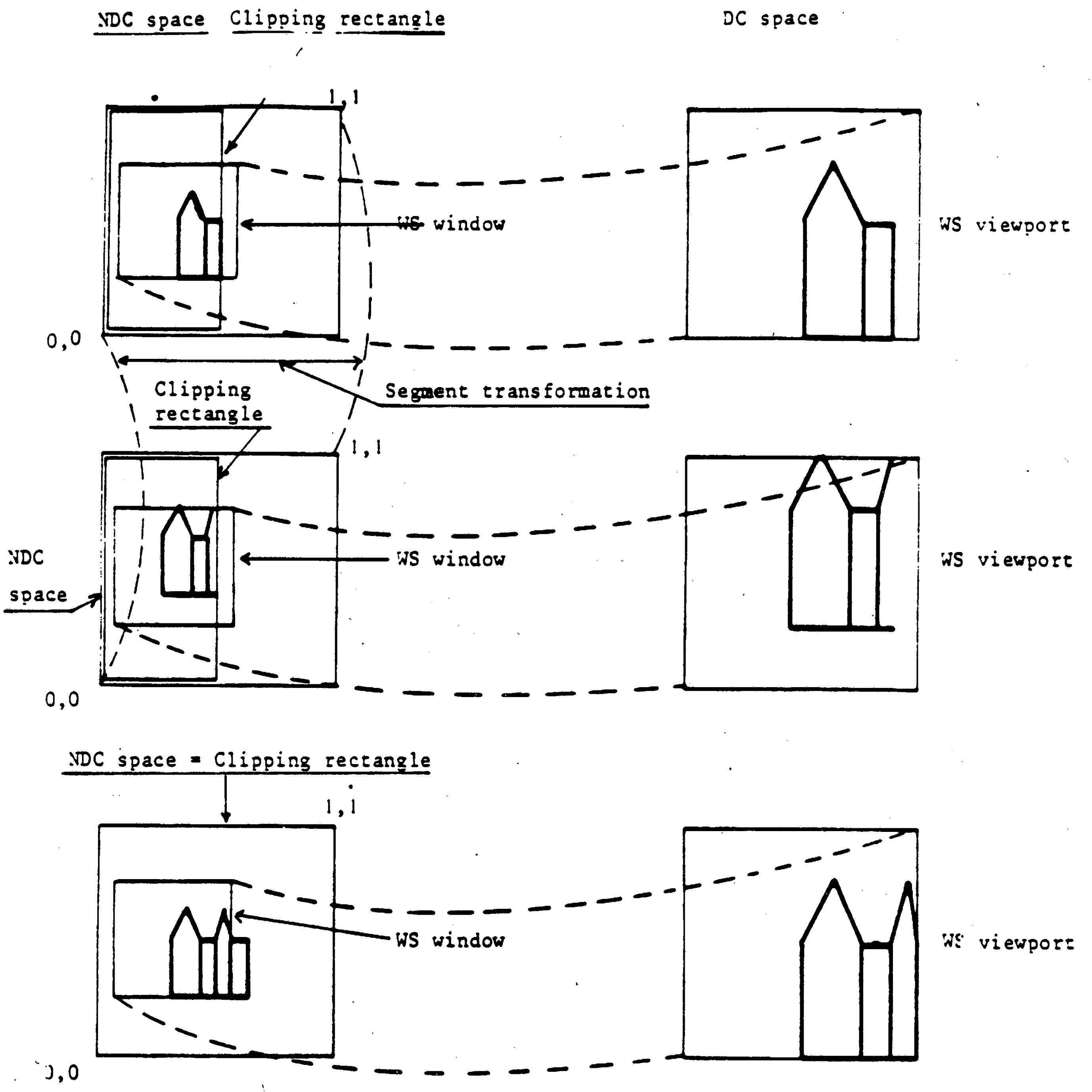


Figure 11 : Clipping and workstation transformation

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.

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 segment or into the stream of primitives outside segments.

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

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

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

5.4.2 Segment attributes

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

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

- VISIBILITY
  A segment is either displayed or not.

- HIGHLIGHTING
  A visible segment is either highlighted or not.

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

- DETECTABILITY
  A segment can either be selected by a pick input device or it cannot.

The segment attributes are unique for each segment and do not vary on different workstations. The default segment attributes (identity transformation, visible, not highlighted, priority 0.0, undetectable) are assigned to a segment when it is created. The segment attributes of any segment in existence, including the open segment, may be changed.

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

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

5.4.3 Segment transformations

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

Segment transformations are characterized by:

- segment name;

- transformation matrix.

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

The segment transformation takes place before any clipping.

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

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

5.4.4 Clipping and WDSS

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

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

5.4.5 Workstation Independent Segment Storage

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

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

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

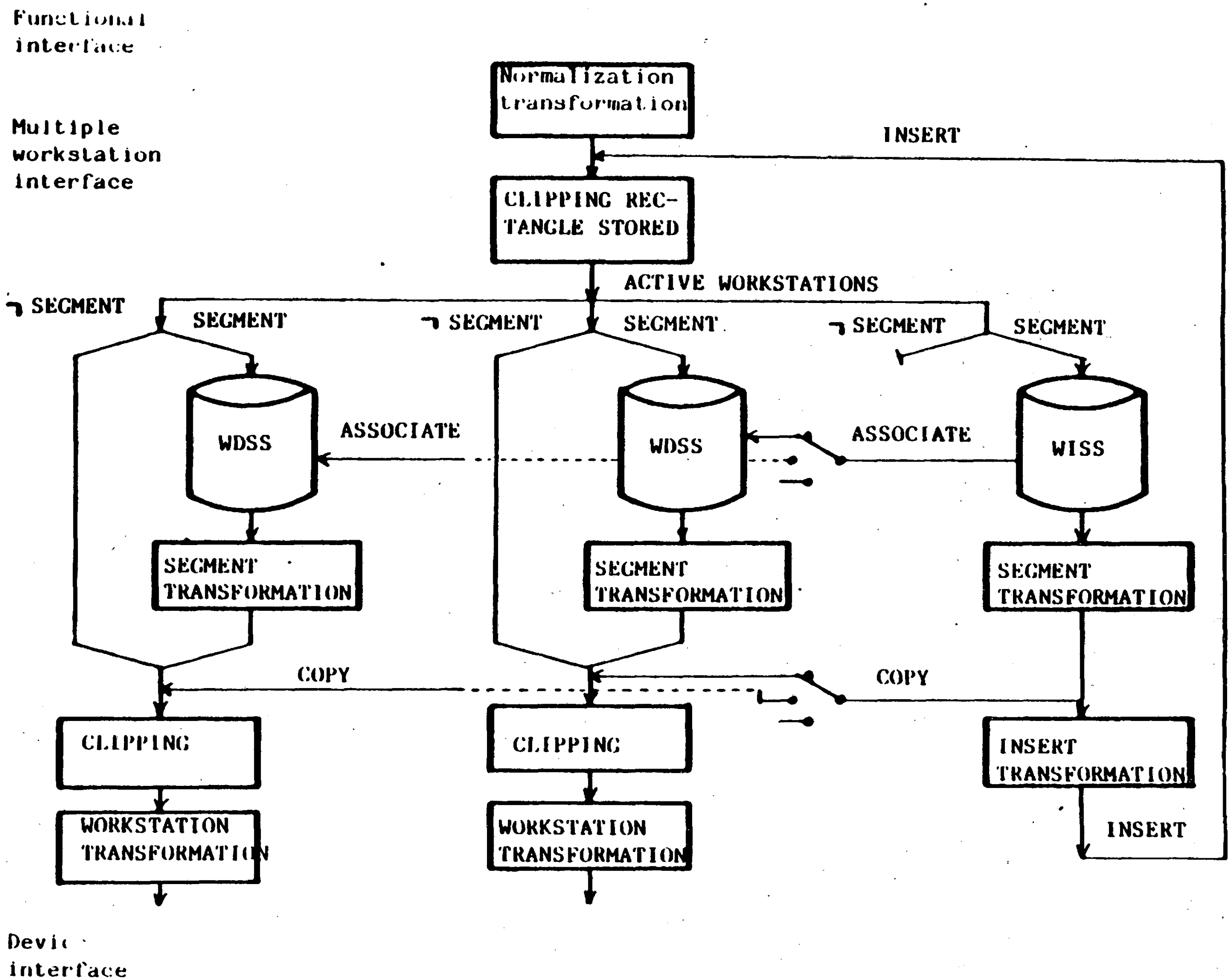


Figure 12 : Data flow chart for graphical output

5.4.6 VISS functions and clipping

Just as in other workstations, a segment is stored in VISS if VISS 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 VISS to be output on the specified workstation. The primitive takes a copy of each primitive and its associated clipping rectangle from a segment in VISS, 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 VISS) to be transformed and again placed into the stream of output primitives. INSERT SEGMENT reads the primitives from a segment in the VISS, 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 VISS, if the VISS 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.

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 control 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 level 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 include PICK if and only if combined with output level 1 capabilities.

Input level c : full input.

a. all input level b capabilities;

b. SAMPLE and EVENT mode input.

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

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

Table 3

Level concept

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

CAPABILITY	LEVEL 0a	0b	0c	1a	1b	1c	2a	2b	2c
Colours (Intensity)	1	1	1	1	1	1	1	1	1
Line types	4	4	4	4	4	4	4	4	4
Line widths	1	1	1	1	1	1	1	1	1
Predefined polyline bundles	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 note 2 and 5)	-	-	-	10	10	10	10	10	10
Hatch styles (see note 3)	3	3	3	3	3	3	3	3	3
Predefined fill area bundles	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 4

Minimal support required at each level.

Note 1

Relevant only for character and string precision text.

Note 2

Relevant only for workstation supporting pattern interior style.

Note 3

Relevant only for workstation supporting hatch interior style.

Note 4

Relevant only for workstation supporting segment priorities.

Note 5

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

Parameters: None.

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

Related primitives: None.

Discussion: None.

6.2.1.7 GDS ESCAPE1

Level: 0a

Parameters:
- GDS escape identifier (I)
- GDS data record. (REC)

Description: The GDS ESCAPE1 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

Parameters: - 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

Parameters: - 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

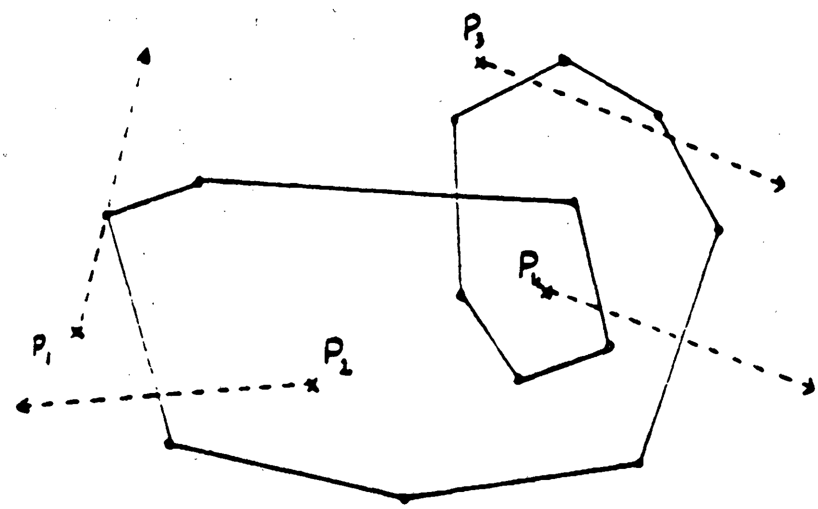
Discussion: None.

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-R direction m (I)
- Number of cells in Q-R direction n (I)
- Cell colour index list (CL)

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

This parallelogram is subdivided in m * n contiguous parallelograms in the following way. The side PR 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 13. 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.

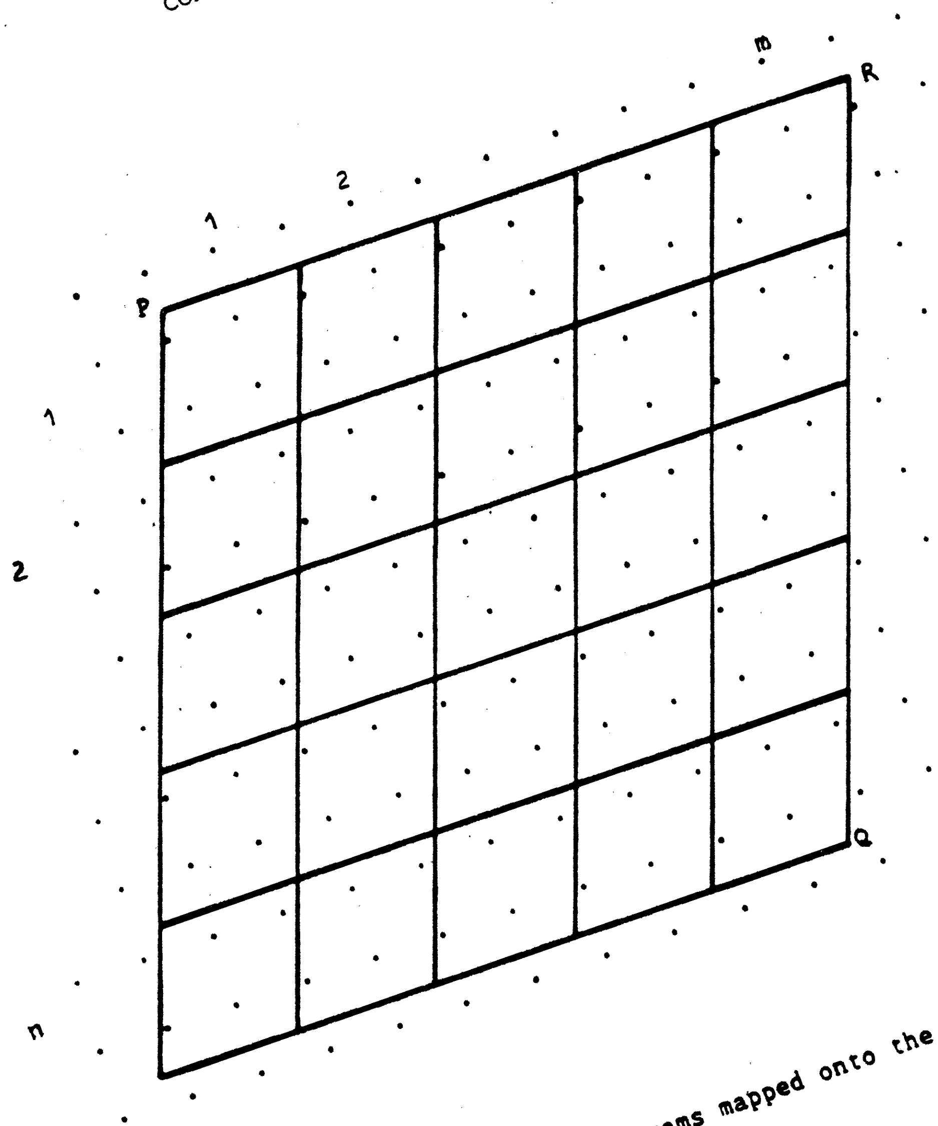


Figure 13 : M by n parallelograms mapped onto the display surface.

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 (= rectangle) (I)
- Two points (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 (= circle) (I)
- Centre (P)
- Radius (V)

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

Related primitives: Same as for FILL AREA.

Discussion: If the radius parameter has the value 0 a dot is displayed at the centre position.

If the radius parameter is negative, the primitive is ignored.

6.2.2.1.6.3 GDP (circular arc 3 point)

Level: 0a

Parameters:
- GDP identifier (= circular arc 3 point) (I)
- Starting point, intermediate point, ending point (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.

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

6.2.2.1.6.4 GDP (circular arc 3 point chord)

Level: 0a

Parameters:
- GDP identifier (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.

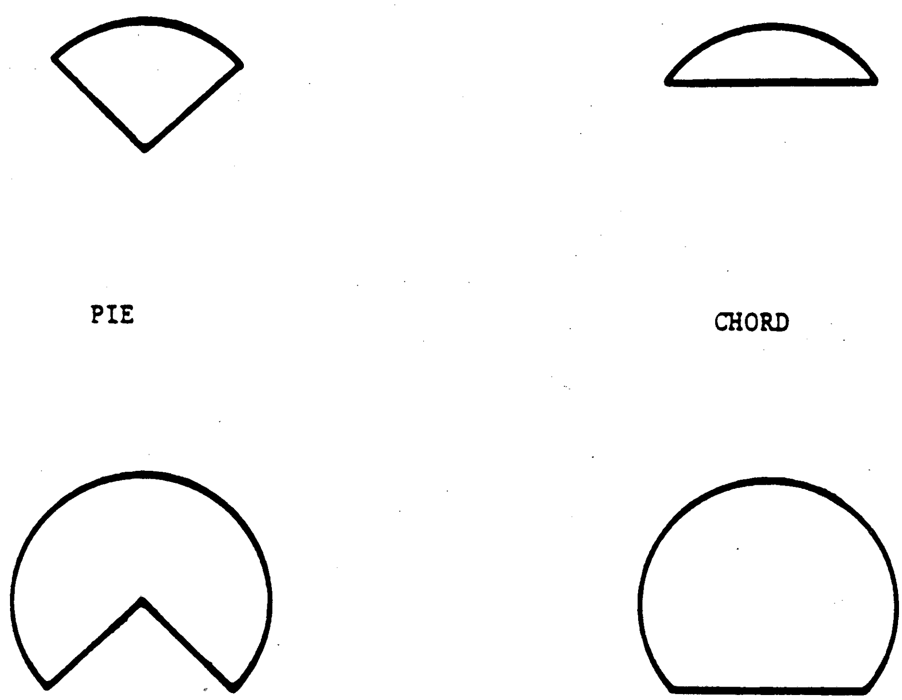


Figure 14 : GDP specifications for "arc pie" and " arc chord".

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 fifth 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 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 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 point (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:

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

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

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

Related primitives: Same as for FILL AREA.

Discussion:

If the three points coincide a dot is displayed.

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

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

6.2.2.1.6.10 GDP (elliptic arc)

Level: 0a

Parameters:

- GDP identifier (= elliptic arc) (I)
- Centre point (P)
- Endpoints (X1, Y1) and (X2, Y2) (2P)
- T_start, T_end (2R)

Description:

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

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

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

Related primitives: Same as for POLYLINE.

Discussion:

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

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

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

If T_start equals T_end the full ellipse is displayed.

6.2.2.1.6.11 GDP (elliptic arc chord)

Level: 0a

Parameters:

- GDP identifier (= elliptic arc chord) (I)
- Centre point (P)
- Endpoints (X1, Y1) and (X2, Y2) (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 colineair a straight line connecting the two endpoints is displayed.

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

If T_start equals T_end the full ellipse is displayed.

6.2.2.1.6.12 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 colineair a straight line connecting the two endpoints is displayed.

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

If T_start equals T_end the full ellipse is displayed.

6.2.2.1.6.13 GDP (spline)

Level: 0a

Parameters:
- GDP identifier (= spline) (I)
- 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.

6.2.2.2.4 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
o CIRCLE
x 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 the specified workstation the given fill area index is associated with the specified parameters:

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

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

Which of the aspects in the entry are used depends upon the setting of the corresponding 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 if {interior style = HATCH} (I)
  or
  Fill area style index if {interior style = PATTERN} (IX)

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 diplay 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(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 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 a value less than or equal to 0, the default character expansion factor will be set.

6.2.2.2.23 SET CHARACTER SPACING

Level: 0a

Parameters: - Character spacing (R)

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

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

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

The parameter represents the desired space to be added between characters of a text string. It is specified as a fraction of the current character height vector attribute. The space is added along the 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 lenght 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 are dependent on the text path at the time of the elaboration of the TEXT display element.

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

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 (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 <u>SET ASPECT SOURCE FLAGS</u>

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

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: XMIN < XMAX, YMIN < 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 <u>CLOSE SEGMENT</u>

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 <u>RENAME SEGMENT</u>

Level: 1a

Parameters:
- Old segment name (ID)
- New segment name (ID)

Description: Each occurence 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 <u>DELETE SEGMENT FROM WORKSTATION</u>

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:

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

The original coordinates are (x, y), the transformed coordinates are (x', y'), both in NDC.

The values 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 VISS 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 VISS 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 VISS or if the specified workstation is the VISS, 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:

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

The original coordinates are (x, y), the transformed coordinates are (x', y'), both in NDC.

The values 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 value as a binary fraction plus one bit for the sign bit).

The granularity code parameter specifies the smallest nonzero value that can be expressed with this precision. This value is called the Basic Grid Unit (BGU). It does this by specifying the smallest exponent that is permitted in the coded representation of a coordinate or a size value.

The default exponent parameter specifies the exponent value, which is used when encoding a point, in case:

- the exponent is omitted in the encoding and

- the encoded point is a single point or the first point of a point list.

It also specifies the exponent value, which is used when encoding a size value, in case the exponent is omitted in the encoding.

The explicit exponent allowed parameter specifies whether or not an exponent part is allowed in the coded representation of a coordinate or a size value.

For example, if the magnitude code = +11 and the granularity code = -16 the coordinates in the range from -1111111111.1111111111111111 (binary) to +1111111111.1111111111111111 (binary) can be encoded, with each encoded coordinate being a multiple of 2**-16 (binary 0.0000000000000001), the BGU.

Related primitives: SET DOMAIN RING

Discussion: The magnitude code must be greater than the granularity code. The magnitude 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 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 D.

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 opcodes    operands
             control
          functions <---------------------->
                          primitives
```

7-bit Code Table as used for GDS coding.

Table 5

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 15.

