INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

ANNEX TO RED BOOK

FASCICLE VII.3 – ANNEX II

DATA SYNTAX II FOR INTERNATIONAL INTERACTIVE VIDEOTEX SERVICE

RECOMMENDATION T.101, ANNEX C

VIIITH PLENARY ASSEMBLY
MALAGA-TORREMOLINOS, 8-19 OCTOBER 1984

Geneva 1985

INTERNATIONAL TELECOMMUNICATION UNION

CCITT

THE INTERNATIONAL
TELEGRAPH AND TELEPHONE
CONSULTATIVE COMMITTEE

ANNEX TO RED BOOK

FASCICLE VII.3 – ANNEX II

DATA SYNTAX II FOR INTERNATIONAL INTERACTIVE VIDEOTEX SERVICE

RECOMMENDATION T.101, ANNEX C

VIIITH PLENARY ASSEMBLY
MALAGA-TORREMOLINOS, 8-19 OCTOBER 1984

Geneva 1985

ANNEX C

(to Recommendation T.101)

Data Syntax II

Note : This data syntax generally corresponds to the "CEPT" presentation layer data syntax adopted by some European countries.

VIDEOTEX PRESENTATION LAYER DATA SYNTAX

CONTENTS

PART 0 - GENERAL

CONTENTS

1.0 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. Alphamosaic

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' (PSDUs) which contain the data itself.

```
<---------------- 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:

Define DRCS:	US 2/3
Define COLOUR:	US 2/6
Define FORMAT:	US 2/13
RESET	US 2/15
ALPHAMOSAIC display data:	US <any character from column 4-7>
GEOMETRIC display data (2D)	US 3/0
GEOMETRIC display data (3D)	US 3/1
PHOTOGRAPHIC pixel data	US 3/4
PHOTOGRAPHIC table data	US 3/5
TRANSPARENT data	US 3/15

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, .75).

The photographic display area also maps to the same area. The top lefthand pixel of the photographic dispaly 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 as closely compatible as possible with the following recommendations and standards:

CCITT Recommendation V3	International alphabet No 5
CCITT Recommendation F300	Videotex service
CCITT Recommendation S100	International information exchange for interactive videotex
ISO Standard 2022 (Rev 79)	Code Extension Techniques for use with the ISO 7-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.

PART 1 - ALPHA MOSAIC DISPLAYS

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 location.

2. One attribute memory where all the attributes are 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 disolay 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 of the CCITT modes 'parallel' and 'serial' 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 leyer (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

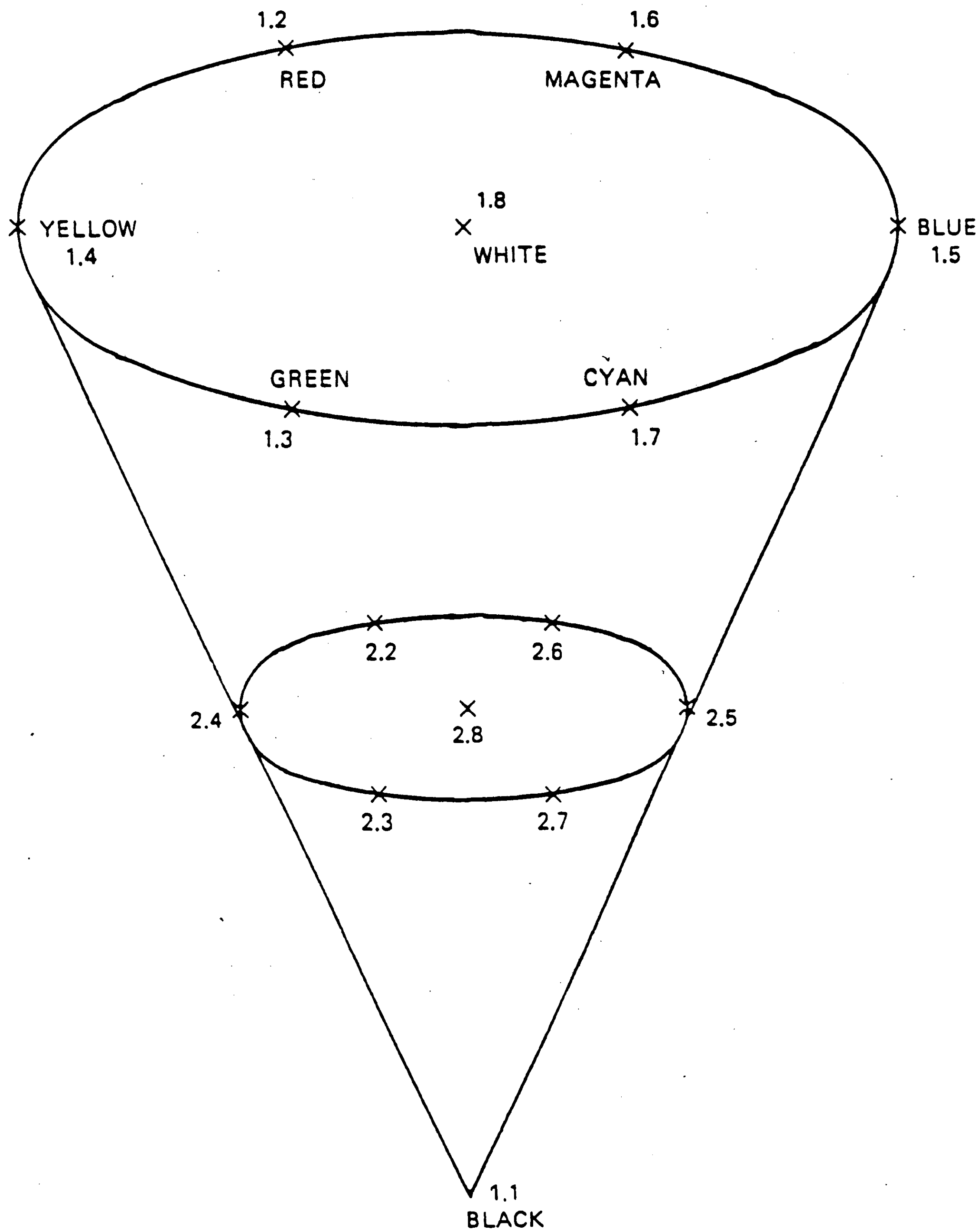


TABLE 4 RED GREEN AND BLUE COMPONENTS OF DEFAULT COLOUR MAP

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 PPCI. The data following the PPCI 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	CODE POS	CODE SET	CODE 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	CODE POS	CODE SET	CODE 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	POS	SET	POS
LD21	ď or d̕	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	İ	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	į	small i with ogonek	G2	4/14	G0	6/9
LI44	Į	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	SET	POS
LL61	ł	small l with stroke	G2	7/8		
LL62	Ł	capital L with stroke	G2	6/8		
LL63	l·	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	'n	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	CODE POS	CODE SET	CODE POS
LO16	Ô	capital O with circumflex	G2	4/3	GO	4/15
LO17	ö	small o with diaeresis or umlaut	G2	4/8	GO	6/15
LO18	Ö	capital O with diaeresis or umlaut	G2	4/8	GO	4/15
LO19	õ	small o with tilde	G2	4/4	GO	6/15
LO20	Õ	capital O with tilde	G2	4/4	GO	4/15
LO25	ő	small o with double acute accent	G2	4/13	GO	6/15
LO26	Ő	capital O with double acute accent	G2	4/13	GO	4/15