```
 b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| - | 0 | 1 |   |   |   |   |   |  1st byte
+---+---+---+---+---+---+---+---+
    |___|   |   |___________|
      T     |         T________  Opcode
 Opcode flag|____________________ Opcode length indicator

+---+---+---+---+---+---+---+---+
| - | 0 | 1 |   |   |   |   |   |  2nd byte (used in double-byte
+---+---+---+---+---+---+---+---+            opcodes only)
    |___|   |_______________|
      T             T__________  Opcode
 Opcode flag
```

Figure 15 : Opcode encoding structure.

For single-byte opcodes the opcode length indicator, bit b5, is ZERO (opcodes of column 2). Bits b4 to b1 are used to encode the opcode.

For double-byte opcodes the opcode length indicator, bit b5 of the first byte, is ONE. Bits b4 to b1 of the first byte and bits b5 to b1 of the second byte are used to encode the opcode (first byte of the opcode is from column 3, the second byte being from column 2 or 3).

The code EXTEND OPCODE SPACE (EOS, 3/15) is used in a different sense (see section 7.1.2).

The basic opcode set, supplied by this encoding technique consists of 496 opcodes, being:

- 16 single-byte opcodes (from column 2);

- 15 x 32 = 480 double-byte opcodes (first byte from column 3 except code 3/15, second byte from column 2 or column 3).

7.1.2 Extension mechanism

The basic opcode set can be extended with an unlimited number of extension opcode sets by means of the EXTEND OPCODE SPACE code (EOS, 3/15).

The N-th extension opcode set consists of opcodes of the basic opcode set, prefixed with n times the code EOS. The three possible formats of an opcode from the N-th extension opcode set are:

```
Opcode format   Extension codes   Basic opcode set codes

      1          <EOS>...<EOS>            <2/x>
                   |_______|
                       |
                    n times

      2          <EOS>...<EOS>            <3/y> <2/z>
                   |_______|
                       |
                    n times

      3          <EOS>...<EOS>            <3/y> <3/z>
                   |_______|
                       |
                    n times
```

<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 16.

```
 b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| - | 1 |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+
      |   |_____________________|
      |          operand
 operand flag
```

Figure 16 : Operand encoding structure.

The encoding of the operands may make use of the following DATATYPES:

a.	Identifier	(ID)
b.	Enumerated type	(E)
c.	String	(S)
d.	Record	(REC)
e.	Point	(P)
f.	Colour index	(CI)
g.	Index	(IX)
h.	Size value	(V)
i.	Integer number	(I)
j.	Real number	(R)
k.	Matrix	(M)
l.	Colour direct	(CD)
m.	Colour list	(CL)

To encode the 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 17:

```
 b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| - | 1 |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+
        |   |   |_______________|
        |   |           |
        |   |       databits
        | extension
        |   flag
        |
 operand flag
```

Figure 17 : Basic format Structure.

Bit b6 of each byte is the extension flag. For single byte operands, the extension flag is ZERO. In multiple-byte operands, the extension flag is ONE in all bytes except the last byte, where it is ZERO.

The Basic format is used to encode:

- Identifiers (ID)
- Enumerated types (E)
- Colour indices (CI)
- Indices other than colour indices (IX)
- Integer numbers (I)

The most significant part of the operand is coded in the first byte. The least significant part of the operand is coded in the last byte.

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 18 : 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 19 : 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 20 : 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 21 : 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 22 : 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 23 : 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 24 : 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 25 : Integer encoding for private use (more than one byte).

7.2.2 Real format

Each Real format consists of a mantissa part and an optional exponent part, both coded in Basic format. The exponent is the power of two by which the mantissa is to be multiplied. The exponent may be implicitly defined as a default exponent, which is then omitted in the Real format 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 26 and 27 show the general format of a mantissa. Figure 28 shows an example of a multiple-byte mantissa.

```
 b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| - | 1 | 1 |   |   |   |   |   |   1st byte
+---+---+---+---+---+---+---+---+
      |   |   |   |   |_______|
      |   |   |   |       |_ data bits
      |   |   |   |_ exponent follows bit
      |   |   |_ sign bit
      |   |_ extension flag
      |_ operand flag

+---+---+---+---+---+---+---+---+
| - | 1 | 1 |   |   |   |   |   |  2nd byte
+---+---+---+---+---+---+---+---+
      .                   .
      .                   .
+---+---+---+---+---+---+---+---+
| - | 1 | 0 |   |   |   |   |   |  n-th byte
+---+---+---+---+---+---+---+---+
      |   |   |_______________|
      |   |           |_ databits
      |   |_ extension flag
      |_ operand flag
```

Figure 26 : Encoding of a mantissa using the Basic format

Explicit exponent allowed = ALLOWED

```
  b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| - | 1 | 1 |   |   |   |   |   |   1st byte
+---+---+---+---+---+---+---+---+
      |   |   |   |_________|
      |   |   |        |_ data bits
      |   |   |_ sign bit
      |   |_ extension flag
      |_ operand flag

+---+---+---+---+---+---+---+---+
| - | 1 | 1 |   |   |   |   |   |  2nd byte
+---+---+---+---+---+---+---+---+
      .                   .
      .                   .
+---+---+---+---+---+---+---+---+
| - | 1 | 0 |   |   |   |   |   |  n-th byte
+---+---+---+---+---+---+---+---+
      |   |   |_______________|
      |   |           |_ databits
      |   |_ extension flag
      |_ operand flag
```

Figure 27 : Encoding of a mantissa using the Basic format.

Explicit exponent allowed = FORBIDDEN

```
      b8  b7  b6  b5  b4  b3  b2  b1
    +---+---+---+---+---+---+---+---+
    | - | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
    +---+---+---+---+---+---+---+---+
sign (negative)_|   |   |_______|
  no exponent ______|       |_______
                                    |
    +---+---+---+---+---+---+---+---+   |
    | - | 1 | 1 | 0 | 0 | 0 | 0 | 0 |   |
    +---+---+---+---+---+---+---+---+   |
                |_______________|       |
                        |_______________|__ magnitude
                                        |   value: 1024
    +---+---+---+---+---+---+---+---+   |
    | - | 1 | 0 | 0 | 0 | 0 | 0 | 0 |   |
    +---+---+---+---+---+---+---+---+   |
                |_______________|       |
                        |_______________|
```

Figure 28 : 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 data type being encoded.

If an exponent is omitted in a real number, a default value determined by SET REAL PRECISION is assumed.

If an exponent is omitted in a size value, a default value determined by SET COORDINATE PRECISION is assumed.

If an exponent is omitted in the x-component of a point, the exponent used for the x-component of the preceding point is assumed. Likewise, the exponent in a y-coordinate defaults to the value of the exponent in the preceding y-coordinate. For 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 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:

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

N must be even. If N is odd, the encoded operand (the point list) must be discarded. If N is even for the first part of the operand and N is odd for the remaining part, the remaining part (with N being odd) is discarded.

The points on the Ring are numbered, starting at the Direction point, from 0 to M-1 for the upper part of the Ring and from -1 to -M for the lower part of the Ring, with M=N/2.

Figure 29 shows a Ring with Radius R=3 and Angular resolution factor p=0 respectively 1.

```
       8  7  6  5  4                  4    3    2
   9 . .  .  .  .  .  . 3             .    .    .

  10 .                . 2      5 .                 . 1

  11 .                . 1

 -12 .       .        . 0     -6 .         .       . 0
          centre                        centre
 -11 .                . -1

 -10 .                . -2    -5 .                 . -1

  -9 . .  .  .  .  .  . -3            .    .    .
      -8 -7 -6 -5 -4                 -4   -3   -2
```

Figure 29 : Some example Rings with point numbering.

The Direction of a Ring is identified by the position of the point with number ZERO. The initial position of this point is on the positive x-axis, while the Cartesian axes are drawn through the centre point of the Ring. The Direction of the Rings following the initial one is dependent on the direction of the increments. This Direction is determined in the following way:

If P1 is the previous centre point and the current centre point is P2 (P2 is a point on the Ring with the centre in P1). The position of the point with number ZERO on the Ring, with P2 as centre point, is opposite to point P1, this is the Direction of the Ring. So the Direction of the Ring is dependent of the writing direction as indicated by the last increment. The position of the increment on the new Ring (centre P2) is described as the difference between the position of point P2 on the previous Ring and the position of the new point P3 on the current Ring.

In the DCC only the differences between points on the consecutive Rings are coded. Or to state it in another way the Direction of the Ring is dependent on the direction of the line to be displayed. As shown in figure 30, the position of point P3 is defined by the difference: P3 - P2 = -1. P3 and P2 being point numbers on the two Rings, numbered as given in figure 29. The Direction (position of the point with number ZERO) is identified by D.

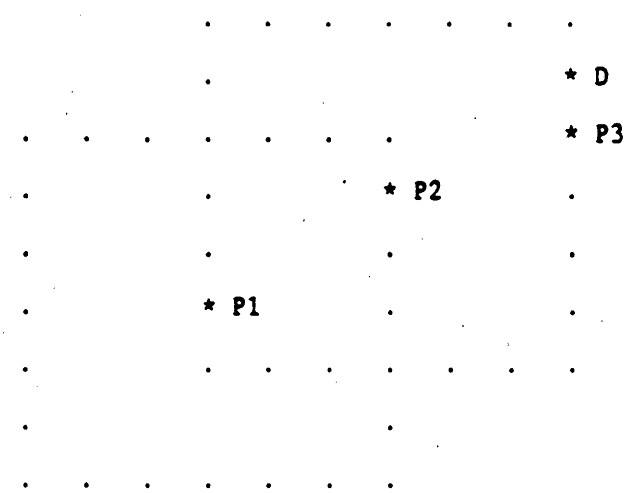


Figure 30 : Change of direction with R=3.

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 Escapes 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 6.

Length	Code-word	Point number
2	00	0
2	10	1
2	01	-1
4	1100	2
4	1101	-2
6	111000	3
6	111001	-3
6	111010	4
6	111011	-4
8	11110000	5
8	11110001	-5
8	11110010	6
8	11110011	-6
8	11110100	7
8	11110101	-7
8	11110110	8
8	11110111	-8
10	1111100000	9
10	1111100001	-9
10	1111100010	10
10	1111100011	-10
10	1111100100	11
10	1111100101	-11
10	1111100110	12
10	1111100111	-12
10	1111101000	13
10	1111101001	-13
10	1111101010	14
10	1111101011	-14
10	1111101100	15
10	1111101101	-15
10	1111101110	16
10	1111101111	-16
10	1111110000	17
10	1111110001	-17
10	1111110010	18
10	1111110011	-18
10	1111110100	19
10	1111110101	-19
10	1111110110	C1
10	1111110111	-20
10	1111111000	C2
10	1111111001	C3
10	1111111010	C4
10	1111111011	C5
10	1111111100	C6
10	1111111101	IM-ESC 1
10	1111111110	IM-ESC 2
10	1111111111	End of Block

Huffman Code table for Incremental mode

Table 6

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 temporarily 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 basic 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 basic Radius 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 basic 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 31

```
                   b8  b7  b6  b5  b4  b3  b2  b1
                  +---+---+---+---+---+---+---+---+
                  | - | 1 | 0 | 1 | 0 | 0 | 0 | 0 |   1st byte
                  +---+---+---+---+---+---+---+---+
      operand flag __|
                extension __|
                flag
                  +---+---+---+---+---+---+---+---+
                  | - | 1 | 0 | 0 | 0 | 0 | 0 | 1 |   2nd byte
                  +---+---+---+---+---+---+---+---+
                extension __|                   |__ DCC flag
                flag
                  +---+---+---+---+---+---+---+---+
                  | - | 1 |   |   |   |   |   |   |   other bytes
                  +---+---+---+---+---+---+---+---+
                          |_______________________|
                           incremental mode data
```

Figure 31 : 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 32.

```
 b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| - | 1 |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+
      |   |_______________________|
      |               |
    oper.          databits
    flag
```

Figure 32 : Bitstream format structure.

The Bitstream format is used to encode:

- Incremental mode coordinates (see section 7.2.2.3.3)
- Colour direct (see section 7.2.3.1)
- Colour index lists (see section 7.2.4.1.1)

Bitstream data are packed in consecutive databits starting from high numbered bits to lower numbered bits of the first byte for the most significant part of the bitstream data.

The end of a Bitstream format operand can not be derived from the Bitstream format itself (the format is not self-delimiting).

- Encoding Incremental mode coordinates, the end of the data (which identifies the end of the Bitstream format operand) is identified by the <End of Block> code.
- Encoding Colour index lists, the number of bits needed to encode the Colour index list (which identifies the end of the Bitstream format operand) is set by the SET COLOUR INDEX PRECISION primitive.
- Encoding Colour direct data, the number of bits needed to encode this data (which identifies the end of the Bitstream format operand) is set by the SET COLOUR HEADER primitive.

7.2.3.1 <u>Colour direct encoding</u>

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 33). The first byte contains the most significant bits. In the last byte, bit b3, b2 and b1 may be left unused.

```
 b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| - | 1 | R | G | B | R | G | B |
+---+---+---+---+---+---+---+---+
      |   |_______|   |_______|
      |       |           |
    oper.  databits    databits
    flag
```

Figure 33 : Colour direct encoding.

Figure 34 gives an example of the colour loading with a unit resolution of 4.

```
+---+---+---+---+---+---+---+---+     +---+---+---+---+---+---+---+---+
| - | 1 | R | G | B | R | G | B |     | - | 1 | R | G | B | R | G | B |
+---+---+---+---+---+---+---+---+     +---+---+---+---+---+---+---+---+

  +---+---+---+---+                binary value of colour
  | R | R | R | R |                component R
  +---+---+---+---+
   b4  b3  b2  b1
      +---+---+---+---+            binary value of colour
      | G | G | G | G |            component G
      +---+---+---+---+
       b4  b3  b2  b1
          +---+---+---+---+        binary value of colour
          | B | B | B | B |        component B
          +---+---+---+---+
           b4  b3  b2  b1
```

Figure 34 : Colour loading with a unit resolution of 4.

7.2.4 Colour lists

A colour list is a structured datatype consisting of a sequence of colour list elements, each element specifying a colour value. The general format of a colour list is:

```
<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:

```
               b8  b7  b6  b5  b4  b3  b2  b1
              +---+---+---+---+---+---+---+---+
              | - | 1 | 0 | 0 | 0 |   |   |   |
              +---+---+---+---+---+---+---+---+
                  |   |           |   |   |__ format flag
    operand ______|   |__ ext.    |   |
    flag                  flag    |   |__ runlength flag
                                  |
                                  |__ mode flag
```

Figure 35 : 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 nov set to ZERO.

In the next sections the different methods of encoding colour lists are described in more detail.

7.2.4.1.1 Individual encoding of colour index lists

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

The colour indices can be encoded using either the Basic format or the Bitstream format.

If Basic format encoding is specified, bit b3 of the first byte set to ZERO, the indices of the Colour index list are encoded using the Basic format described in section 7.2.1 for each separate index.

If Bitstream format encoding is specified, bit b3 of the first byte set to ONE, the indices of the Colour index list are encoded using the Bitstream format described in section 7.2.3. The length (in bits) of each index is identified by the SET COLOUR INDEX PRECISION primitive. These indices are packed in consecutive bits starting from bit b6 of the first Bitstream format byte. If an index does not fit in one byte, the most significant part is packed in the first byte, the remaining part is packed in the second byte and so on.

Some examples are given in figures 36 and 37.

```
                 b8  b7  b6  b5  b4  b3  b2  b1
                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 0 | 0 | 0 | 0 | 0 |   1st byte
                +---+---+---+---+---+---+---+---+
   operand flag __|                   |   |   |__ format flag
       extension __|                  |   |__ runlength flag
       flag                           |__ mode flag

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 0 | 0 | 0 | 1 | 0 |   Colour index 1
                +---+---+---+---+---+---+---+---+

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 0 | 0 | 0 | 1 | 1 |   Colour index 2
                +---+---+---+---+---+---+---+---+

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 0 | 0 | 0 | 1 | 1 |   Colour index 3
                +---+---+---+---+---+---+---+---+

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 0 | 0 | 0 | 1 | 1 |   Colour index 4
                +---+---+---+---+---+---+---+---+

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 0 | 0 | 0 | 1 | 1 |   Colour index 5
                +---+---+---+---+---+---+---+---+

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 0 | 0 | 1 | 0 | 0 |   Colour index 6
                +---+---+---+---+---+---+---+---+

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

Figure 36 : Colour index list individually encoded using the Basic format.

```
                 b8  b7  b6  b5  b4  b3  b2  b1
                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 0 | 0 | 0 | 0 | 1 |   1st byte
                +---+---+---+---+---+---+---+---+
operand flag __|                      |   |   |__ format flag
      extension __|                   |   |__ runlength flag
      flag                            |__ mode flag

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 1 | 0 | 0 | 1 | 1 |   Colour index 1 and 2
                +---+---+---+---+---+---+---+---+

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 1 | 1 | 0 | 1 | 1 |   Colour index 3 and 4
                +---+---+---+---+---+---+---+---+

                +---+---+---+---+---+---+---+---+
                | - | 1 | 0 | 1 | 1 | 1 | 0 | 0 |   Colour index 5 and 6
                +---+---+---+---+---+---+---+---+

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

Figure 37 : Colour index list individually encoded using the Bitstream format.

7.2.4.1.2 Runlength encoding of colour index lists

When using runlength encoding of a colour index list both 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 38.

```
              b8  b7  b6  b5  b4  b3  b2  b1
            +---+---+---+---+---+---+---+---+
            | - | 1 | 0 | 0 | 0 | 0 | 1 | 0 |   1st byte
            +---+---+---+---+---+---+---+---+
operand flag __|                    |   |   |__ format flag
     extension  __|                 |   |__ runlength flag
     flag                           |__ mode flag

            +---+---+---+---+---+---+---+---+
            | - | 1 | 0 | 0 | 0 | 0 | 1 | 0 |   Run colour 1
            +---+---+---+---+---+---+---+---+

            +---+---+---+---+---+---+---+---+
            | - | 1 | 0 | 0 | 0 | 0 | 0 | 1 |   Run count 1
            +---+---+---+---+---+---+---+---+

            +---+---+---+---+---+---+---+---+
            | - | 1 | 0 | 0 | 0 | 0 | 1 | 1 |   Run colour 2
            +---+---+---+---+---+---+---+---+

            +---+---+---+---+---+---+---+---+
            | - | 1 | 0 | 0 | 0 | 1 | 0 | 0 |   Run count 2
            +---+---+---+---+---+---+---+---+

            +---+---+---+---+---+---+---+---+
            | - | 1 | 0 | 0 | 0 | 1 | 0 | 0 |   Run colour 3
            +---+---+---+---+---+---+---+---+

            +---+---+---+---+---+---+---+---+
            | - | 1 | 0 | 0 | 0 | 0 | 0 | 1 |   Run count 3
            +---+---+---+---+---+---+---+---+

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

Figure 38 : Colour index list encoded using runlength encoding.

7.2.4.1.3 <u>Encoding of colour direct lists</u>

In this standard only colour index lists are defined, so the mode flag, bit b3 of the first byte, is always set to ZERO. The encoding of colour direct lists is reserved for future use.

7.2.5 <u>String format</u>

The structure of a String format is given in figures 39 and 40.

```
 b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| - | 0 | 0 | 1 | 1 | 0 | 1 | 1 |   1st byte (ESC)  \
+---+---+---+---+---+---+---+---+                    |
                                                     |- SOS
+---+---+---+---+---+---+---+---+                    |
| - | 1 | 0 | 1 | 1 | 0 | 0 | 0 |   2nd byte (5/8)  /
+---+---+---+---+---+---+---+---+

+---+---+---+---+---+---+---+---+
| - | C | C | C | C | C | C | C |   3rd byte
+---+---+---+---+---+---+---+---+

+---+---+---+---+---+---+---+---+
| - | C | C | C | C | C | C | C |   4th byte
+---+---+---+---+---+---+---+---+
      .                   .
      .                   .
+---+---+---+---+---+---+---+---+
| - | 0 | 0 | 1 | 1 | 0 | 1 | 1 |   (n-1)-th byte (ESC) \
+---+---+---+---+---+---+---+---+                        |
                                                         |- ST
+---+---+---+---+---+---+---+---+                        |
| - | 1 | 0 | 1 | 1 | 1 | 0 | 0 |   n-th byte (5/12)    /
+---+---+---+---+---+---+---+---+
```

Figure 39 : String format structure in a 7-bit environment.

Databits are marked C.