LO31	ō	small o with macron	G2	4/5	GO	6/15
LO32	Ō	capital O with macron	G2	4/5	GO	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	GO	7/0		
LP02	P	capital P	GO	5/0		
LQ01	q	small q	GO	7/1		
LQ02	Q	capital Q	GO	5/1		
LR01	r	small r	GO	7/2		
LR02	R	capital R	GO	5/2		
LR11	ŕ	small r with acute accent	G2	4/2	GO	7/2
LR12	Ŕ	capital R with acute accent	G2	4/2	GO	5/2
LR21	ř	small r with caron	G2	4/15	GO	7/2
LR22	Ř	capital R with caron	G2	4/15	GO	5/2
LR41	ŗ	small r with cedilla	G2	4/11	GO	7/2
LR42	Ŗ	capital R with cedilla	G2	4/11	GO	5/2
LS01	s	small s	GO	7/3		

ID	GRAPHIC	NAME OR DESCRIPTION	CODE SET	CODE POS	CODE SET	CODE 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	CODE SET	CODE POS
LY11	ý	small y with acute accent	G2	4/2	GO	7/9
LY12	Ý	capital Y with acute accent	G2	4/2	GO	5/9
LY15	ŷ	small y with circumflex	G2	4/3	GO	7/9
LY16	Ŷ	capital Y with circumflex	G2	4/3	GO	5/9
LY17	ÿ	small y with diaeresis or umlaut	G2	4/8	GO	7/9
LY18	Ÿ	capital Y with diaeresis or umlaut	G2	4/8	GO	5/9
LZ01	z	small z	GO	7/10		
LZ02	Z	capital Z	GO	5/10		
LZ11	ź	small z with acute accent	G2	4/2	GO	7/10
LZ12	Ź	capital Z with acute accent	G2	4/2	GO	5/10
LZ21	ž	small z with caron	G2	4/15	GO	7/10
LZ22	Ž	capital Z with caron	G2	4/15	GO	5/10
LZ29	ż	small z with dot	G2	4/7	GO	7/10
LZ30	Ż	capital Z with dot	G2	4/7	GO	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	CODE 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	TM	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 (not jointive)	G0	7/14
SM48	_	lower bar (not jointive)	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;

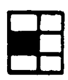
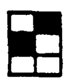
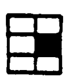
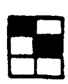
24 line drawing graphics;

4 jointive arrows;

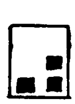
4 miscellaneous drawing graphics including one graphic with a dot-pattern where approximately 40% of the character cell area has the foreground colour and the remaining area has the background colour.

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.

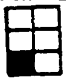
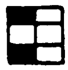
Block Mosaic Graphics

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

There are two fonts for mosaic graphics characters: 'contiguous' and 'separated'

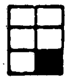
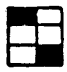
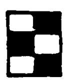
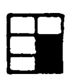
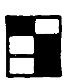
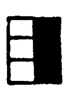


separated graphics representation

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

There are two fonts for mosaic graphics characters: 'contiguous' and 'separated'

separated graphics representation

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

There are two fonts for mosaic graphics characters: 'contiguous' and 'separated'

 separated graphics representation

ID	GRAPHIC	CODE SET	CODE 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

There are two fonts for mosaic graphics characters: 'contiguous' and 'separated'

separated graphics
representation

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

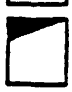
There are two fonts for mosaic graphics characters: 'contiguous' and 'separated'


separated graphics representation

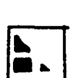
ID	GRAPHIC	CODE SET	POS
SG15		G1	5/13
SG16		G1	5/12
SG17		G1	5/11
SG18		G1	5/10
SG19		G1	5/9
SG20		G1	5/8
SG21		G1	5/7
SG22		G1	5/6
SG23		G1	5/5
SG24		G1	5/4
SG25		G1	5/3
SG26		G1	5/2
SG27		G1	5/1.
SG28		G1	5/0

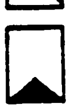