```
 b8  b7  b6  b5  b4  b3  b2  b1
+---+---+---+---+---+---+---+---+
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |   1st byte (SOS)
+---+---+---+---+---+---+---+---+

+---+---+---+---+---+---+---+---+
| C | C | C | C | C | C | C | C |   2nd byte
+---+---+---+---+---+---+---+---+

+---+---+---+---+---+---+---+---+
| C | C | C | C | C | C | C | C |   3rd byte
+---+---+---+---+---+---+---+---+
      .                   .
      .                   .
+---+---+---+---+---+---+---+---+
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |   n-th byte (ST)
+---+---+---+---+---+---+---+---+
```

Figure 40 : String format structure in a 8-bit environment.

Databits are marked C.

The number of bytes needed to encode a string operand equals to the number of characters of the string plus four (for SOS and ST, coded 1/11, 5/8 and 1/11, 5/12) in a 7-bit environment and equals to the number of characters of the string plus two (for SOS and ST, coded 9/8 and 9/12) in a 8-bit environment.

The encoding of a string is the only exception of the general coding rules indicated in section 7.2.

7.2.6 Record format

The structure of 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 ESCAPE1	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	-

Primitives	Opcode Byte 1	Byte 2
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
CLIPPING PRIMITIVES		
SET CLIPPING RECTANGLE	3/2	2/3

Primitives	Opcode Byte 1	Opcode Byte 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 occurence
<symbols> (n)	= n or more occurences (with n >= 1)
<symbols> (=n)	= n occurences (with n >= 1)
<symbols> (0)	= optional, 0 or more occurences
[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 point>

<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 : deltax> [DX]

<integer : deltay> [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>

<u>with</u>

<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 <u>SET TEXT INDEX</u>

<SET TEXT INDEX opcode: 3/1, 3/6>

<index :text index>

8.2.2.2.21 <u>SET TEXT FONT AND PRECISION</u>

<SET TEXT FONT AND PRECISION opcode: 3/1, 3/4>

<integer : text font>

<enumerated : text precision>

<u>with</u>

<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 <u>SET CHARACTER EXPANSION FACTOR</u>

<SET CHARACTER EXPANSION FACTOR opcode: 3/1, 3/0>

<real : character expansion factor>

8.2.2.2.23 <u>SET CHARACTER SPACING</u>

<SET CHARACTER SPACING opcode: 3/1, 3/3>

<real : character spacing>

8.2.2.2.24 <u>SET TEXT COLOUR INDEX</u>

<SET TEXT COLOUR INDEX opcode: 3/1, 2/15>

<colour index : text colour index>

8.2.2.2.25 <u>SET TEXT PATH</u>

<SET TEXT PATH opcode: 3/1, 3/2>

<enumerated : text path>

<u>with</u>

<enumerated : text path> =

<enumerated: 0> [RIGHT]

|<enumerated: 1> [LEFT]

|<enumerated: 2> [UP]

|<enumerated: 3> [DOWN]

|<enumerated: >3> [reserved]

8.2.2.2.26 <u>SET CHARACTER VECTORS</u>

<SET CHARACTER VECTORS opcode: 3/1, 3/1>

<point : height vector>

<point : width vector>

8.2.2.2.27 <u>SET TEXT ALIGNMENT</u>

<SET TEXT ALIGNMENT opcode: 3/1, 3/5>

<enumerated : horizontal alignment>

<enumerated : vertical alignment>

<u>with</u>

<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 <u>SET COLOUR REPRESENTATION</u>

<SET COLOUR REPRESENTATION opcode: 3/2, 3/8>

<identifier : workstation identifier>

<colour index : starting entry in the colour table>

<colour direct : colour data> (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 opcode: 3/3, 2/8>

<identifier : segment name>

8.2.3.1.6 REDRAW ALL SEGMENTS ON WORKSTATION

<REDRAW ALL SEGMENTS ON WORKSTATION opcode: 3/3, 2/9>

<identifier : workstation identifier>

8.2.3.1.7 SET HIGHLIGHTING

<SET HIGHLIGHTING opcode: 3/3, 2/6>

<enumerated : highlighting flag>

<identifier : segment name>

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 VISS 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 <u>SET COLOUR HEADER</u>

<SET COLOUR HEADER opcode: 3/2, 2/8>

<integer : unit resolution>

<enumerated : coding method>

<u>with</u>

<enumerated : coding method> =

<enumerated: 1> [RGB-coding]

|<enumerated: >1> [reserved]

8.2.6.3 <u>SET COORDINATE PRECISION</u>

<SET COORDINATE PRECISION opcode: 3/2, 2/9>

<integer : magnitude code>

<integer : granularity code>

<integer : default exponent>

<enumerated : explicit exponent allowed>

<u>with</u>

<enumerated: 0> [ALLOWED]

|<enumerated: 1> [FORBIDDEN]

8.2.6.4 <u>SET REAL PRECISION</u>

<SET REAL PRECISION opcode: 3/2, 3/9>

<integer : magnitude code>

<integer : granularity code>

<integer : default exponent>

<enumerated : explicit exponent allowed>

<u>with</u>

<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**max(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	
Deltax	1
Deltay	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
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

(2604)

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 9).
WORKSTATION WINDOW *)	(0.0, 0.0) and (1.0, 1.0)

*) 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.

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.

APPENDIX A

PRIMITIVES, WORKSTATION CATEGORIES, LEVELS AND OPTIONS

This appendix forms an integral part of the standard.

This attached table lists the primitives contained in the document.

For each 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

PRIMITIVES; WORKSTATION CATEGORIES, LEVELS AND OPTIONS

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 ESCAPE1	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
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

PRIMITIVES, WORKSTATION CATEGORIES, LEVELS AND OPTIONS

PRIMITIVE	LEVEL	APPLIES TO		OPTION
SET CHARACTER SPACING	L0a	SS	0	M
SET TEXT COLOUR INDEX	L0a	SS	0	M
SET TEXT PATH	L0a	SS	0	M
SET CHARACTER VECTORS	L0a	SS	0	M
SET TEXT ALIGNMENT	L0a	SS	0	M
SET COLOUR REPRESENTATION	L0a		0	M
SET ASPECT SOURCE FLAGS	L0a	SS	0	M
SET WORKSTATION WINDOW	L0a		0	M
SET WORKSTATION VIEWPORT	L0a		0	M
SET CLIPPING RECTANGLE	L0a	SS	0	M
UPDATE WORKSTATION	L0a		0	M
SET DEFERRAL STATE	L1a		0	M
EMERGENCY CLOSE	L0a	SS	0	M
CREATE SEGMENT	L1a	SS	0	M
CLOSE SEGMENT	L1a	SS	0	M
RENAME SEGMENT	L1a	SS	0	M
DELETE SEGMENT FROM WORKSTATION	L1a	SS	0	M
DELETE SEGMENT	L1a	SS	0	M
REDRAW ALL SEGMENTS ON WORKSTATION	L1a		0	M
SET HIGHLIGHTING	L1a	SS	0	M
SET VISIBILITY	L1a	SS	0	M
SET SEGMENT TRANSFORMATION	L1a	SS	0	M
SET SEGMENT PRIORITY	L1a	SS	0	N
ASSOCIATE SEGMENT WITH WORKSTATION	L2a	SS	0	M
COPY SEGMENT TO WORKSTATION	L2a	SN	0	M
INSERT SEGMENT	L2a	SS	0	M
SET COLOUR HEADER	L0a	SS	0	N
SET DOMAIN RING	L0a	SS	0	M
SET COORDINATE PRECISION	L0a	SS	0	M
SET REAL PRECISION	L0a	SS	0	M
SET COLOUR INDEX PRECISION	L0a	SS	0	M

APPENDIX B

B-SPLINE CURVES AND ELLIPSES

B.1 SHORT NOTE ON B-SPLINE CURVES

A spline is a piecewise polynomial function passing through a set of points called knots (see figure B.1).

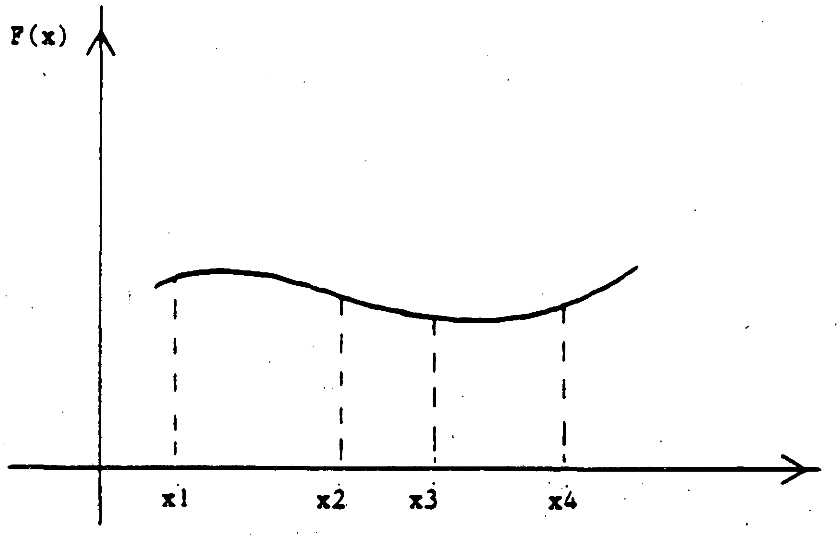


Figure B.1 : A spline passing through 4 knots

The values X1, X2, ..., Xk 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 <u>uniform</u> 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 Nim is a m-degree B-spline.

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

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

The corresponding knots are:

$$\frac{P_i + P_{i+1}}{2} \qquad\qquad \frac{P_{i+1} + P_{i+2}}{2}$$

An example of such a curve is given in figure B.2. Knots are located on the middle of the segments joining the breakpoints and the curve is tangent to the segment at this point.

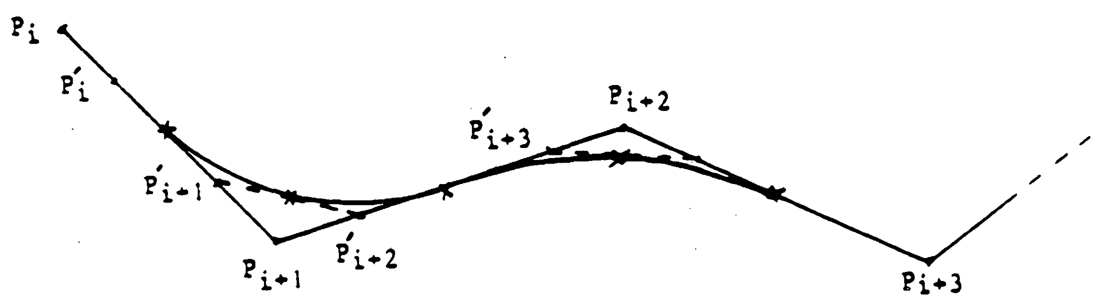


Figure B.2 : Uniform quadratic B-spline curve

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:

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

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

The nev guiding points vill 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 nev guiding points are closer to the curve than the former ones (see figure B.2).

By simply repeating this procedure, until the curve segmen:s reach the size of a pixel, the spline curve can be dravn. 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 end-points of the curve are no guiding points but knots.

The coordinates as specified in the GDP(spline)-primitive vill 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., R.F. RIESENFELD. "A theoretical Development for the Computer Generation and Display of Piecevise Polynomial Surfaces". I.E.E.E. Trans. on P.A.M.I. - Vol. PAMI - 2, No 1 (Jan 80) 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.

B.2 ELLIPSE PRIMITIVES

A central question is how to represent a generally oriented ellipse such that the necessary properties of the picture are preserved across all graphical transformations. Unfortunately, the major and minor axes cannot be used for an ellipse in a general orientation since, as shown in figure B.3 below, these axes do not remain mutually orthogonal (do not remain axes) across a scaling transformation which does not preserve aspect ratio.

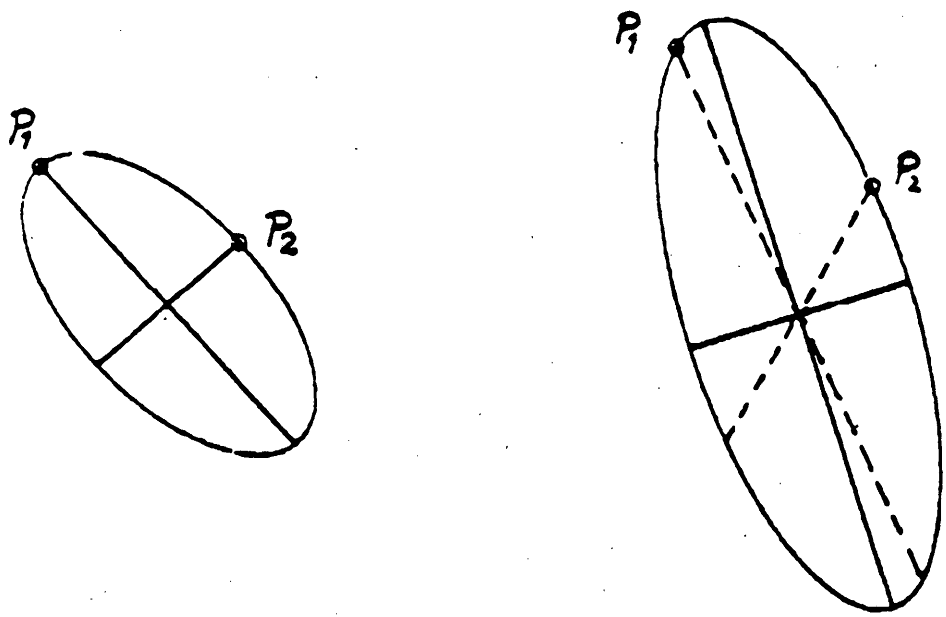


Figure B.3 :The scaling of an ellipse and its axes such that X'=X and Y'=2Y

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 B.4 in which the ellipse has been scaled by a factor of two in the Y-direction only.

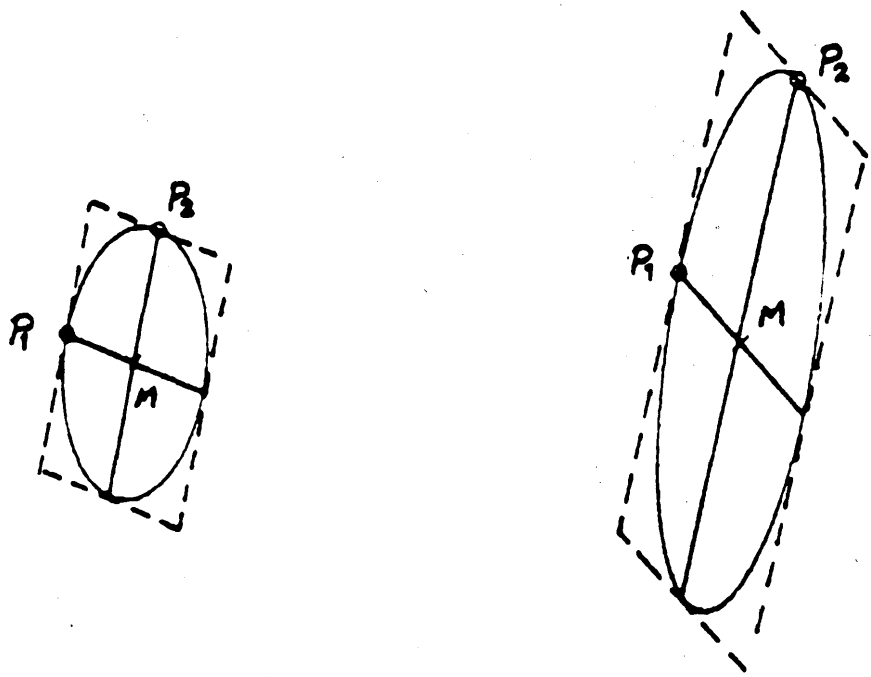


Figure B.4 :Ellipses defined by a Conjugate Diameter Pair

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 C

CROSS REFERENCES

C.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

Primitives	Chapter number Description	Encoding
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
CLIPPING PRIMITIVES		
SET CLIPPING RECTANGLE	6.2.2.4.1	8.2.2.4.1

Primitives	Chapter number Description	Encoding
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

C.2 ALPHABETIC ORDER

Primitives	Chapter number	
	Description	Encoding
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 D

SELECTION OF THE GEOMETRIC DISPLAY

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

<US> <3/2> <y>

as described in part 0 of T/CD 6.1. 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 D-1.

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

Table D.1

Relation between <y> code and levels

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

CONTENTS

Note: Further study is awaiting the outcome from other bodies (CCITT, ISO, ESPRIT) involved in this area.

LIST OF FIGURES

1.0 INTRODUCTION

The photographic facility allows for the transmission and display of an image consisting of individually defined picture elements (pixels) with many grey/colour levels. The image may be subjectively similar to a still broadcast quality television picture. Colour television techniques may be used to define the image and digital signal processing techniques may be used to compress the image for storage and transmission.

The protocol allows for many different photographic videotex systems to be specified but recommendations are given based on the CCIR digital television studio standard (Recommendation 601).

A transmission mode allowing all the presentation level bits to be used for photographic data (transparent mode) provides an efficient means of transmitting the relatively large amounts of data needed for photographic images.

1.1 Protocol Principles

The transmission of a photographic image is accomplished using two Videotex Presentation Data Elements (VPDEs). For each VPDE type there are two subtypes; a header and a tranfer unit. Their functions are outlined below.

Pixel header unit - gives the parameters defining the composition of the displayed image and the method of coding used for storage and transmission.

Pixel transfer unit - contains the actual data describing the image (grey/colour levels of each pixel).

Table header unit - specifies the type and format of tables used in in the decoding process (eg set up a quantiser).

Table transfer unit - contains the actual values to be loaded into the tables (eg fill the quantiser).

1.2 Coding/Protocol Definitions

In this part of the recommendation the following definitions apply:

PHOTOGRAPHIC PARAMETER
A photographic parameter is a quantity that conveniently characterises a particular aspect of the transmission or display of the photographic image (eg display resolution).

PARAMETER FIELD
A parameter field is the complete coding specification of a parameter. It consists of a parameter identifier and one or more parameter values.

PARAMETER IDENTIFIER
A parameter identifier introduces a parameter field and defines the particular parameter being specified.

COMPONENT
Certain aspects of the display or transmission may have to split into separate parts, these are referred to as components. (eg colour components Y, U and V).

DELIMITER
A delimiter may be used to separate parameter values or data for different components.

1.3 Coding Principles

The coding scheme provides for unambiguous identification of videotex control codes (columns 0, 1), photovideotex parameters (columns 2, 3) and allows 6 bits (columns 4, 5, 6, 7) to be used for data. A diagram of the code table is shown in Figure 1.

Column 2 - indentifies the parameter being specified, see Figure 2

Column 3 - is used to specify
- a value in decimal form
- to give a 'type number'
- to separate parameters/data for different components.

Codes 3/0 to 3/9 represent decimal values 0 to 9.

3/11 is used to delimit decimal values. Leading zeroes may be omitted.

3/12 may be used if desired within a parameter to separate different components.

If a default value exists for a parameter it is assigned the 3/0 type value.

When a parameter has several components the values for each component are specified in sequence. If the value of a trailing component(s) is(are) the same as the previous values then it(they) may be omitted. If a whole parameter is omitted then it is assumed
- that its value is implicit from other information,
- it is a default value or
- the parameter is not applicable in this particular case.

Data may be coded using columns 4, 5, 6 and 7 of the code table. The protocol also allows for all code bits received at the presentation level to be used for data (transparency).

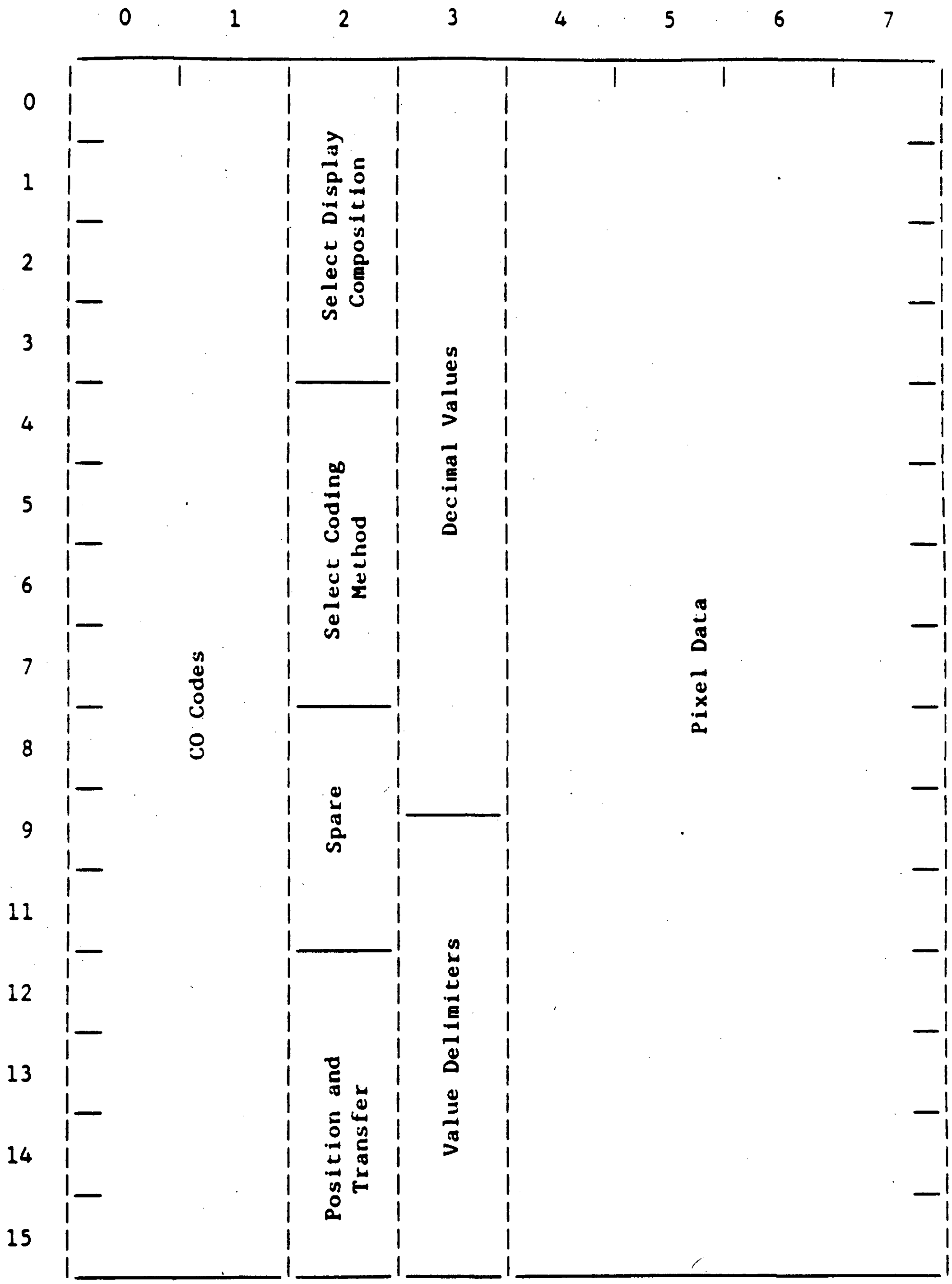


FIGURE 1 PHOTOVIDEOTEX CODE TABLE

	PARAMETER CODE		DEFAULT
SELECT DISPLAY COMPOSITION	2/0	Components	(Y, U*, V*)
	2/1	Resolution	(4:2:2)
	2/2	Bits/Pixel/Component	(8/8/8)
	2/3	Sampling Structure	(Orthogonal Coincident)
SELECT CODING METHOD	2/4	Image Coding	(Linear PCM)
PIXEL TRANSFER	2/12	Origin	(Top Left of Defined Display Area)
	2/13	Area	(Defined Display Area)
VALUES	3/0	0	
	3/1	1	
	3/2	2	
	3/3	3	
	3/4	4	
	3/5	5	
	3/6	6	
	3/7	7	
	3/8	8	
	3/9	9	
DELIMITERS	3/11	Decimal Delimeter	
	3/12	Component Delimeter	

FIGURE 2 PIXEL PARAMETER CODES AND DEFAULTS

* CCIR colour difference signals.

2.0 PHOTOGRAPHIC PIXEL HEADER UNIT

The header unit will take the form

US 3/4 2/0 <SDC> <SCM>

The definitions of parameter fields above are given below. The header applies for all following pixel transfer units until another header is sent or until the end of the session.

2.1 <SDC> : Select Display Composition

This field specifies the composition of the photographic display. It can contain up to 4 parameters.

<SDC> : <COM> <RES> <BPC> <STR>

2.1.1 <COM> : Display Components

A displayed image may be formed from one or more components. For a monochrome image only one component is needed but colour requires three. Sets of different component possibilities are given in the table below and a code is assigned to each.

<COM> = display component identifier, component type number

= 2/0 3/C

3/C = 3/0 colour YU*V*
3/1 monochrome
:
: (for later
3/15 (allocation

2.1.1.1 CCIR Monochrome And Colour Component

A colour image is defined as being comprised of a luminance (brightness) and a pair of colour difference (colouring) components. A monochrome image contains the luminance (brightness) component only.

The luminance signal is obtained from gamma-corrected primary signals, R,G,B and corresponds to the equation

$$Y = 0.299R + 0.587G + 0.114B$$

* CCIR colour difference signals.

The colour-difference signal components are then defined as:

$$R\text{-}Y = 0.701R - 0.587G - 0.114B$$

$$B\text{-}Y = -0.299R - 0.587G + 0.886B$$

The colour-difference signals have a range of 0.701 to -0.701 for R-Y, and 0.886 to -0.886 for B-Y. To restore the signal excursion of the colour-difference signals to unity (i.e. +0.5 to -0.5), coefficients are applied to the R-Y and B-Y. The weighted colour-difference signals U* and V* are then defined as

$$U^* = 0.564(B\text{-}Y)$$

$$V^* = 0.713(R\text{-}Y)$$

2.1.2 <RES> : Display Resolution

Display resolution is defined as the number of pixels horizontally and vertically in the defined area.

The CCIR recommendation 601 specifies for the digital television studio standard a 13.5 MHz luminance sampling frequency and a 6.75 MHz chrominance sampling frequency for 625 and 525 line systems. The actual number of pixels is dependent on the size of the area. For this existing standard and for other systems based on it a shorter coding can be used to specify the horizontal and vertical resolution. The CCIR nomenclature for sampling frequencies is used where the frequencies of the three components are expressed in sequence and relative to 3.375MHz (eg 13.5 / 6.75 / 6.75 MHz = 4:2:2).

<RES> = resolution ident, resolution type number

= 2/1 3/R

3/R = 3/0 4:2:2 (CCIR studio standard)
3/1 2:1:1

: (for later allocation)
3/15 Decimally defined value - see below

2.1.2.1 Decimally Defined Resolution

Other resolutions may be specified if required by specifying in decimal form the number of pixels horizontally and vertically. If the pixel is formed from more than one component the resolution of each component is specified in sequence in descending order of resolution. The highest resolution component is referred to as the first component. The resolution of the other lower resolution components are specified as a fraction of the resolution of the first component and are coded as the reciprocal of the fraction (eg 1/4 is specified as 4).

<RES> = resolution ident, no of horiz pixels, no of vert pixels

```
= 2/1  3/15  ...3/uHa  3/11  ...3/uVa  3/11    (1st component)
                ...                                :
             ...3/uHn  3/11  ...3/uVn  3/11    (nth component)
```

2.1.3 <BPC> : Bits per Display Component

This parameter gives the number of grey or colour levels a pixel may have. The number of levels available for each component is expressed in terms of the number of bits of storage per pixel per component if stored in an uncompressed PCM form. Normally this will be a value in the range 1 to 9 and can be specified by a single code value.

<BPC> = bits/pixel/comp ident, no of bits/pixel/component

```
= 2/2  ...3/Ba  3/11          (1st component)
          ...                        :
       ...3/Bn  3/11          (nth component)

where:     3/B = 3/0   8 bits/pixel (default)
                 3/1   1 bit/pixel
                 3/2   2 bits/pixel
                  :
                  :
                 3/9   9 bits/pixel

                 3/15  Decimally defined value - see below
```

2.1.3.1 Decimally Defined Bits per Display Component

The number of bits per component may if necessary be specified in full decimal form.

<BPC> = bits/pixel/comp ident, no of bits/pixel/component (in decimal form)

```
= 2/2  3/15  ...3/uBa  3/11          (1st component)
                ...                         :
             ...3/uBn  3/11          (nth component)
```

2.1.3.2 CCIR Level Assignment

The CCIR recommendation defines certain reference binary levels for a uniformly quantised pcm image having 8 bits per sample. Luminance samples are represented by a positive binary number and colour difference samples by a offset binary number. The total nominal excursion of the luminance signal corresponds to 220 quantisation levels, with black corresponding to level 16, and nominal white to level 235 (Figure 3). There is an unequal quantisation margin above and below the nominal signal, because there is a greater variation in the nomiral white level than in the nominal black level and the effect of clipping the overshoot will be more preceptible in the white region.

Given that the luminance signal is to occupy 220 levels and that black is to be at level 16, the digital luminance signal Yd may be calculated by

$$Yd = 219Y + 16$$

Y is the luminance analogue signal of any colour, expressed as a fraction of unity.

Yd is the corresponding level number after quantisation to the nearest integer value.

The colour-difference signals each occupy 225 levels in the central part of the quantisation scale, with zero signal corresponding to level 128 (Figure 3).

LEVEL		BINARY	HEX
255		(11111111)	FF
235	White	(11101011)	EB
16	Black	(00010000)	10
0		(00000000)	0

LUMINANCE CODING RANGE

LEVEL		BINARY	HEX
255		(11111111)	FF
240	Maximum	(11110000)	F0
128	Zero	(10000000)	80
16	Minimum	(00010000)	10
0		(00000000)	0

COLOUR DIFFERENCE CODING RANGE

FIGURE 3 LEVEL ASSIGNMENT

Given that the colour-difference signals are to occupy 225 levels and that zero level is to be 128, the decimal values of the colour-difference signals. V*, U* may be calculated

V*d = 224[0.713(R-Y)]+128

U*d = 224[0.564(B-Y)]+128

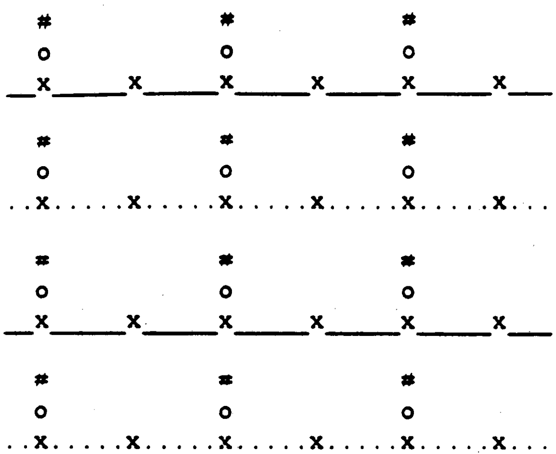
V*d U*d are the corresponding level numbers after quantisation to the nearest integer value.

R-Y, B-Y are the colour-difference analogue values of any colour expressed as a fraction of unity.

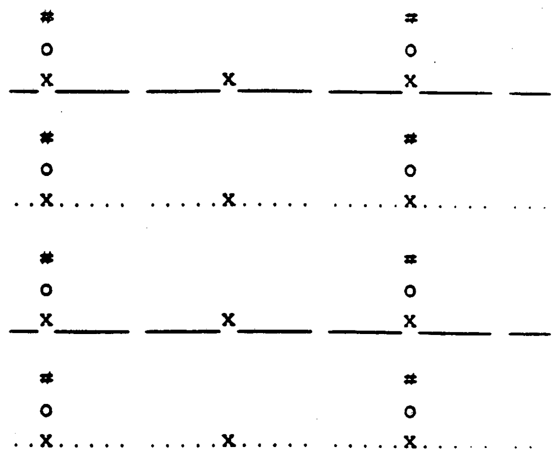
2.1.4 <STR> : Sampling Structure

The structure parameter defines the spatial and temporal relationship between pixels on adjacent lines and fields, see Figure 4 (Sampling Structures). The relationship between samples of the first component is specified first followed by the relative structure of the other components to the first.

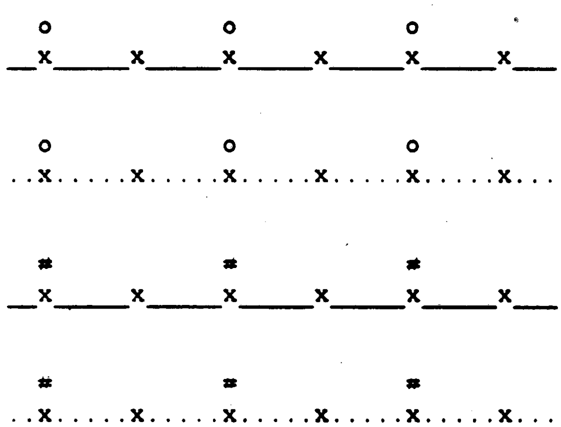
```
<STR> = structure ident, structure type number(s)

      = 2/3  3/S  3/R

             3/S = 3/0  line orthogonal field orthogonal  )
                   3/1  line orthogonal field quincunx    ) interlaced
                   3/2  line quincunx field orthogonal    )
                   3/3  line orthogonal single field
                   3/4  line quincunx single field
                    :
                    :   (for later
                   3/15 (allocation

             3/R = 3/0  coincident
                   3/1  alternate samples
                   3/2  sequential line
                    :
                    :   (for later
                   3/15 (allocation
```



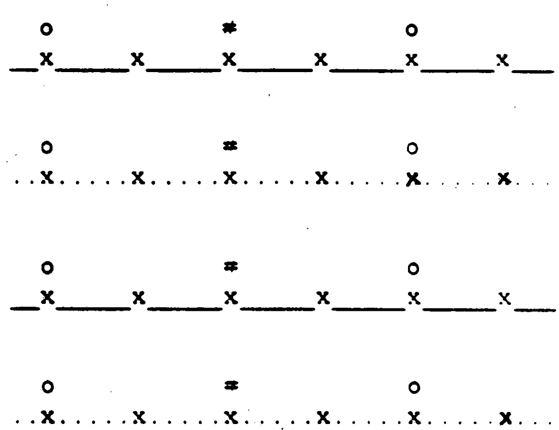
Field Orthogonal and Line Orthogonal
Coincident Samples (CCIR mode 4.2.2)



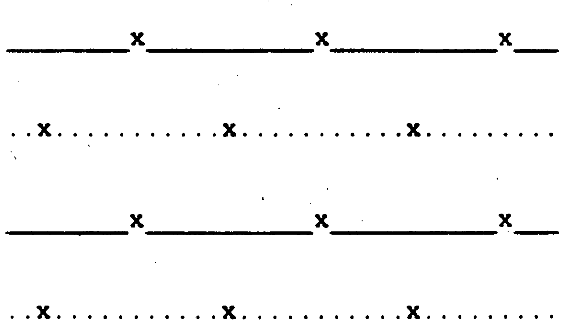
Field Orthogonal and Line Orthogonal
Coincident Samples (CCIR mode 2.1.1)



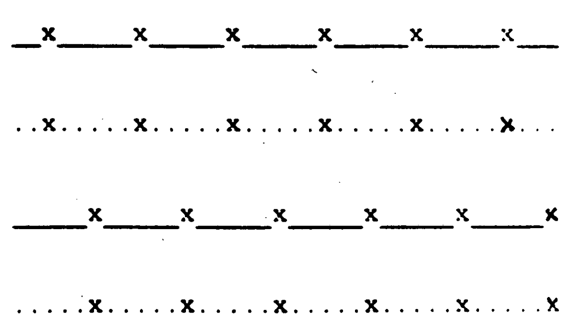
Field Orthogonal and Line Orthogonal
Sequential Samples



Field Orthogonal and Line Orthogonal
Alternate Samples



Field Quincunx and Line Orthogonal
Coincident Samples



Field Orthogonal and Line Quincunx
Coincident Samples

x	1st component	Y
o	2nd component	U
#	3rd component	V

_____	Field 1
.....	Field 2

FIGURE 4 SAMPLING STRUCTURES

2.2 <SCM> : Select Coding Method

The way in which data is coded for storage and/or transmission is described in this section. At present only one parameter field is specifiable, other parameters are given in the descriptions of particular coding techniques. Later when other generally useful parameters have been identified these may be made individually specifiable.

<SCM> : <ICT>

2.2.1 <ICT> : Image Coding Techniques

A photographic image is normally encoded using digital signal processing techniques (eg pulse code modulation or a mathematical tranform). The various methods offer advantages such as high compression, a desirable image build-up, good quality, or be suitable for a certain type of image. A table of image coding techniques is given and a code is assigned to each. A subtype may be used to distinguish between different techniques of the same type. Each technique may have an independent set of subtypes specified in a list.

<ICT> = coding ident, coding type no

= 2/4 <TY> <STY> <SSTY>

```
<TY> : Type             <STY> : Subtype        <SSTY> : Subsubtype

3/1  dpcm ------------ 3/0   one dimensional
                        |     (Appendix A)
                        |

3/2  transform ------- 3/1   Cosine  --------- 3/0   two dimensional
                        |                   |         (Appendix B)
 :                      |                   |
 :
     ( for future
3/15 ( allocation
```

The details of the recommended coding methods are given in the appendices. A particular coding technique may implicitly specify certain parameters such as the number of bits per sample or the sequence for transmission.

3.0 PHOTOGRAPHIC PIXEL TRANSFER UNIT

This VPDE carries the data defining the grey/colour level of the pixels forming the photographic image and specifies where the image is to be located on the display. It takes the form

US 3/4 <ORG> <ARE> <DAT>

3.1 <ORG> : Origin

The origin is the first pixel position to which the data following will refer. This is the top left corner of the rectangular area as defined below. It is specified in terms of the horizontal and vertical pixel position of the first component with respect to the Defined Display Area. (see 2.1.2). See Figure 5 for Full Screen, Origin and Defined Display Area relationships.

<ORG> = origin ident, horiz pix pos, vert pix pos

= 2/12 ...3/uX 3/11 ...3/uY 3/11

```
 _______________________________________
/                                       \
|  +...............................  |
|  :Pixel 1,1                     :  |
|  :                DX            :  |
|  :       Origin +.............  :  |<-----Full Visible Display
|  :              |            |  :  |
|  :              |   Photo    |  :  |
|  :            DY|   Area     |  :  |
|  :              |            |  :  |
|  :              '............'  :  |
|  :                              :  |
|  :                      <------------- Defined Display Area
|  :                              :  |
|  '..............................'  |
\_______________________________________/
```

FIGURE 5 DISPLAY AREAS

3.2 <ARE> : Area

This defines a rectangular area to be filled by the photographic data following. The width and height of the area are specified in terms of the number of pixels of the first component. A non-rectangular image can be constructed from small rectangles, if desired of only one line's height. Alternatively a mask of the required shape may be created on an outer layer.

<ARE> = area ident, area width, area height

= 2/13 ...3/uAW 3/11 ...3/uAD 3/11

When an area has been completely filled by all components the origin is assumed to be set back to the origin of the area.

Other methods of defining the area to be filled by the image are for further study.

3.3 <DAT> : Pixel Data

3.3.1 Within The Code Table

Following the introduction variables described above any codes from columns 4, 5, 6 or 7 will be automatically interpreted as data, the least significant 6 bits of each code being considered as a continuous bit stream containing concatenated data values.

<DAT> = 4/H 5/H 6/H 7/H ...

3.3.2 Transparent Data

As photographic images contain a relatively large amount of data it is desirable for increased efficiency to use all the presentation level code bits for actual data (8 bits per character). In such a mode all codes pass uninterrupted by the normal presentation level control codes and the mode is thus termed transparent.

The transparent mode is entered using the TRANSPARENT data VPDE (see Part 7).

4.0 PHOTOGRAPHIC TABLE HEADER UNIT

For certain photovideotex schemes various tables are needed in the decoding process whose contents have to be changed for different images. The table header unit allows for a set of tables to be specified. Figure 6 shows examples of table structures.

The header unit will take the form:

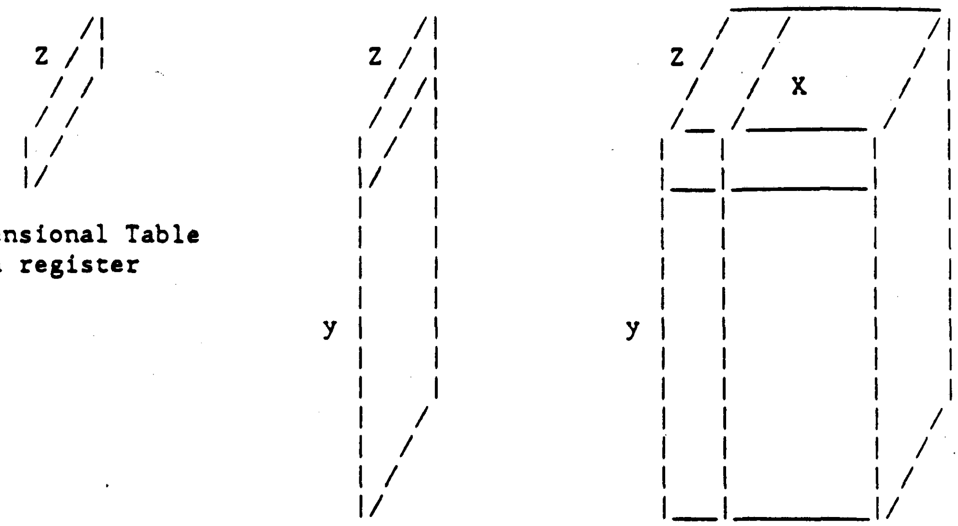
US 3/5 2/0 <SET> <SIZ>

4.1 <SET> : Table Set

The use to which this table is put is defined within a particular coding technique description. The parameter field specifies the table type identity number and the number of tables of that type required. The parameter field is defined within the particular coding technique.

<SET> = set ident, table type no, no of tables

= 2/1 ...3u/T 3/11 ...3/uN 3/11



One Dimensional Table
eg a register

Two Dimensional Table
eg a quantiser

Three Dimensional Table
eg a map

FIGURE 6 PARAMETER TABLE STRUCTURE

4.2 <SIZ> : Table Size

A generalised table is defined in three dimensionsal form. If a table has only two dimensions (eg a quantiser) the third dimension is omitted.

<SIZ> = table size ident, table depth (Z), height (Y), width (X)

= 2/2 ...3/uZ 3/11 ...3/uY 3/11 ...3/uX 3/11

5.0 PHOTOGRAPHIC TABLE TRANSFER UNIT

A table transfer unit is used to fill a previously defined table(s).

US 3/5 2/0 <ID> <LOC> <DAT>

5.1 <ID> : Identity

A particular set of tables is identified using its table set type number and table number within the set. Where there are a number of tables of a given type they will be filled in sequence. If only one value is given it is assumed to be the first table of the set that is addressed.

<ID> = table identification ident, set type no, table no

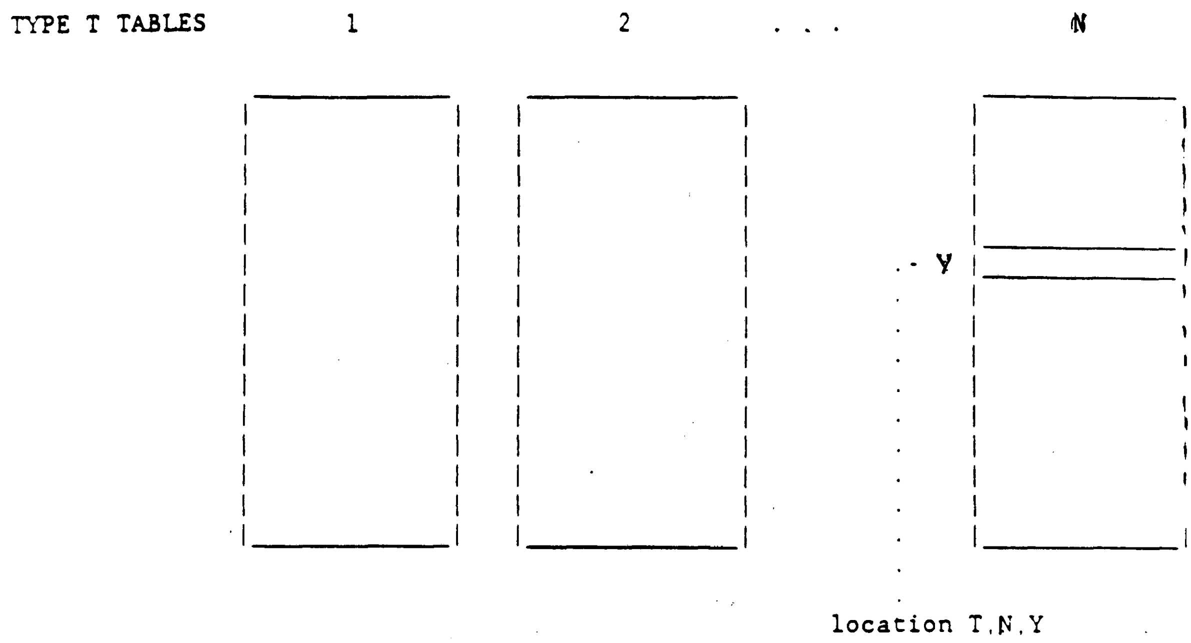
= 2/1 ...3/uT 3/11 ...3/uN 3/11

5.2 <LOC> : Location

A particular location or bit within a table may be addressed by using its XYZ coordinates as used for the table dimensions. See Figure 7

<LOC> = location ident, XYZ address

= 2/2 ...3/uX 3/11 ...3/uY 3/11 ...3/uZ 3/11



A SET OF N TYPE T TABLES EACH OF SIZE X BY Y

FIGURE 7 PARAMETER TABLE ADDRESSING

5.3 <DAT> : Data

5.3.1 Within The Code Table

Table data is coded using the 6 least significant bits of codes from columns 4,5,6,and 7.

<DAT> = 4/H 5/H 6/H 7/H ...

5.3.2 Transparent Data

A table may be filled using a transparent mode as described in section 3.3.2.

APPENDIX A

DPCM IMAGE CODING - ONE DIMENSIONAL

A.1 INTRODUCTION

Differential pulse code modulation (DPCM) is a digital signal coding technique in which the differences between adjacent digitally encoded sample values are used for storage and/or transmission. DPCM is also referred to as predictive coding since the difference between the present value and a predicted value may be used.

A coding scheme may be derived using a difference value based on a one dimensional or two dimensional prediction. The quantiser relating difference codes to actual value changes may be fixed or adaptive (ie changed according to image characteristics).

A.2 ONE DIMENSIONAL - PREVIOUS ELEMENT PREDICTION DPCM

A.2.1 General

This scheme provides for 50% data compression. It is relatively simple to decode, allowing for cheap and/or high speed decoding. The image is built up pixel by pixel, line by line in two scans. The first scan fills the whole area with a monochrome picture which is then coloured in the second scan.

A.2.2 Compression Technique

The pixel colour information is described using the television signal components luminance (Y) and chrominance (U and V). The prediction for the next sample value (Pn+1) is that it will be the same as the present value on the same TV line. The prediction is reset at the start of each line to a mid-range value. For luminance and chorminance this is the value 128 (decimal). The difference or error (D) between the prediction (P) and the actual value (V) is coded and used for storage and/or transmission.

$$\text{Transmitted value } D_{n+1} = V_{n+1} - P_{n+1} = V_{n+1} - V_n$$

Data compression is achieved by using a non-uniform quantiser. A 16 level quantiser is used and so 4 bits/sample are transmitted. The scheme is non-adaptive and thus the quantiser is fixed as shown below. The same quantiser is used for luminance and both chrominance signals.

The value for display is reconstructed by adding the difference/error to the prediction:

Display value $V_{n+1} = P_{n+1} + D_{n+1} = V_n + D_{n+1}$

INPUT DIFFERENCE	TRANSMITTED CODE	OUTPUT DIFFERENCE (D)
0-2	0	1
3-6	1	4
7-12	2	9
13-21	3	16
22-35	4	27
36-61	5	44
62-99	6	79
100-255	7	120
-(1-2)	8	-1
-(3-6)	9	-4
-(7-12)	10	-9
-(13-21)	11	-16
-(22-35)	12	-27
-(36-61)	13	-44
-(62-99)	14	-79
-(100-255)	15	-120

A.2.3 Transmission Sequence

The luminance values for the area are transmitted first. Each byte contains two samples. Starting at the origin, values are sent in sequence pixel by pixel, line by line. When the area has been filled by one component the next pixel position is reset to the origin of the area. The chrominance values are sent in UV pairs in sequence pixel by pixel, line by line. Again two samples are sent per byte.