There are two fonts for mosaic graphics characters: 'contiguous' and 'separated'

separated graphics representation

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

There are two fonts for mosaic graphics characters: 'contiguous' and 'separated'

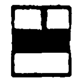
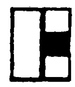
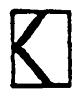
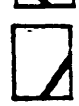
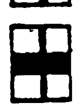
 separated graphics representation

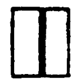
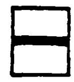
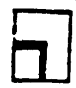
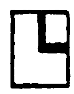
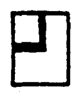
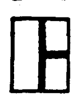
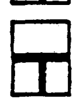
ID	GRAPHIC	CODE SET	CODE POS
SG43		G3	7/0
SG44		G3	7/1
SG45		G3	7/2
SG46		G3	7/3
SG47		G3	7/4
SG48		G3	7/5
SG49		G3	7/6
SG50		G3	7/7
SG51		G3	7/8
SG52		G3	7/9
SG53		G3	7/10
SG54		G3	7/11
SG55		G3	7/12
SG56		G3	7/13

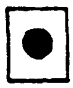
There are two fonts for mosaic graphics characters: 'contiguous' and 'separated'

separated graphics representation

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 grpahic 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
MCB	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 5/7
- 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 5/7
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 C1 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 C1 set.

The second supplementary set of mosaic characters is designated the G1 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)

				b_7	0	0	0	0	1	1	1	1
				b_6	0	0	1	1	0	0	1	1
				b_5	0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1		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	SO	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.