```
Y11 Y12 ........        Y1N
Y21 Y22 ........        Y2N
    :                    :
YM1 YM2 ........        YMN
UV11 UV13 ......        UV1N/2
UV21 UV23 ......        UV1N/2
    :                    :
UVM1 UVM3 ......        UVMN/2
```

A.3 **EXAMPLE OF CODING FOR DPCM**

Photographic Data Header Unit

This will normally only be sent once at the start of a photovideotex session.

```
VPCE : US  3/4  2/0       (photgraphic data header unit)

VSCE : 2/1   3/1          (display composition = 2:1:1)
       2/3   3/3          (structure = line orthogonal single field)
       2/4   3/1  3/0     (coding technique = DPCM - one dimensional)

       Defaults not transmiited - components      = Y,U*,V*
                                  bits /component = 8/8/8
```

Photographic Data Transfer Unit

This unit is sent for each photovideotex image.

```
VPCE : US  3/4                       (photographic data transfer unit)

PDSU : 2/12  3/hX  3/tX  3/uX  3/11  3/hY  3/tY  3/uY  3/11
      not 2/0                        (origin pixel location = htuX  htuY)

     : 2/13  3/hW  3/tW  3/uW  3/11  3/hH  3/tH  3/uH  3/11
                                     (area width and height = htuW x htuH)

      US  3/15  N  Y  Y  Y  Y  ...  (transparent data for N bytes)
                                     luminance data
       :
       :
       :
      US  3/15  N  Y  Y  Y  Y  ...

      US  3/15  N  U*  V*  U*  V*  ...
                                     chrominance data
       :
       :
       :
      US  3/15  N  U*  V*  U*  V*  ...
```

APPENDIX B

DCT IMAGE CODING - TWO DIMENSIONAL

B.1 INTRODUCTION

The main steps for image coding and decoding on a transform basis are shown in Figure A-1.

```
 Transmitted Image
  ____________________
 |                    |
 |   sub-image f(j,k) |                 F(u,v)
 |      .-----.       |                 .-----.
 |      | +---------------------------->|     |----------------------->.
 |      '-----'       |   Forward       '-----'     Quantisation       :
 |                    |   transform                                    |
 |____________________|                                                |
                                                                       |
 Received Image                                          Transmission  :
  ____________________                                                 |
 |                    |                                                |
 |   sub-image f̂(j,k) |                 F̂(u,v)                         |
 |      .-----.       |                 .-----.                        v
 |      | <-----------------------------|     |<-----------------------'
 |      '-----'       |   Inverse       '-----'     Inverse
 |                    |   transform                 Quantisation
 |____________________|
```

FIGURE B-1 TRANSFORM CODING AND DECODING

The principle characteristics of image coding by transform methods are:

By using an orthogonal transform such as the Discrete Cosine Transform high energy compaction is achieved;

Adaptivity due to sorting the transform sub-images of an image into classes by the level of image activity present;

Averaging of channel noise over the whole sub-image.

The Discrete Cosine Transform (DCT) is a coding method belonging to the general class of discrete orthogonal transforms.

The typical performance obtained with DCT is:

1. 0.5 - 1 bit per pixel for monochrome images
2. 1 - 2 bits per pixel for colour images

B.2 THE DISCRETE COSINE TRANSFORM

B.2.1 General

The two-dimensional Cosine Transform of a discrete function

$f(j,k)$ $j, k = 0, 1, \ldots, N-1$ is defined as:

$$F(u,v) = \frac{4C(u).C(v)}{N^2} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} f(j,k).\cos\left[\frac{(2j+1)\pi.u}{2N}\right].\cos\left[\frac{(2k+1)\pi.v}{2N}\right]$$

where $u, v = 0, 1, \ldots, N-1$

The inverse transform is:

$$f(j,k) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u).C(v)\ F(u,v).\cos\left[\frac{(2j+1)\pi.u}{2N}\right].\cos\left[\frac{(2k+1)\pi.v}{2N}\right]$$

where $C(0) = \frac{1}{\sqrt{2}}$

$C(u) = C(v) = 1$ for $u, v = 1, 2, \ldots, N-1$
$C(u) = C(v) = 0$ elsewhere

Block diagrams of a DCT adaptive coding system are shown in Figures B-2 and B-3.

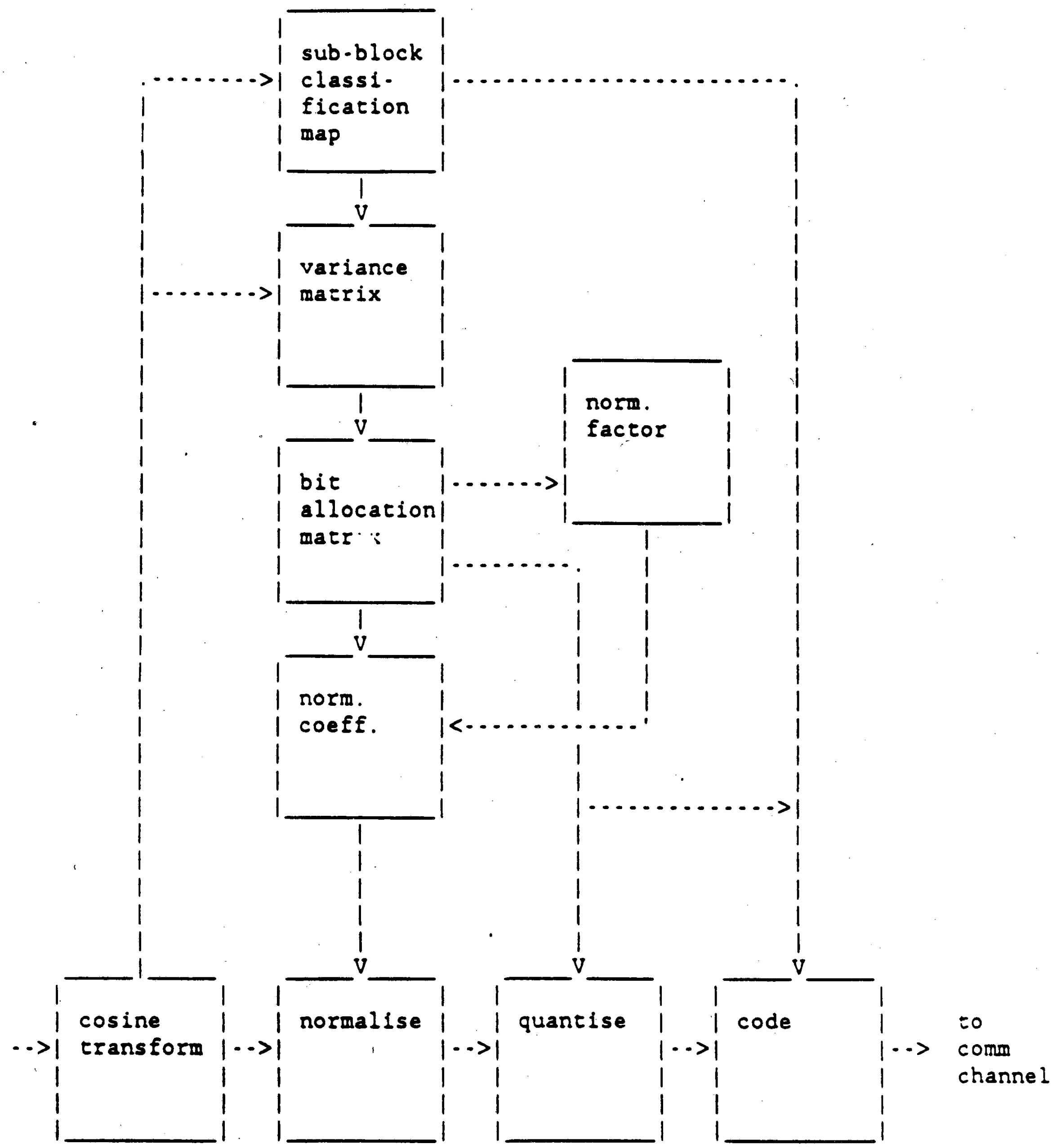


FIGURE B-2 COSINE TRANSFORM ADAPTIVE CODING SYSTEM

```
                     ___________
                    |           |
                    | sub-block |
                .-->| classi-   |---.
                |   | fication  |   |
                |   | map       |   |
                |   |___________|   |
                |         |         |
                |    _____V_____    |
                |   |           |   |
                |   | bit       |   |
                |-->| allocation|---------------.
                |   | matrix    |   |           |
                |   |           |-->|           |
                |   |___________|   |           |
                |         |         |           |
                |    _____V_____    |      _____V_____
                |   |           |   |     |           |
                |   | norm.     |   |     | norm.     |
                |-->| factor    |-------->| coeff.    |
                |   | matrix    |   |     |           |
                |   |           |   |     |           |
                |   |___________|   |     |___________|
                |           |       |           |
                |           |       |           |
                |           |       |           |
                |           |       |           |
       _________|____  _____V_______V_    ______V_____    ___________
      |              ||               |  |           |   |           |
from  | decode       || inverse       |  | inverse   |   | inverse   |
comm -->|            |-->| quantise   |-->| normalise |-->| cosine  |-->
channel |            ||               |  |           |   | transform |
      |              ||               |  |           |   |           |
      |______________||_______________|  |___________|   |___________|
```

FIGURE B-3 COSINE TRANSFORM ADAPTIVE CODING SYSTEM - DECODE

B.2.2 Transform Sub-block Classification

The transform sub-block classification attempts to sort the transform blocks of an image according to criteria which may be functions of image activity, directionality, fineness etc present within each transform sub-image.

After the calculation is performed, the whole range of values of the chosen criterium is divided amongst the number of classes which serve the principle of adaptivity. Within limits, the greater the number of classes the better the adaptivity.

Finally a classification map is generated within which each sub-image is identified by its class identifier. This identifier acts as an index to the bit allocation table associated with that class.

B.2.3 Bit Allocation Table

This step allocates a number of coding bits to individual elements according to their class reference and to a fixed data rate for an average distortion at or below an acceptable level (rate-distortion theory). Bits are then distributed between "busy" and "quiet" image areas to provide the desired adaptivity - more bits being assigned to areas of high image activity and less to those of low activity.

The bit allocation strategy is at the designer's convenience. The following scheme provides for a maxium of 16 different classes.

B.2.4 Normalisation of Coefficients

This calculation is performed to

- avoid clipping of the transform samples prior to quantisation
- use normalised quantisers associated with normalised probability laws.

In order to use normalised probability densities

eg $\rho(x) = \frac{1}{\sqrt{2\pi}} \cdot e^{-\frac{x^2}{2}}$

for the definition of quantisers it is appropriate to specify normalised values for the transform coefficients on which the quantification process is applied.

The normalisation of the transform coefficients is performed through the following scheme:

for u, v = 0 the normalised value to be quantised is:

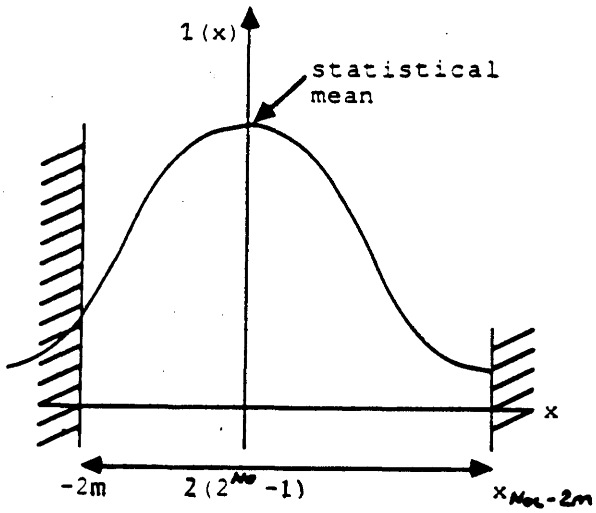
$$x = \frac{F_{m,l}(0,0) - 2m}{2(2^{N_4}-1)/X_{128}}$$

where M = E f(j,k)

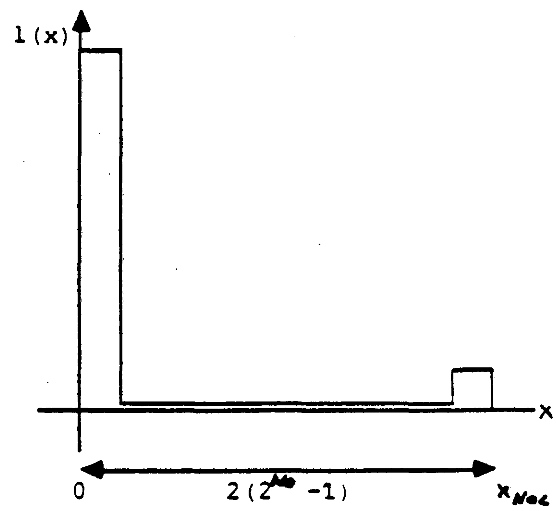
N number of bits per pixel in the original image f(j,k)

X the highest 8 bit decision level

This guarantees that no dc values are clipped.



Typical Case



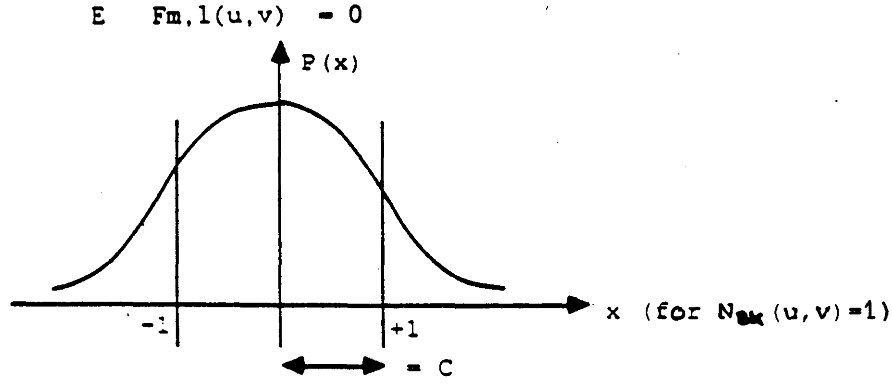
Worst Case
(one sub-block white, the other black)

for u, v $\neq$ 0

$$x = \frac{Fml\,(u.v)}{C \cdot 2^{NBx(u.v)-1}}$$

where C = $\sup_{u,v,k} C_x\,(u,v)$ for Nbk (u,v) = 1

E Fm,l(u,v) = 0



At the receiving end after inverse quantisation the value of the nomalisation factor for Fm,l(o,o) and Fm,l(u,v) is required in order to obtain the correct value of the coefficient before processing the inverse cosine transform.

- For Fm,l(o,o) the value of Nd is required (default value is 8 bits)

- For Fm,l(u,v) the value of C is required (no default value)

B.2.5 Quantisation

The normalised samples are optimally quantised with the number of quantisation levels (bits) set according to the bit allocation tables. The quantisation process attempts to define a relationship between a transform coefficient Fm,l(u,v) and a binary number from Nbk (u,v) bits from the appropriate bit allocation table. This relationship is derived from the distribution law of the coefficients and some function of the error between the input and the output of the quantiser.

The Gaussian distribution law used:

$$p(x) = \frac{1}{\sqrt{2\pi}\,\sigma(u.v)} e^{-\frac{x^2}{2}}$$

with x in the general form

$$x = \frac{Fm,l(u,v) - 2m}{\sigma_K(u,v)}$$

The criterion used as a function of the error between the input and the output of the quantiser is the mean squared error:

$$D = \sum_{i=1}^{2^{Nbk}} \int_{s_{in}(i)}^{s_{in}(i+1)} (s_{in} - s_{out})^2 . p(x) . dx$$

which gives

if x_k are the end points of the $2^{Nbk(u,v)}$ input ranges
y_k are the output levels of the corresponding input ranges

$$x_i = (y_i + y_{i-1} - 1)/2 \qquad \text{for } i = 2, \ldots 2^{Nbk(u,v)}$$

and

$$\int_{x_i}^{x_{i+1}} (x - y_i) . o(x) . dx = 0 \quad \text{for } i = 1, \ldots 2^{Nbk(u,v)}$$

TABLE B-1 EXAMPLE FOR Nbk = 1, 2, 3 BITS

	N = 2		N = 4		N = 8	
i	x_i	y_i	x_i	y_i	x_i	y_i
1	0.0	0.7980	0.0	0.4528	0.0	0.2451
2			0.9816	1.510	0.5006	0.7560
3					1.050	1.344
4					1.748	2.152
Error	0.3634		0.1175		0.03454	

B.3 APPLICATION OF THE DISCRETE COSINE TRANSFORM

DCT image coding may be performed in various ways, this application uses 1 to 2 bits per pixel to code colour images.

This application is coded for with

<ICT> = 2/4 3/2 3/1 3/0

(coding ident, transform, cosine, two dimensional)

B.3.1 Image Structure

The whole image (two fields) is used.

```
ORG --->|_____________________________
        |      :      |      :      |      \
        |      :      |      :   <-------------- Sub-image of 8 x 8 pixels Y
        |      :      |      :      |       \
        |------:------|------:------|--      )- Row of Sub-blocks
        |      :      |      :      |       /
        |      :      |      :      |      /
        |______:______|______:______|__  _/
        |      :      |             |
        |             |
        |             |
        |--    <---------------------------- Sub-block of 16 x 16 pixels Y
        |             |
        |             |
        |______:______|_
        |             |    <---------------- Photo Area
        |
```

FIGURE B-4 IMAGE STRUCTURE

B.3.1.1 Spatial Structure

The sampled image has the following structure (Figure B-5).

```
 *           *           *
 o           o           o
_x_____x_____x_____x_____x_____x___

..x.....x.....x.....x.....x.....x...

 *           *           *
 o           o           o
_x_____x_____x_____x_____x_____x___

..x.....x.....x.....x.....x.....x...
```

Field Orthogonal and Line Orthogonal
Coincident Samples (CCIR mode 2.1.1)

```
x   1st component   Y
o   2nd component   U          _____   Field 1
*   3rd component   V          .....   Field 2
```

FIGURE B-5   SAMPLING STRUCTURE

B.3.1.2 Temporal Structure

The transmitted transform image has the following structure (Figure B-6).

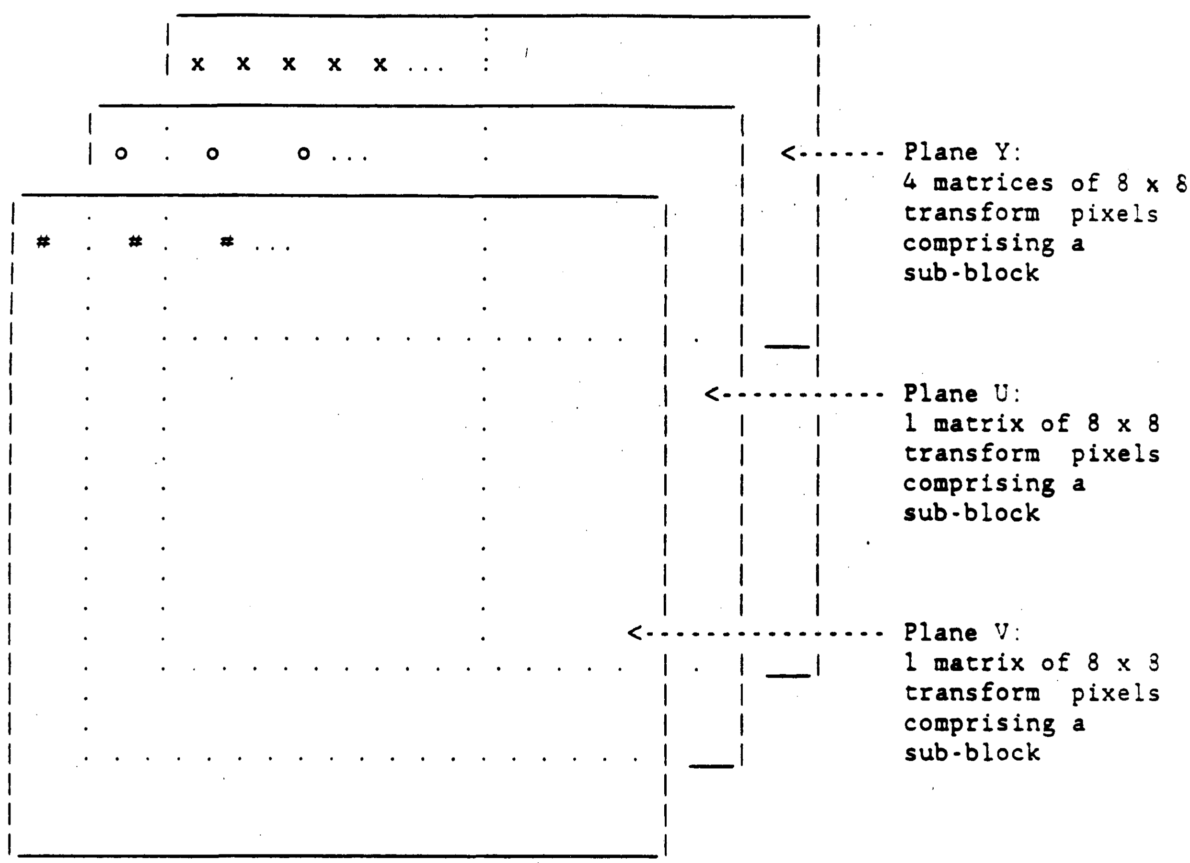


FIGURE B-6 TRANSMITTED TRANSFORM IMAGE STRUCTURE

In mode 2.1.1 one sub-block U (8 x 8) and one sub-block V (8 x 8) are transmitted for each sub-block Y (16 x 16).

B.3.2 Coding Parameters

These parameters will be implicity specified via <SCM> in the sub-sub-type byte with the coded value 3/0.

The parametes are:

sub-block of 16 x 16 pixels (based on first component)

sub-image of 8 x 8 pixels (based on first component)

1 to 16 activity classes

quantisation law using Gaussian distribution of densities

source image quantised with 8 bits per component.

B.3.3 Table Types and Structures

The image is defined using three tables.

B.3.3.1 Table 1

This defines the following:

mY = Ey (f(j,k))	: mean of the Y component over the whole image on one unsigned byte
CY	: normalisation coefficient* of Ym for (u,v) = 0 coded as M x 2 with the mantissa M on two bytes and the exponent n on one signed byte
mU = Eu (f(j,k))	: mean of the U component over the whole image on one unsigned byte
CU	: normalisation coefficient* of Y for (u,v) = 0 coded as M x 2 with the mantissa M on two bytes and the exponent n on one signed byte
mV = Ev (f(j,k))	: mean of the V component over the whole image on one unsigned byte
CV	: normalisation coefficient* of V for (u,v) = 0 coded as M x 2 with the mantissa M on two bytes and the exponent n on one signed byte

The normalisation coefficient for Fm,l(o,o) is not transmitted as the most likely value of Nd is 8 bits and thereforee this is the default value.

B.3.3.1.1 Data Structure

The data will be sent in the following sequence (mY first):

mY CY mU CU mV CV

(bytes are sent most significant bit first)

B.3.3.2 Table 2

This defines the following:

N luminance bits allocation tables
N U component bits allocation tables
N V component bits allocation tables

B.3.3.2.1 Data Structure

Four bits are provided for allocation of fifteen bits per coefficient.

$$\begin{array}{|llcl|}\hline Y_{11} & : Y_{12} & & : Y_{18} \\ \ldots & \ldots & & \ldots \\ \vdots & & & \vdots \\ & & & \\ \ldots & \ldots & & \ldots \\ Y_{81} & : Y_{82} & & : Y_{88} \\ \hline \end{array} \qquad \begin{array}{|llcl|}\hline U_{11} & : U_{12} & & : U_{18} \\ \ldots & \ldots & & \ldots \\ \vdots & & & \vdots \\ & & & \\ \ldots & \ldots & & \ldots \\ U_{81} & : U_{82} & & : U_{88} \\ \hline \end{array} \qquad \begin{array}{|llcl|}\hline V_{11} & : V_{12} & & : V_{18} \\ \ldots & \ldots & & \ldots \\ \vdots & & & \vdots \\ & & & \\ \ldots & \ldots & & \ldots \\ V_{81} & : V_{82} & & : V_{88} \\ \hline \end{array}$$

FIGURE B-7 BIT ALLOCATION TABLES

For each table data is transmitted a row by row, comencing with byte (most significant bit first). Tables are transmitted in order Y, U and then V.

If required the data related to each component may be transmitted in separate Transparent data VPDEs. (See Part 7.)

B.3.3.3 Table 3

This is a table of variable length which describes the activity over the whole image. Each sub-image is given a 4-bit reference to one of the classes.

B.3.3.3.1 Data Structure

Activity of Y
within the first
sub-image of 8 x 8 ---->
(Ay_{ij} - 1,2 ... 16)

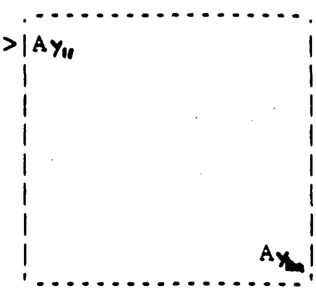


FIGURE B-8 ACTIVITY TABLE FOR Y

Similar tables are transmitted for U and V with appropriate numbers of coefficients. (Four times fewer U and V coefficients than Y when in 2.1.1 mode.)

For each table data is transmitted a row by row, comencing with byte (most significant bit first). Tables are transmitted in order Y, U and then V.

If required the data related to each component may be transmitted in separate Transparent data VPDEs. (See Part 7.)

B.3.4 Photographic Pixel Transfer Unit

The header is not described here as it depends on the particular image being transmitted. Photographic pixel data is transmitted in Transparent data mode (See Part 7).

In order to avaoid propagation of errors through consecutive sub-blocks, each sub-block description is resynchronised by a new Transparent VPDE with an appropriate length indicator.

Each sub-block is coded with transformed and quantised coefficients sent in sequence as follows:

```
US  3/4   <ORG>  <ARE>

US  3/15  length  Y1   Y2   Y3   Y4   U   V

US  3/15  length  Y'1  Y'2  Y'3  Y'4  U'  V'

US  3/15  length  Y"1  Y"2  . . .

:
:
```

B.4 EXAMPLE OF CODING FOR DCT

The VPDEs will take the form:

PIXEL HEADER UNIT

VPCE	US	3/4	2/0			
SDC						Colour YUV (default)
Display composition	2/1	3/4	3/1	3/6	3/11	416Hy x
		3/6	3/2	3/5	3/11	208Vy
		3/2	3/0	3/8	3/11	208Huv x
		3/3	3/1	3/2	3/11	312Vuv
	2/3	3/0	3/1			fields & lines alternate samples
SMC coding method	2/4	3/2	3/0	3/0		discrete cosine transform - two dimensional

TABLE TRANSFER UNIT

	Table 1 Header	
VPCE	US 3/5 2/0	
Table Set	2/1 3/1 3/11 3/3 3/11	Table 1; 3 sub-tables
Table Size	2/2 3/8 3/11 3/9 3/11	Z = 8 bits, Y = 9
	Table 1 Transfer	
VPCE	US 3/5	
Address	2/1 3/1 3/11	Table 1; address 0
Data	US 3/15 0/11 transparent data ...	
	Table 2 Header	
VPCE	US 3/5 2/0	
Table Set	2/1 3/2 3/11 3/3 3/11	Table 2; 3 sub-tables
Table Size	2/2 3/4 3/11 3/1 2/2 3/8 3/11	Z = 4 bits, Y = 128
	Table 2 Transfer	
VPCE	US 3/5	
Address	2/1 3/2 3/11	Table 2; address 0
Data	US 3/15 5/3 transparent data ...	
	Table 3 Header	
VPCE	US 3/5	
Table Set	2/1 3/3 3/11	Table 3;
Table Size	2/2 3/4 3/11	Z = 4 bits, Y depending on image size
	Table 3 Transfer	
VPCE	US 3/5	
Address	2/1 3/3 3/11	Table 3; address 0
Data	US 3/15 length transparent data ...	

PIXEL TRANSFER UNIT

VPCE	US 3/4	
<ORG>		Top Left DDA (default)
<ARE>	2/13 3/1 3/6 3/0 3/11 3/3 3/2 3/0 3/11	X = 160 pixels Y = 320 pixels
Data	US 3/15 length transparent data ...	

CONTENTS

1.0 INTRODUCTION

A DRCS is a set of characters whose shapes are sent from the service and down-loaded via the line. It may be used to represent alphabetic characters, special symbols, or picture element symbols for constructing fine graphics. Once loaded, the DRCS are regarded as members of a library that can be designated by appropriate ESCAPE sequences as G0, G1, G2 or G3 sets.

Two types of DRCS have been identified. The first type is the basic DRCS. Only the shapes of the characters are down-loaded. Characters are displayed on the screen in the prevailing foreground colour on the prevailing background colour. In the second type of DRCS the down-loaded characters are completely defined in foreground colours, ie all the dots of a character cell have a defined foreground colour, chosen from a number of colours.

The protocol defined below for down-loading of DRCS allows down-loading of both types of DRCS. The protocol is open ended to allow for future extensions.

The down-loading of DRCS is accomplished using units of two types:

DRCS header units

DRCS pattern transfer units.

A DRCS header unit describes the general properties of the DRCS to be loaded. The actual pattern transfer takes place using DRCS pattern transfer units. Both units are coded as Videotex Presentation Data Elements (VPDE) in acccordance with the Presentation Level Data Syntax (PLDS) as:

US 2/3 Y <data>

Y = 2/0 indicates DRCS header units
Y ≠ 2/0 indicates DRCS pattern transfer units.

2.0 DRCS HEADER UNITS

A DRCS header unit applies for all following pattern transfer units until the header is redefined or until the end of a session. A DRCS header unit is coded as:

US 2/3 2/0 <ICS> <SDC> <SCM> <SSA>

The various fields of the DRCS header unit are coded with bytes from different columns of the code table. If the default conditions apply these fields are omitted. The default conditions are specified in the description of the fields. They are independent of previously loaded header units.

2.1 <ICS>: Identification of Character Set

The <ICS> field identifies the DRCS to be loaded, by a number which will consequently be used in the designation sequence for this set. With the exception of the last byte, all the bytes of the <ICS> field are taken from column 2 of the code table.

<ICS> : 2/k F

k: Indicates whether the set belongs to the first or second repertory in the library (see Section 5 for the designation sequence). It also indicates whether a possibly existing DRCS identified by the same <ICS> field should be deleted or merely be overwritten by the following pattern transfer units.
- 0: first repertory, do not delete existing DRCS
- 1: second repertory, do not delete existing DRCS
- 8: first repertory, delete existing DRCS
- 9: second repertory, delete existing DRCS

F: If the DRCS is registered in ISO 2375 then F is the character allocated by ISO.
If the DRCS is a non-registered set then the sequence 2/0 Fx is transmitted. Fx can be taken from columns 4 to 7 of the code table.

The default for <ICS> is: 2/0 2/0 4/0, which will identify a non-registered set of the first repertory to be loaded in the library with final character for designation 4/0. An existing DRCS in this library position will not be deleted.

2.2 <SDC>: Select Dot Composition

The <SDC> field describes the structure of the cells of the DRCS to be loaded. The <SDC> field also discriminates between the two types of DRCS.

There are two alternative types of <SDC>.

2.2.1 SDC Type 1

The first type is coded with bytes from column 3 of the code table. This is the extended type of <SDC>. Its coding will allow for future extension of the DRCS architecture.

<SDC> type 1 : <character cell> <blocking factor> <pixel characteristics>

<character cell> : 3/th 3/uh 3/11 3/tv 3/uv 3/11

th : (0,1...9) : tens of horizontal pixels, leading zeros may be omitted *
uh : (0,1...9) : units of horizontal pixels, no default for the number of horizontal pixels
tv : (0,1...9) : tens of vertical pixels, leading zeros may be omitted *
uv : (0,1...9) : units of vertical pixels, default number of vertical pixels = 10

<blocking factor> : indicates the grouping of character cells (horizontal x vertical) for a rectangular character block. This character block is considered as a single character cell during the character description. When down-loaded in the terminal a character block occupies h x v consecutive character positions in the DRCS. The coding of the <blocking factor> is: 3/th 3/uh 3/11

3/tv 3/uv 3/11

th : (0,1...9) : tens of horizontal character cells, leading zeros may be omitted *
uh : (0,1...9) : units of horizontal character cells
tv : (0,1...9) : tens of vertical character cells, leading zeros may be omitted *
uv : (0,1...9) : units of vertical character cells

The default coding for the <blocking factor> is 3/1 3/11 3/1 3/11. indicating a character block of 1x1 character cell.

<pixel characteristics> : for further study. The default condition is 1 bit/dot/pixel basic DRCS.

* The coding scheme allows more significant digits to be added if needed.

2.2.2 SDC Type 2

The second type of <SDC> is coded with bytes from columns 4 and 3 of the code table. This is the shorthand type of <SDC>.

<SDC> type 2 : 4/p <blocking factor> 4/q

p : indicates envisaged recommended dot matrix sizes (horizontal x vertical). There is no default for p.

0:	16x24	8:	8x12
1:	16x20	9:	8x10
2:	16x12	10:	6x12
3:	16x10	11:	6x10
4:	12x24	12:	6x 5
5:	12x20	13:	4x10
6:	12x12	14:	4x 5
7:	12x10	15:	6x 6

<blocking factor> : as above

q : indicates the number of bits per dot used to code the DRCS. The default for q is 1, indicating 1 bit per dot basic DRCS. Colour DRCS is coded with q $\neq$ 1 :

1: 1 bit/dot basic DRCS
2: 2 bit/dot colour DRCS, 4 colours
default DCLUT: black, red, green, yellow
3: 3 bit/dot colour DRCS, 8 colours
default DCLUT: 1st colour palette
4: 4 bit/dot colour DRCS, 16 colours
default DCLUT: 1st and 2nd colour palettes (or 3rd and 4th colour palettes if the 1st and 2nd colour palettes are not redefinable)

The colours as mentioned above can be modified by loading the DRCS Colour Look Up Table (DCLUT). See Part 5, redefinable colours.

Since there is no default for some bytes in the <SDC> a DRCS header unit must always contain at least part of an <SDC>. If <SDC type 1> is used, at least 3/uh and 3/11 must be contained in the header; if <SDC type 2> is used, 4/p must be included in the header.

2.3 <SCM>: Select Coding Method

The <SCM> field determines the way in which the DRCS patterns are coded as they are down-loaded. Details of the coding are also determined by the <SDC> field (eg matrix size, bits per dot, blocking factor, pixel characteristics). The bytes of the <SCM> field are taken from column 5 of the code table, see Page 11.

<SCM> : <type> [<sub-type>]

<type> : 5/t (default t=0)

<sub-type> : 5/st

The <type> field identifies the coding method of the DRCS to be down-loaded. Some coding methods require a <sub-type> field to identify options within the coding method.

t = 0: 'direct' coding, described in 4.1
no <sub-type> needed
t = 1: 'Runlength' coding, described in 4.2
the default coding for st = 0

Other coding methods are for further study.

2.4 <SSA>: Select Set Attributes

The <SSA> field describes the actions which certain attributes will have on the DRCS characters once they are displayed on the screen. The bytes of the <SSA> field are taken from column 6 of the code table see Page 11.

The details of the <SSA> are for further study.

The default for <SSA> is such that the LINED attribute causes an underline, as for alphanumeric characters, but has no effect on colour DRCS.

3.0 PATTERN TRANSFER UNITS

Pattern transfer units are coded as:

US 2/3 Y <pattern data>

Y: the code of the first character (or character block) described in the unit; it has a value in the range 2/1 to 7/14 inclusive.

The <pattern data> field of a pattern transfer unit describes the patterns for the characters of the down-loaded DRCS, in accordance with the last received DRCS header unit.

The value of the Y parameter defines the code of the first defined character. If the pattern transfer unit contains more character definitions, they will be assigned subsequent codes. Data contained in a pattern transfer unit for a character subsequent to a character with code 7/14 will be discarded.

The coding methods to be used in the <pattern data> are described in Section 4.

4.0 CODING METHODS

In the following sub-sections the recommended coding methods, as indicated in the <SCM> field of the DRCS header units, are defined.

4.1 'Direct' Coding Method

The 'direct' coding method is identified by t=0 (default value) in the <SCM> field of the DRCS header unit. No <sub-type> is needed for this coding method. The method can be used to load basic DRCS as well as colour DRCS.

4.1.1 Basic DRCS

A DRCS character cell consists of m dots horizontally and n dots vertically (in total m x n dots). The values of m and n are determined by the <SDC> field of the DRCS header unit. The direct coding method can be used for all possible values of m and n.

The dots of a character are coded using bytes from columns 4 to 7 of the code table, these bytes are called D bytes. The dots are loaded six dots at a time, row by row, starting from the top left hand corner, using the six least significant bits. Dots defined as '1' are displayed in foreground colour.

To improve the efficiency of this code a number of special commands have been added. They are coded as bytes from column 2 of the code table (see page 11) and are called S-bytes. The coding of these bytes is:

code	name	description
2/0	Sf	fill rest of character with '0's
2/1	R1	repeat last complete row once
2/2	R2	repeat last complete row twice
2/3	R3	repeat last complete row 3 times
2/4	R4	repeat last complete row 4 times
2/5	R5	repeat last complete row 5 times
2/6	R6	repeat last complete row 6 times
2/7	R7	repeat last complete row 7 times
2/8	R8	repeat last complete row 8 times
2/9	R9	repeat last complete row 9 times
2/10	R10	repeat last complete row 10 times
2/12	S0	defines a complete row containing '0's
2/13	S1	defines a complete row containing '1's
2/14	Sr	fill rest of character with last complete row
2/15	Ss	fill rest of character with '1's
3/0	B1	start of pattern block for new character

The pattern block for each DRCS character is preceded by the command B1 (3/0).

The insertion of an S-byte may leave a number of remaining bits in the previous D-byte, which will not define a complete row. The use of these bits is explained below.

The actions of the Sf (2/0) command are as follows. The remaining bits of the last D-byte are used as the first bits of the next row; the rest of this row and the possibly remaining rows of the character are filled with '0's. The action of the Ss (2/15) command is equivalent, but with the character filled with '1's.

The Sr (2/14) command causes the last complete row to be copied in the remaining rows of the character. Remaining bits in the last D-byte are discarded.

For the remaining commands S0 (2/12), S1 (2/13) and R1 (2/1) to R10 (2/10) the processing of the remaining bits in the last D-byte is postponed until the action indicated by the command is executed. Together with the next D-byte (or Sf or Ss) these bits are used for the definition of the remaining part of the character. If the rest of the character is completely defined by the commands mentioned in this paragraph the remaining bits are discarded.

The extent of the repeat command cannot cross the border of a character block. If a repeat command is used as the first byte of a character definition (ie the first byte after a B1 command) the action is as if the last complete row consisted of all '0's.

If a B1 command is received before a character is completely defined, the remaining part is defined as all '0's. Excess bytes before a B1 command are ignored.

4.1.2 Colour DRCS

In the pattern transfer units for colour DRCS, a number of bits per dot are down-loaded to identify the colour of each dot. In the 'direct' coding method the pattern information for the DRCS is transmitted as one or more pattern blocks for each DRCS character. A pattern block defines one bit of each of the dots of the DRCS character as shown in Figure 1 below. The pattern blocks are separated by separation bytes (B-bytes) coded from column 3 of the code table (see Table 1 page 11).

code	name	description
3/0	B1	start of the 1st pattern block of a DRCS character, defining the least significant bit of the dot
3/1	B2	start of the 2nd pattern block
3/2	B3	start of the 3rd pattern block
3/3	B4	start of the 4th pattern block

Equal pattern blocks only have to be transmitted once. In that case the pattern block is preceded by the two (or more) separation bytes to which the pattern block applies.

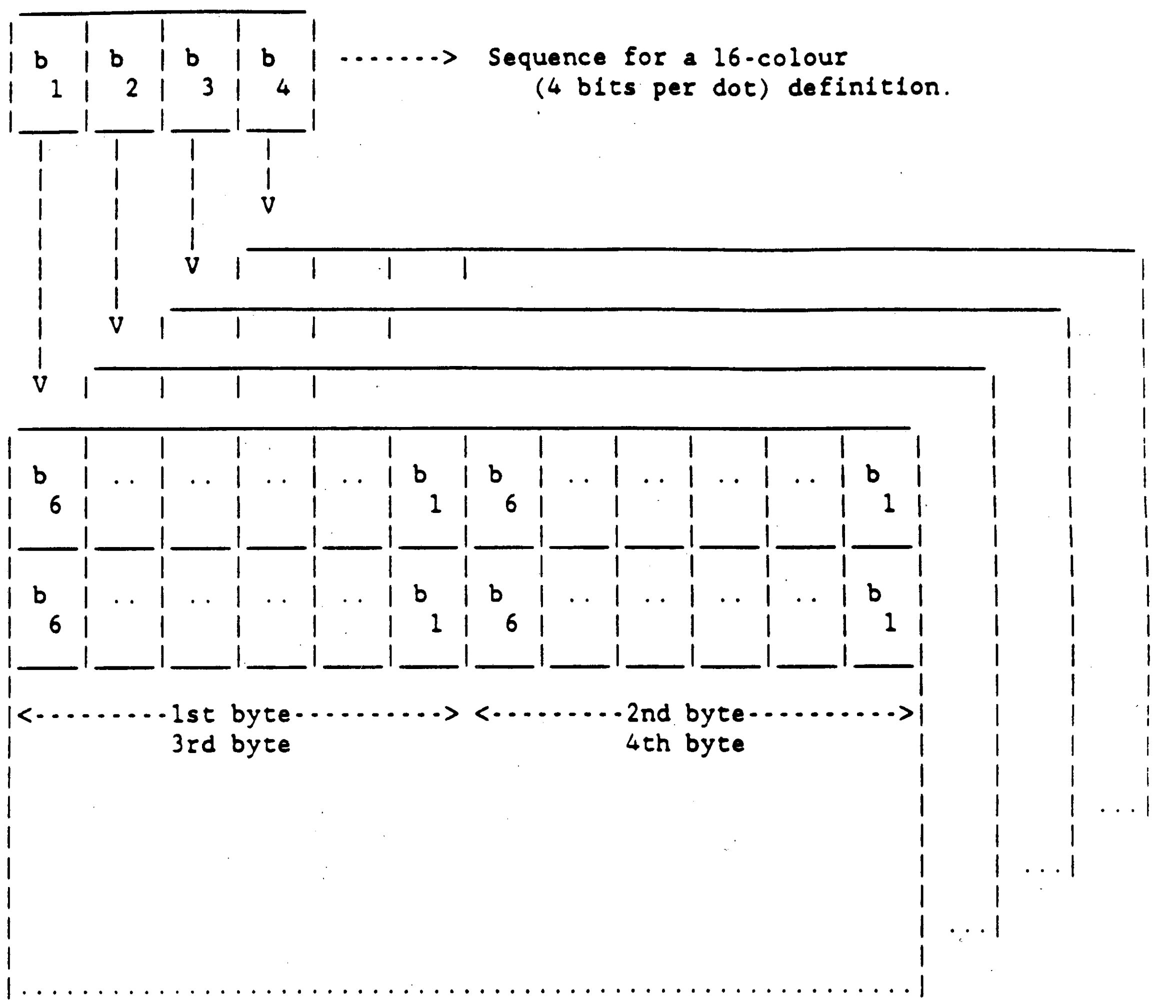


FIGURE 1   DRCS DOWNLOADING FORMAT DEFINITION FOR A 12 x 10 DOT MATRIX

Examples of pattern transfer units for colour DRCS are given below:

Sixteen-colour DRCS (4 bits per dot):

US 2/3 Y 3/0 <1st pattern block> 3/1 <2nd pattern block>
3/2 <3rd pattern block> 3/3 <4th pattern block> 3/0 ...

US 2/3 Y 3/0 3/1 <pattern block> 3/3 2/0 3/0 ...

In the second example the pattern blocks for the first three bits of the dots are equal, while the fourth bit is '0'.

Four-colour DRCS (2 bits per dot):

US 2/3 Y 3/0 <1st pattern block> 3/1 <2nd pattern block> 3/0 ..

4.1.3 Direct Coding Code Table

TABLE 1 DIRECT CODING CODE TABLE

	0	1	2	3	4	5	6	7
0			Sf	B1	0	16	32	48
1			R1	B2	1	17	33	49
2			R2	B3	2	18	34	50
3			R3	B4	3	19	35	51
4			R4		4	20	36	52
5			R5		5	21	37	53
6			R6		6	22	38	54
7			R7		7	23	39	55
8			R8		8	24	40	56
9			R9		9	25	41	57
10			R10		10	26	42	58
11					11	27	43	59
12			S0		12	28	44	60
13			S1		13	29	45	61
14			Sr		14	30	46	62
15			Ss		15	31	47	63
			S-	B-	<-------- D-bytes			-------->

4.2 'Runlength' Coding Method

The runlength coding method is identified by t=1 in the <SCM> field of the DRCS header unit. In some cases a <sub-type> field is needed for this coding method. The default value for the <sub-type> is 5/0 (st=0).

The runlength coding method can be used for basic DRCS as well as colour DRCS, although the method may be best used for advanced types of colour DRCS using a non-default <blocking factor>.

The following general rules apply for runlength coding.

Runlength coding uses the six least significant bits from bytes of columns 4 to 7 of the code table. These bits are identified as b6, b5, b4, b3, b2 and b1 (b1 is the least significant).

Runlength coding is applied on character blocks as defined by the <SDC> field of the header unit (default 1 x 1), row by row, starting from the top left hand corner of the block.

If a runlength exceeds the right hand border of the character block, the remaining part of the runlength is continued on the next row. If it exceeds the right hand border of the last row of the character block the remaining part is ignored.

4.2.1 Basic DRCS

Two types of runlength coding for basic DRCS are specified. The first type is identified by st=0 (default). In this case the runlength is coded with three bits:

b6,b5,b4: runlength for the background colour
b3,b2,b1: runlength for the foreground colour

The coding for each runlength is:

code	length
001	0
010	1
011	2
100	3
101	4
110	5
111	6
000	escape

If the escape code is used, the six bits of the following byte are completely used to code the runlength (1 to 63). If both runlengths in a byte are coded as escape, the second byte will contain the runlength of the background colour and the third byte the runlength of the foreground colour.

The second type of runlength coding for basic DRCS is defined by st=1. In this case the coding is:

b6: 0 runlength for background colour
    1 runlength for foreground colour

b5,b4,b3,b2,b1: runlength (1 to 31)

4.2.2 Colour DRCS

For colour DRCS the runlength is coded per colour.

In the case of sixteen-colour DRCS the runlength coding will be:

b6,b5,b4,b3: colour definition

b2,b1: runlength
01 length 1
10 length 2
11 length 3
00 escape

If the escape code is used, the six bits of the next byte define the runlength (1 - 63).

For eight-colour DRCS the runlength coding will be:

b6,b5,b4: colour definition

b3,b2,b1: runlength (1-7)
000 escape

For four-colour DRCS the runlength coding will be:

b6,b5: colour definition

b4,b3,b2,b1: runlength (1-15)
0000 escape

5.0 DESIGNATION AND INVOCATION OF DRCS

Once a DRCS (or part of it) is down-loaded, the set is considered part of the library. The set can then be designated by the ESC-sequence.

ESC I F

I= 2/k+i

k: 8 or 12 indicating the first or second repertory. The value for k should be in accordance with the value for k in the <ICS> field of the header unit of the required DRCS.

i: 0,1,2 or 3 depending on whether the set is designated as a G0,G1, G2 or G3 set respectively.

F : If the DRCS is registered in ISO 2375 then F is the character allocated by ISO.
If the DRCS is a non-registered set then the sequence 2/0 Fx is transmitted, where Fx is equal to Fx in the <ICS> field of the header unit of the required DRCS.

Once the set is designated, it can be invoked in the normal manner.

If a character block is to be displayed on the screen, the top left hand character cell will be positioned at the active position. After the display of the block the active position will be in the next position following the top right hand character cell of the block.

6.0 APPLICABILITY OF ATTRIBUTES

Unless the <SSA> field of the DRCS header unit defines otherwise all attributes shall apply in the normal way to DRCS characters, the only exception being that the LINED attribute is not applicable to colour DRCS.

Although in colour DRCS a character is completely defined in foreground colour, it should be remembered that at the position where a colour DRCS character is displayed there is a defined background colour, which should be applied in the case of, for example, the INVERT or the FLASH attribute.

CONTENTS

1.0 INTRODUCTION

The alphamosaic C1 sets provide for the selection of eight colours. In this part the method used to extend this colour system and to redefine colours will be described.

2.0 COLOUR SYSTEM EXTENSION

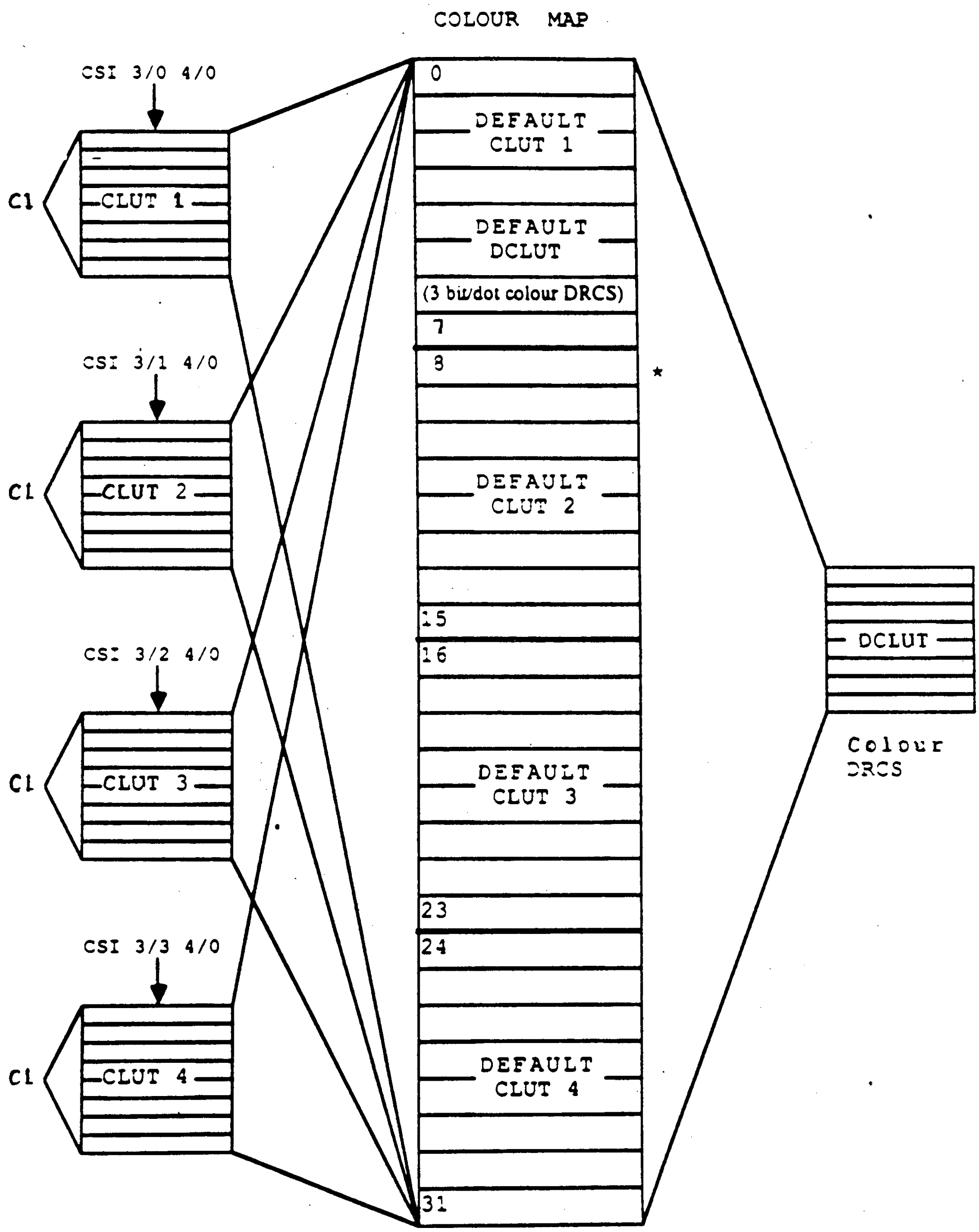
The extension of the colour system is accomplished by providing a number of colour tables of eight colours each. At a given instant only one table can be in use. This table is selected using a CSI sequence (see Part 1 Section 2.3.13). Each table is implemented as a Colour Look Up Table (CLUT) with eight entries. The entry in the 'in use' CLUT is selected using the C1 controls according to the table below.

C1 control colour	entry no. in CLUT
black	0
red	1
green	2
yellow	3
blue	4
magenta	5
cyan	6
white	7

The entry in the CLUT contains an ordinal number in the colour map. The contents of this colour map entry define the colour. In the alphamosaic mode four CLUTs are used. They are named CLUT1, CLUT2, CLUT3 and CLUT4. The size of the colour map is 32 entries, divided into 4 parts of 8 entries each.

For colour DRCS (see Part 4) separate look up tables called DCLUTs may be provided. A DCLUT contains a number of entries which are used to define the colours used in colour DRCS. The colour extension scheme is shown schematically in Figure 1.

FIGURE 1 COLOUR EXTENSION SCHEME



* If this entry (No 8) is defined as BLACK (as it is by default) it will be interpreted as TRANSPARENT

3.0 CODING OF REDEFINABLE COLOURS

The Define COLOUR VPDE is used to redefine the contents of the colour map, or to redefine the contents of the CLUTs or the DCLUTs. The coding is:

US 2/6 Y <data>

Y : determines the function of the Define COLOUR VPDE

2/0 : define COLOUR header unit
2/1 : define COLOUR reset unit
3/x : COLOUR transfer unit

3.1 COLOUR Header Unit

A COLOUR header unit applies for all following colour transfer units until the header is redefined or until the end of a session. The header unit is coded as:

US 2/6 2/0 <ICT> <SUR> <SCM>

<ICT> : Identification of Colour Table, is coded as: 2/a I

a : indicates the type of colour table
0 : colour map
1 : CLUT
2 : DCLUT

I : indicates the number of the unit indicated in 2/a.
I is in the range 2/0 to 7/15.

The default coding for <ICT> is 2/0 2/0, identifying the colour map No 1.

<SUR> : Select Unit Resolution, is coded as: 3/c.

c : (1,2...9) indicating the number of bits used to define each unit of the identified table.

The default value for <SUR> is 3/4.

<SCM> : Select Coding Method, is coded as: 4/d.

d : indicates the coding method
0 : entries in colour map
1 : load colour map using R,G,B

The default value for <SCM> = 4/1.

If necessary an extra byte (coded 4/e) may be added.

3.2 COLOUR Reset Unit

The COLOUR Reset Unit is used to reset all the colour tables (CLUTs, DCLUTs and colour map) to their default values. The reset unit is coded as:

US 2/6 2/1

3.3 COLOUR Transfer Units

COLOUR transfer units are used to load colour tables. The colour table to be loaded and the loading method used are defined by the 'Define COLOUR header unit. The COLOUR transfer units are coded as:

US 2/6 Y <colour data>

Y : will indicate the first table entry to be loaded, and is coded as: 3/t 3/u

t : (0,1...9) tens of address, leading zeros may be omitted
u : (0,1...9) units of address *

<colour data> : bytes in the range of 4/0 to 7/15.

The meaning of the <colour data> depends on the preceding 'Define COLOUR' header unit and is defined in the following sub-sections.

3.3.1 Loading a CLUT or DCLUT

This function is identified by the last received 'Define COLOUR' header unit with <ICT> = 2/1 (CLUT) or 2/2 (DCLUT). The least significant <SUR> bits are taken from each byte of the colour data and stored in consecutive locations of the identified CLUT or DCLUT, starting at the address indicated by Y.

Data received for addresses outside the identified CLUT or DCLUT will be discarded.

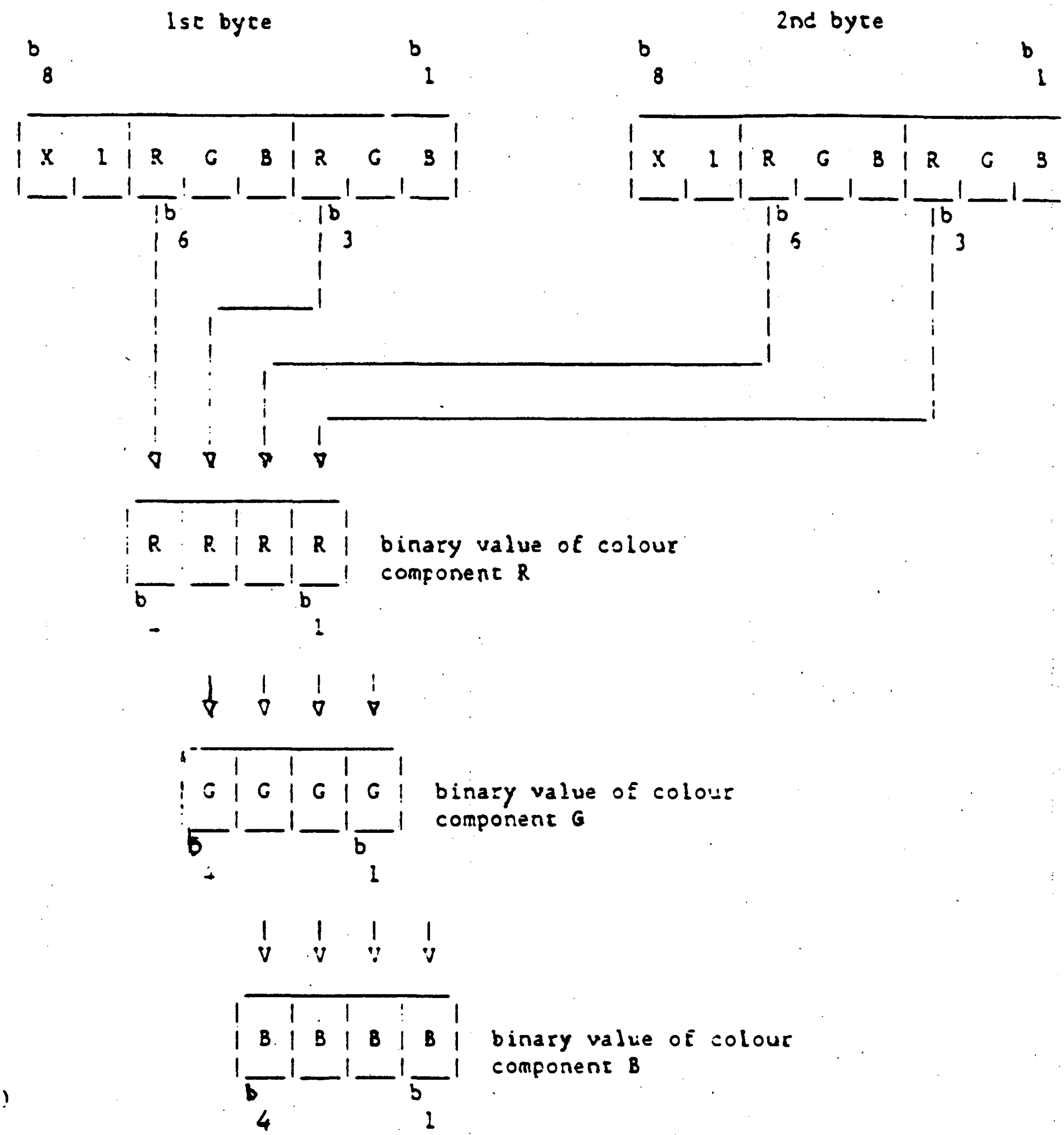
* The coding scheme allows more significant digits to be added if necessary.

3.3.2 Loading the Colour Map using R,G,B

This function will be identified by the last received 'Define COLOUR' header unit with <ICT> = 2/0 and <SCM> = 4/1, or the default header unit. The colour map is loaded starting at the address indicated by Y. The colours are defined in terms of their Red, Green and Blue components, each of which is defined by <SUR> bits.

Each <colour data> byte contains two bits for each of the primary colours, except for the last byte, which may contain only one relevant bit for each primary colour. The coding of the six least significant bits of the <colour data> bytes is: R G B R G B, the most significant bits defining the more significant bits of the colour components. A value of '0' for a colour component indicates zero intensity. All bits '1' indicate full intensity. Intermediate values are interpreted in equal brightness steps (eye corrected).

FIGURE 2 EXAMPLE OF COLOUR TRANSFER IN THE CASE OF DOWN LOADING VALUES OF RGB WITH 4 BITS EACH



CONTENTS

1.0 INTRODUCTION

The 'Define FORMAT' VPDE is used to define the number of rows and columns displayed within the defined display area for the alphamosaic display.

The default alphamosaic display format is 24 rows of 40 characters.

The possibility of changing the aspect ratio of the defined display area using the 'Define FORMAT' VPDE is for further study.

2.0 CODING

The coding of the 'Define FORMAT' VPDE is as follows:

US 2/13 Y <CH> <CT> <CU> 3/11 <RH> <RT> <RU> 3/11 <WC>

If Y is 4/1 to 4/14 one of the following formats is defined:

4/1 : 40 columns by 24 rows
4/2 : 40 columns by 20 rows
4/3 : 80 columns by 24 rows
4/4 : 80 columns by 20 rows
4/5 : 48 columns by 20 rows
4/6 to 4/14 are for further study

If Y is 4/15 the number of columns and rows is defined by the following data, where:

<CH> <CT> <CU> is the number of columns in hundreds, tens and units, coded from column 3 (leading zeros may be omitted); *

<RH> <RT> <RU> is the number of rows in hundreds, tens and units, coded from column 3 (leading zeros may be omitted). *

<WC> is used to define the wraparound controls. <WC> takes the following values:

7/0 : wraparound ON
7/1 : wraparound OFF

3.0 DEFAULTS

The default 'Define FORMAT' VPDE is:

US 2/13 4/1 7/0

* The coding scheme allows more significant digits to be added if necessary.

TRANSPARENT Data

CONTENTS

1.0 INTRODUCTION

Certain Videotex applications such as geometric and photographic displays contain a relatively large amount of data. Consequently it is desirable for increased efficiency to use all the presentation level code bits for actual data (7 or 8 bits per byte). In such a mode all codes pass uninterrupted by the normal presentation level control codes and the mode is thus termed transparent.

2.0 PROTOCOL

The 'TRANSPARENT data' VPCE is used to enter transparent mode. There are two methods of leaving the transparent mode, either following a byte count (where a byte is 8 bits), or when a new VPCE is detected. Immediate exit by a new VPCE ensures 'RESET to service break to row X' will operate.

When a US (01/15) code (which is used to signify the start of a new VPCE) appears naturally in the data it should be transmitted twice (this technique is known as byte stuffing). A new VPCE is detected by a single US in the data stream.

The value of the first byte 'N' in the transparent mode data indicates the normal method of exit from the transparent mode. When 'N' is zero then no byte count is defined and transparent mode is only exited at the start of a new VPCE. If N has a value of between 1 and 254 inclusive, then this value specifies the number of bytes that is to be received before a return is automatically made to the previous VPDE. The transparent mode byte count is performed on received bytes after stuffing bytes have been removed.

Transparent Mode (with US stuffing)

```
                 ______________________________________________
                |   |                      |                 |
                |   |  Count N             |        Count    |
      US 3/15 N |   |  Characters      US  |        Complete | US
                |   |                      |                 |
________________|___|                      V                 |____________________
Normal Mode                                |                 Next VPDE (may be more
                                           |                 transparent data)
   N = 0 exit by new VPCE                  |
   N = 1 to 254 inclusive                  |______________________________________
       exit following a byte                     New VPDE
       count or by new VPCE
   N = 255 is not defined
```

FIGURE 1 TRANSPARENT MODE SWITCHING

CONTENTS

1.0 INTRODUCTION

This function is used to set predefined states in the terminal and thereby synchronise the Videotex service and the terminal at the presentation layer.

2.0 CODING STRUCTURE.

The coding structure for the Reset function is as follows:

US 2/15 <operation> <parameter>

<operation>: This character indicates the display mode being reset, and the operation required. This character is coded from columns 2 to 4 of the code table.

<parameter>: This character is coded from columns 4 to 7 of the code table. Its meaning depends upon the reset operation.

3.0 FUNCTIONAL DESCRIPTION AND CODING.

3.1 General Display Reset

By means of the General Display Reset the terminal can be brought into a defined condition, regardless of the the terminal mode. It's function is to combine the actions of the Reset operation for the Alpha-mosaic display, the Geometric display, the Photographic display etc. It's actions are summarized below:

- The actions as described in section 3.2. (Reset Control and Graphic Sets) are executed.
- The format is set to default to 24 rows of 40 characters.
- The Alpha-mosaic DDA shall be filled with spaces.
- The active position is set to the first character position of the first row.
- All attributes of the Alpha-mosaic display are set to their default values as described in Part 1 para 1.5.2.

- The actions on the geometric display can be described as follows:

  General Display Reset is equivalent to the execution of the following sequence of primitives:

```
DEACTIVATE WORKSTATION (ws_id) for all workstations
CLOSE WORKSTATION (ws_id) for all workstations
OPEN WORKSTATION (ws_id) for ws_id = 0
ACTIVATE WORKSTATION (ws_id) for ws_id = 0
SET DEFAULTS
SET WORKSTATION WINDOW (ws_id, P1, P2)
     ws_id = 0, P1 = (0.0, 0.0), P2 = (1.0, 0.75)
SET WORKSTATION VIEWPORT (ws_id, XMIN, XMAX, YMIN, YMAX)
     ws_id = 0, XMIN = 0.0, XMAX = 1.0, YMIN = 0.0, YMAX = 0.75
```

  Note: OPEN WORKSTATION (ws_id) ensures that the display surface is cleared.

- For the Photographic display all the Photographic tables are cleared and the Photographic display is set to transparent. (This is for further study)

- The action of a Timing Control Wait VPDE is terminated and the data received since the start of the Wait is deleted.

- The data following the General Display Reset function is to be interpreted as Alpha-mosaic data.

The coding of the General Display Reset is as follows:-

a) The serial C1 set is invoked

   US 2/15 4/1

b) The parallel C1 set is invoked

   US 2/15 4/2

Note: This command will be extended to reset other terminal display functions yet to be defined.

3.2 Reset Control and Graphic Sets

The actions of this function are as follows:-

- The default graphic sets as described in Part 1 para 1.5.6 are designated.

- In the 7-bit environment the G0 set is invoked into columns 2 to 7 of the code table.

- In the 8-bit environment the G0 set is invoked into columns 2 to 7 of the code table and the G2 set is invoked columns 10 to 15 of the code table.

- Data following the Reset Control and Graphic Sets is to be interpreted as Alpha-mosaic data.

and
a) The serial C1 set is invoked

US 2/15 4/3

or
b) The parallel C1 set is invoked

US 2/15 4/4

3.3 Service Break to Row X

This function affects the terminal from the time it is received until the next US command is received. The terminal resets to the previous state before the next US command is executed. The action of this function is as follows:

Previous display states, including character sets, colours attribute controls and the active position will be stored in the terminal but no longer active.

Down loading processes to the terminal will be terminated.

The primary set of graphic characters is designated the G0 set and the supplementary set of graphic characters is designated the G2 set. Other character sets are not affected.

In the 7-bit environment the G0 set is invoked into columns 2 to 7 of the code table.

In the 8-bit environment the G0 set is invoked into columns 2 to 7 of the code table and the G2 set is invoked into columns 10 to 15 of the code table.

The format is unaffected but wrap-around is inactive.

The active position is set to the first character position of the designated row.

Only the following controls of the primary control function set are valid:

in the 7-bit environment
APB, APF, APR, CAN, SS2, ESC(in combination with a character from columns 4 or 5 of the code table), US.

in the 8-bit environment
APB, APF, APR, CAN, US.

If the serial C1 set is invoked the following controls are invalid:

in the 7-bit environment
5/0 to 5/7, 5/11, 5/14, 5/15.

in the 8-bit environment
9/0 to 9/7, 9/11, 9/14, 9/15.

The protected area attribute is inactive, all other attributes are unchanged.

Colour look up table is active.

Data following is interpreted as Alpha-mosaic.

If a Service Break to Row X is received by a terminal while it is executing a Timing Control Wait VPDE, the Service Break is executed immediately, after which the original wait will be continued.

Coding: US 2/15 <CS> (RN)

<CS> The designated C1 set is coded as follows:
4/0 = serial C1-set
4/5 = parallel C1-set

(RN) The designated row is coded from columns 4 to 7 of the code table. The row number is indicated by the binary value of the 6 least significant bits. If the row X specified is outside the D.D.A. following data is displayed on the bottom row of the D.D.A.

To reset to the previous state following a Service Break the following coding is defined:

US 2/15 4/15

This function should follow a Service Break to Row X. It is only valid used in this way.

The protected area attribute is inactive, all other attributes are unchanged.

Previous display states, including character sets, colours, attribute controls and the active position will be restored by the function Reset to the Previous States.

PART 9 TELESOFTWARE

(For further study)

TERMINAL FACILITY IDENTIFIER

CONTENTS

1.0 INTRODUCTION

The Terminal Facility Identifier may be used to ascertain the capabilities of a "terminal", (where a terminal may actually be a physical terminal or another videotex service). The TFI may be used in three ways:

1) To determine the terminal profile and additional terminal capabilities.

2) To determine to which parts of the SRM the terminal conforms.

3) To determine whether the terminal can execute a particular VPDE.

2.0 CODING STRUCTURE

Two VPCEs are used for the TFI, US 2/0 and US 2/1.

3.0 CODING

To request the terminal profile the code US 2/0 4/0 is transmitted to the terminal.

The terminal will reply with US 2/0 followed by a series of bytes from columns 4 to 7 of the code table (profiles and capabilities), or by a series of bytes from columns 4 and 5 of the code table, (representing the parts of the SRM to which the terminal conforms) terminating with the code 4/0.

To determine whether the terminal can execute a particular VPDE, the service transmits the code US 2/0 followed by the header of the VPDE and terminated by US 2/1 to the terminal. The terminal replies with either US 2/0 2/0 indicating it can execute the VPDE or with US 2/0 2/1 indicating it cannot execute the VPDE.

3.1 Conformance to the SRM

The following codes have been assigned for identifying parts of the SRM (see Annex B).

4/0 Terminator
4/1 Alphamosaic (part 1)
4/2 Geometric (part 2)
4/3 Photographic (part 3)
4/4 Define DRCS (part 4)
4/5 Define Colour (part 5)
4/6 Define Format (part 6)
4/7 Transparent data (part 7)
4/8 Reset (part 8)
4/9 Telesoftware (part 9)
4/10 Terminal Facility Identifier (part 10)
4/11 Timing Control (part 11)

A terminal conforming to parts 1,4,5,6,8 and 10 would transmit:

US 2/0, 4/1, 4/4, 4/5, 4/6, 4/8, 4/10, 4/0

If different levels of conformance are defined in the future (as for the Geometric display) then the level of conformance will be indicated by a byte from column 3 of the code table following the relevant byte from column 4.

For the Geometric mode (primary byte 4/2) the following sub levels will be defined:

3/0 - 3/7 reserved

3/8 - 3/15 sub-levels to be defined are for further study

A terminal which conforms to parts 1, 2 (sub-level coded 3/9), 4, 5, 6, 8 and 10 would transmit:

US 2/0, 4/1, 4/2, 3/9, 4/4, 4/5, 4/6, 4/8, 4/10, 4/0

3.2 Profiles

The following codes for terminal profiles have been assigned:

6/0	Alpha-mosaic profile 1 of Annex C
6/1	Alpha-mosaic profile 2 of Annex C
6/2	Alpha-mosaic profile 3 of Annex C
6/3	Alpha-mosaic profile 4 of Annex C
6/4-6/7	Reserved for Alpha-mosaic profiles
6/8	Geometric profile x1
6/9	Geometric profile x2
6/10-6/15	Reserved for future Geometric profiles
7/0-7/7	Reserved for Photographic profiles
7/8-7/14	Reserved for special profiles
7/15	Escape code. The following bytes (Cap. bytes) from columns 4-7 specify additional terminal capabilities by using a bitwise identification (Cap. bits) in the following way:

Cap. byte 1 :

-	1	1	0	1	1	1	1

extension bit
if b6 = 1: more Cap. bytes follow
if b6 = 0: last Cap. byte

reserved

Telesoftware capability

Transparent mode capability

80 character capability

Chipcard capability

The end of the TFI sequence is determined either by the code 4/0, or by the last Cap. byte (extension bit set to 0).

Examples:

a) US 2/0 6/1 4/0 : alphamosaic profile 2

b) US 2/0 6/0 6/9 7/15 4/6 =

-	1	0	0	0	1	1	0

: alphamosaic profile 1 with geometric profile x2, transparent mode and 80 character capability.

3.3 Ability to execute a particular VPDE

The service transmits the VPDE header, (without the US code), and terminates the header with a US 2/1.

If the service requires to know whether the terminal can execute 8 by 10 dot DRCS, it transmits the following codes to the terminal:

US, 2/0, 2/3, 2/0, 4/9, US, 2/1
(drcs header)

If the terminal can execute 8 by 10 dot DRCS it replies with an acknowlegement ie. the code US 2/0 2/0.

If it cannot it replies with the code US 2/0 2/1.

4. DEFAULTS

If no reponse is received from a terminal within a specified time (depending upon the transmission network being used) then the basic terminal used within that network is assumed.

TIMING CONTROL

CONTENTS

1.0 INTRODUCTION

When displaying data on a terminal, it may be desirable to introduce certain timing controls such as the instruction to wait for a specified period before continuing the display of data. These controls are coded in Control Videotex Presentation Data Elements (Control VPDE) in the following form:

US 2/4 < control identifier > < data >

The Control Identifier determines the type of control and may be followed by optional data. The following Control Identifier has been coded:

2/1 : WAIT

The remaining codes for the Control Identifier are for future standardisation.

2.0 WAIT

The Wait Control VPDE is coded as follows:

US 2/14 2/1 < period >

The period specifies the wait time, in tenths of a second, to be observed between the decoding of the Wait Control VPDE and the start of the execution of the VPDE following the Wait Control VPDE. Data received during a wait period is stored in the terminal after it has been scanned for the inclusion of a General Display Reset (Part 8 para 3.3) or a Service Break to Row X (Part 8 para 3.3). If a General Display Reset is detected the wait period is terminated and all the data stored in the terminal between the start of the wait and the detection of the General Display Reset is deleted. The General Display Reset is executed immediately.

If a Service Break to Row X is detected it is executed as soon as possible. The original wait will then continue. No data stored in the terminal is deleted.

The period is decimally coded, most significant digit first, by codes in the range 3/0 to 3/9 inclusive and is terminated by the code 3/11.

A Wait Control VPDE indicating a wait period of 30 seconds would be coded as follows:

US 2/14 2/1 3/3 3/0 3/0 3/11

APPENDIX 1 (to T.101 Annexe C)

VIDEOTEX SERVICE REFERENCE MODEL : CONFORMANCE

CONTENTS

APPENDIX 1 (to T.101 Annexe C)

Service Reference Model : Conformance to Data Syntax II

To conform to Data Syntax II a selection of specific facilities which should be supported by all services has been made. The degree to which a service implements these facilities is determined by the service profile. The service profiles in use are described in Appendix 2.

The rules for transcoding between the recognised profiles are for further study.

The following definitions apply:

RECOGNISE means to determine the syntactic form but not necessarily the semantics of a code sequence.

EXECUTE means to process a code sequence to allow the display of information conveyed by the code sequence and by subsequent code sequences.

PRESENT means to display the information conveyed by a code sequence and in the case of a control function, to display information affected by the control function.

Part 0 General

All Videotex Presentation Control Elements (VPCEs) shall be recognised (see , Part 0, section 1.1.1).

Part 1 Alpha-mosaic Displays

1.1 Alpha-numeric characters

1.1.1 Simple Alpha-numeric characters

The primary set of graphic characters (Part 1 Table 8), excluding character positions 2/3, 2/4, 4/0, 5/11 to 5/14, 6/0 and 7/11 to 7/14 (which may be presented using fall-back characters) shall be recognised and presented.

1.1.2 Extended Alpha-numeric characters

All Alpha-numeric characters of the repertoire (see Part 1 Section 2.1.1) shall be recognised and presented.

1.2 Mosaic and other characters

1.2.1 Simple Mosaic and other characters

The block-mosaic character (see Part 1 Section 2.1.2) shall be recognised and presented.

1.2.2 Extended Mosaic and other characters

All mosaic, line drawing and miscellaneous characters of the repertoire shall be recognised and presented.

1.3 Format Effectors

1.3.1 Simple format effectors

The following format effectors; APB, APF, APD, APR, APU, CS and APH shall be recognised and executed.

1.3.2 Extended format effectors

All format effectors shall be recognised and executed, (see Part 1 Section 2.2)

1.4 Attribute Controls

1.4.1 Parallel attribute controls

The following shall be recognised and presented:

-Foreground colours
-Background colours
-Start lining and stop lining
-Normal size, double-height, double-width, double-size
-Flash and steady
-Conceal display and stop conceal
-Inverted polarity and normal polarity
-Start box and end box

1.4.2 Serial attribute controls

The following shall be recognised and presented:

-Alpha and Mosaic foreground colours
-New background and black background
-Start lining and stop lining
-Normal size and double height
-Flash and steady
-Conceal Display and Stop Conceal
-Start box and end box

1.4.3 Extended attribute controls

The following parallel and serial attribute controls shall be recognised and presented:

-Flash states: Flash, Steady, Inverted Flash and Reduced Intensity Flash
-Flash rates: Normal Flash, Fast Flash (three phases), Increment Flash and Decrement Flash
-Marking

Note: The fall back for Flash rate and Flash state is Normal Flash.
The fall back for Marking is Non Marking.

1.4.4 Full Row attribute controls

The following shall be recognised and presented:

-Foreground Colours
-Background Colours
-Lining
-Normal Size
-Normal Flash
-Steady
-Invert
-Window
-Conceal
-Protecting

Note: The fall back for Protecting is Non Protecting.

1.4.5 Full Screen Attribute Controls

The following shall be recognised and presented:

-Background Colour

1.4.6 Scrolling

1.4.6.1 Simple Scrolling

Implicit scrolling shall be executed.

1.4.6.2 Extended Scrolling

The definition of one scrolling area shall be recognised and executed.

Implicit and explicit scrolling shall be recognised and executed.

1.5 Device controls

All device control functions shall be recognised (see Part 1 Section 2.4).

Cursor On and Cursor Off shall be executed.

1.6 Coding Structure

7-bit or 8-bit coding shall be executed.

1.7 Invocation of Character Sets

The invocation of character sets for the 7 or 8 bit environment as appropriate shall be executed (see Part 1 Sections 3.1.2 and 3.1.3).

1.8 Designation of Character Sets

The designation of 4 fixed character sets shall be executed, (see . Part 1 Section 3.4.3).

Note that no designation sequence is required for the set as this is fixed.

1.9 Colour Table Controls

The invocation of four colour tables shall be recognised and executed. (see Part 1 Section 3.5.6).

Part 2 Geometric Display

(For further study)

Part 3 Photographic Displays

3.1 General

All header and transfer units (see Part 3 Sections 2.0, 3.0, 4.0 and 5.0) shall be recognised.

For further study

Part 4 Define DRCS

4.1 Designation and Identification

The designation of one DRCS set shall be executed, (see Part 4 Section 5.0).

Note: The library identification of the DRCS set is given in the down loading sequence.

The default Identify Character Set (ICS) (see Part 4 Section 2.1) shall be executed.

4.2 Character Matrices

4.2.1 Preferred Character Matrices

Select Dot Composition (SDC) type 2 (see Part 4 Section 2.2.2) shall be executed for the following character matrices:

12 X 10	6 X 10	6 X 5
12 X 12	6 X 12	6 X 6

4.2.2 8 dot type Matrices

8 X 10	4 X 10	4 X 5

The 8 dot type matrix set may be implemented as an alternative but where this is done the means shall be
(2604) provided within the system for translating from and to systems having character matrices of a 12 X 10 type and its derivatives.

4.3 Bits Per Dot

4.3.1 Basic DRCS

Basic DRCS shall be executed and presented. (see Annex A Part 4 Section 2.2.2)

4.3.2 Colour DRCS

Colour DRCS shall be executed and presented with 4 and 16 colours per character, (see - Part 4 Section 2.2.2), but 4 colour DRCS is acceptable for a transitional period.

4.4 Coding Method

The direct coding method shall be executed. (see Part 4 Section 2.3).

4.5 Addressing Capability

One DRCS set of 94 characters of basic DRCS shall be executed and presented.

Note: The presentation of the characters will be dependent upon the capabilities of the display device.

Part 5 Define COLOUR

5.1 Structure of the Colour Map

5.1.1 Size

The colour map shall consist of 32 colours. (see Part 5 Section 2.0).

5.1.2 CLUTs

Four fixed CLUTs (each of 8 colours) shall be executed and presented.

5.1.3 DCLUTs

Two DCLUTs shall be executed and presented:

- one for 4 colour DRCS
- one for 16 colour DRCS

5.2 Definition

The RGB loading method shall be executed. (see - Part 5 Section 3.3.2.).

5.2.1 Colour Map

The definition of colours 16 to 31 of a single colour map (ICT 2/0 2/0) shall be executed (see Part 5 Section 3.1).

5.2.2 DCLUTs

The definition of a single DCLUT for use with 4 colour DRCS (ICT 2/2 2/0) shall be executed (see Part 5 Section 3.1)

5.2.3 Resolution

A Select Unit Resolution (SUR) of 4 for the colour map shall be executed (see Part 5 Section 3.1).

A Select Unit Resolution (SUR) of 5 for the DCLUT for use with 4 colour DRCS shall be executed (see Part 5 Section 3.1).

5.2.4 Reset

Part 6 Define FORMAT

6.1 Coding

All codings of Define FORMAT shall be recognised.

6.2 Format and Wrap-around

6.2.1 Simple

A format of 24 rows of 40 columns shall be presented with automatic wrap-around.

6.2.2 Extended

A format of 20 rows (US 2/13 4/2) of 40 columns shall be presented (see Part 6 Section 2.0).

The wraparound ON and OFF commands (see Part 6 Section 2.0) shall be executed.

Part 7 Transparent Data

The whole of the transparent data mode shall be executed (see Part 7).

Part 8 Reset

All reset sequences shall be recognised (see Part 8). All those which affect the implemented display modes shall be executed.

Part 9 Telesoftware

For further study.

Part 10 Terminal Facility Identifier

All TFI sequences shall be executed (see Part 10).

Part 11 Timing Control

All timing control shall be executed (see Part 11).

APPENDIX 2 (to T.101 Annexe C)

VIDEOTEX SERVICE REFERENCE MODEL : PROFILES

CONTENTS

APPENDIX 2 (to T.101 Annexe C)

Service Reference Model - Profiles of terminals applying to Data Syntax II

A. Alphamosaic Profiles

The following profiles are recognised for existing Videotex services.

A.1 Profile 1

This is the profile including the first horizon (see
It uses 8 bit coding and satisfies the following parts of the conformance requirements described in App 1

Part 0 General

Part 1 Alphamosaic Displays (8 bit coding)

Part 4 Define DRCS (except 4.2.2)

Part 5 Define Colour

Part 6 Define Format

Part 8 Reset

Part 10 Terminal Facility Identifier

A.2 Profile 2

This is the profile of a service which implements only parallel attribute controls and extended format effectors.

It uses 7 bit coding and satisfies the following parts of the conformance requirements described in App 1

Part 1 Alphamosaic displays

The following paragraphs only:

1.1.1 Simple alpha-numeric characters

1.1.2 Some extended G2 characters:

2/3, 2/4, 2/6, 2/12, 2/13, 2/14, 2/15

3/0, 3/1

4/1, 4/2, 4/3, 4/8, 4/11

6/10

7/10

Note: The following accented and special letters shall be displayed:

the 13 lower case letters:

à, è, ù, é, â, ê, î, ô, û, ë, ï, œ, ç.

the 8 upper case letters:

Â, É, È, Î, Ô, Ù, Œ, Ç.

1.2.1 Simple mosaic and other characters

1.3 Format effectors

1.4.1 Parallel attribute controls

Notes: 1. Background colour controls, start lining and stop lining require a space when used with alpha-numeric characters.

2. Start Box and End Box require a space.

3. Double Height and Double Size controls cannot be used on adjacent rows.

1.4.6.1 Implicit scrolling of the whole screen

1.5 Device controls

1.6 Coding structure -7 bit coding

1.7 Invocation of character sets

Part 6 Format and wraparound

The following paragraph only:

6.2.1 Simple format and wrap-around

A.3 Profile 3

This is the profile of a service which implements only serial attribute controls and simple format effectors.

It uses 7 bit coding and satisfies the following parts of the conformance requirements described in

Part 1 Alphamosaic displays

The following paragraphs only:

1.1.1 Simple alpha-numeric characters

1.2.1 Simple mosaic and other characters

1.3.1 Simple format effectors

1.4.2 Serial attribute controls (excluding Stop Conceal)

Note: Attribute controls usually require a space.

1.5 Device controls

1.6 Coding structure -7 bit coding

1.7 Invocation of character sets

Part 6 Define Format

The following paragraph only:

6.2.1 Simple format and wrap-around

A.4 Profile 4

This is the profile of a service which implements only serial attribute controls and extended format effectors.

It uses 7 bit coding and satisfies the following part of the conformance requirements described in App 1

Part 1 Alphamosaic displays

The following paragraphs only:

1.1.1 Simple alpha-numeric characters

1.2.1 Simple mosaic and other characters

1.3 Format effectors

1.4.2 Serial attribute controls (excluding Stop Conceal)

Note: Attribute controls usually require a space.

1.5 Device controls

1.6 Coding structure -7 bit coding

1.7 Invocation of character sets

Part 6 Define Format

The following paragraph only:

6.2.1 Simple format and wrap-around

B. Geometric Profiles

(For further study)

C. Photographic Profiles

For further study.

D Transparent Data Profiles

For further study.

E Telesoftware Profiles

For further study.

APPENDIX 3 (to T.101 Annexe C)

FUTURE HORIZON

Fig 1c

STANDARD FOR NEW ENHANCED SERVICES AS YET UNFORESEEN OR NOT YET DEFINED IN SERVICES (OPTIONAL)

- must be based on one of the four basic building blocks and "harmonised enhanced" service

  or

  must be based on one of the four basic building blocks and be able to display information generated by systems using the harmonised enhanced service standard.

- must be able to be adopted without modifications to any parameters defined in the references given in Fig 1b

1ST HORIZON

Fig 1b

STANDARD FOR THE "HARMONISED ENHANCED" SERVICE (OPTIONAL)

Annex B	0	6
	1.1.2	6.2.2
	1.3.2	8
	1.4.1 and 1.4.2 (simultaneously)	10
	1.4.3 to 1.4.5	
	1.5 to 1.9	
	4	
	5	

Fig 1a

FLEXIBLE INITIAL BUILDING BLOCKS
(alternatives)

Profile 1	Profile 2	Profile 3	Profile 4

See Rec 2.4 (Optional)

FIG 1 - CEPT RECOMMENDATION ON ENHANCED VIDEOTEX SERVICE AND ITS RELATIONSHIP WITH EXISTING BASIC VIDEOTEX SERVICE SYSTEMS AND NEW AS YET UNDEFINED FACILITIES

(2604)