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 invoked by the sequence ESC 2/2 4/0.

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

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

TABLE 6 THE SERIAL SUPPLEMENTARY CONTROL FUNCTION SET

				b_7	0	0	0	0	1	1	1	1
				b_6	0	0	1	1	0	0	1	1
				b_5	0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1		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	BBD		
1	1	0	1	13					DBH	NBD		
1	1	1	0	14					DBW	HMS		
1	1	1	1	15					OBS	RMS		

TABLE 7 THE PARALLEL SUPPLEMENTARY CONTROL FUNCTION SET

				b_7	0	0	0	0	1	1	1	1
				b_6	0	0	1	1	0	0	1	1
				b_5	0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_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	SCD		

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 G0 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	|
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 LINE) 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
					0	1	2	3	4	5	6	7
b4	b3	b2	b1									
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)

				b_7	0	0	0	0	1	1	1	1
				b_6	0	0	1	1	0	0	1	1
				b_5	0	1	0	1	0	1	0	1
b_4	b_3	b_2	b_1		0	1	2	3	4	5	6	7
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 LINE) 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)

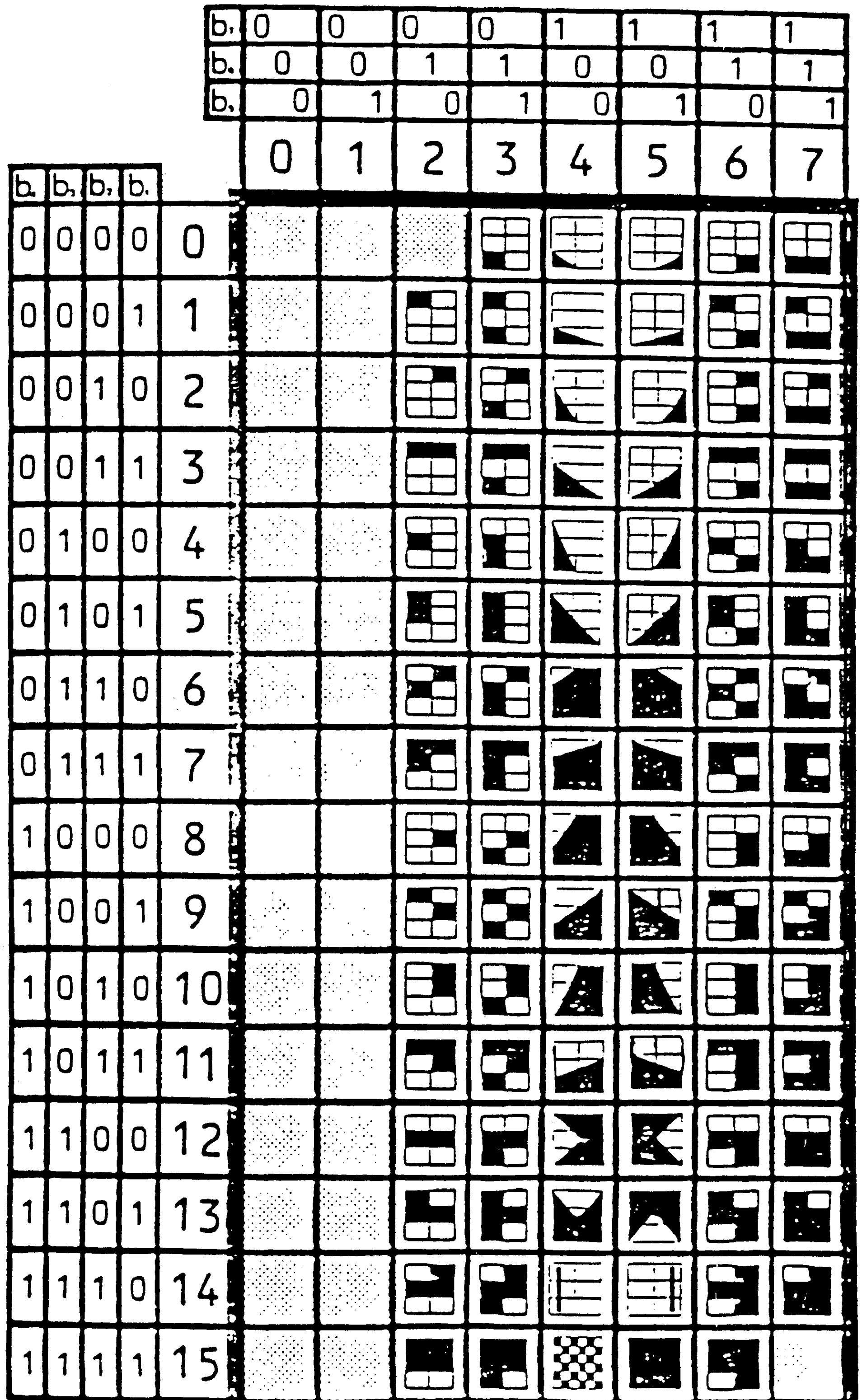
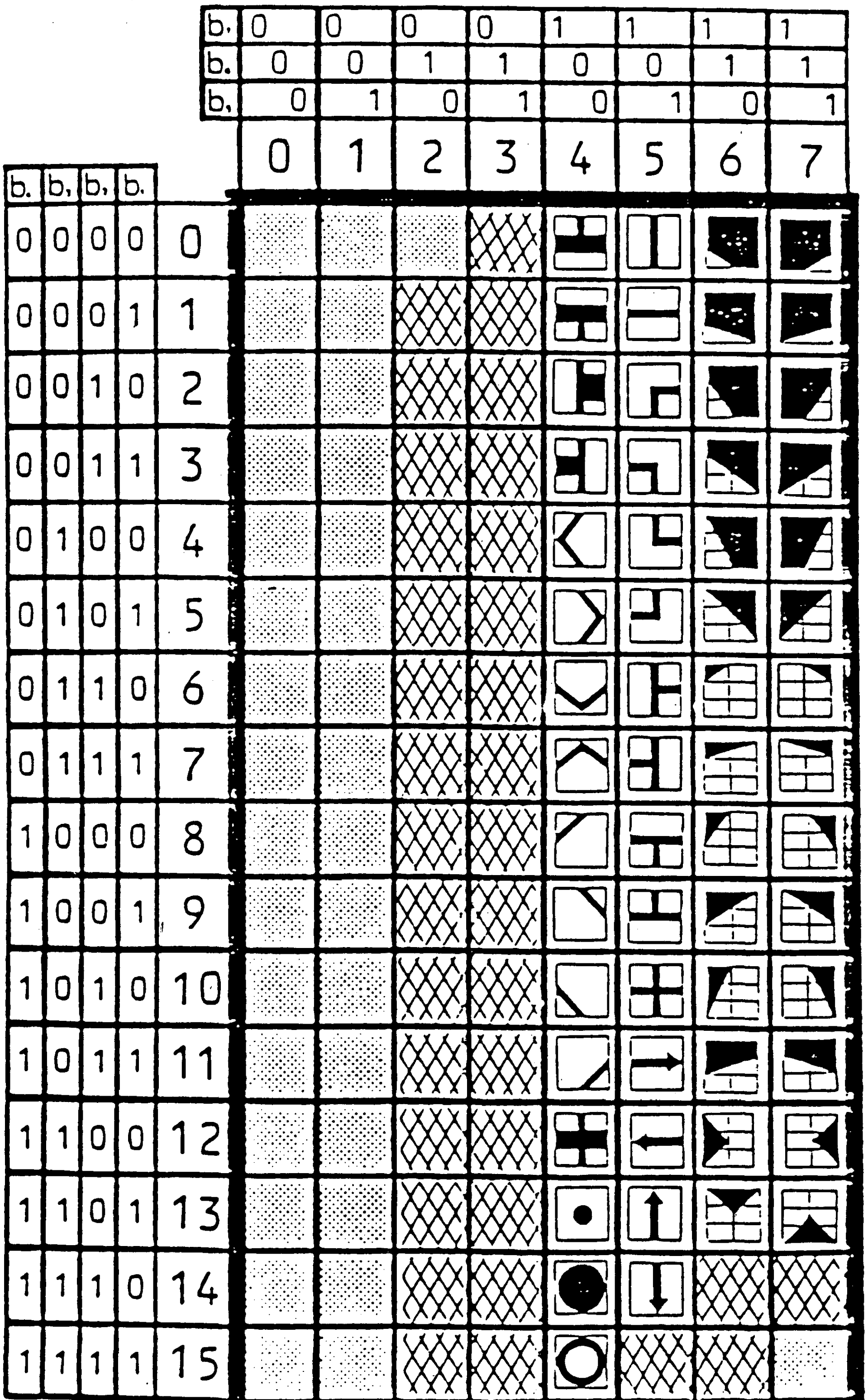


TABLE 12 THE THIRD SUPPLEMENTARY SET OF MOSAIC CHARACTERS (DEFAULT G3 SET)



(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 (G1)	to G0
ESC	2/9	6/4	:	to G1
ESC	2/10	6/4	:	to G2
ESC	2/11	6/4	:	to G3

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 fcur-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

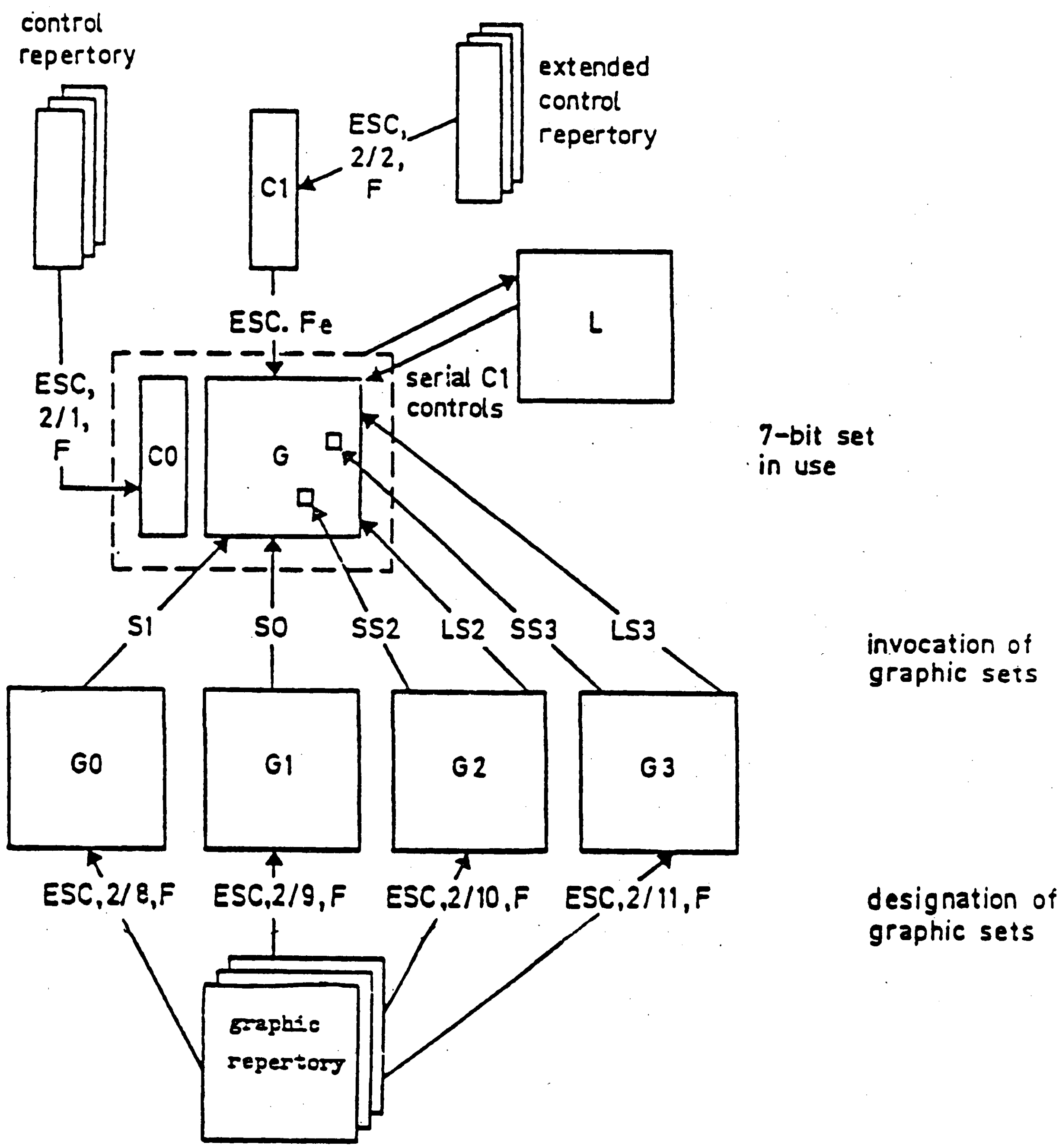


FIGURE 2 CODE EXTENSION IN A 7-BIT ENVIRONMENT

* See section 3.1.2.

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

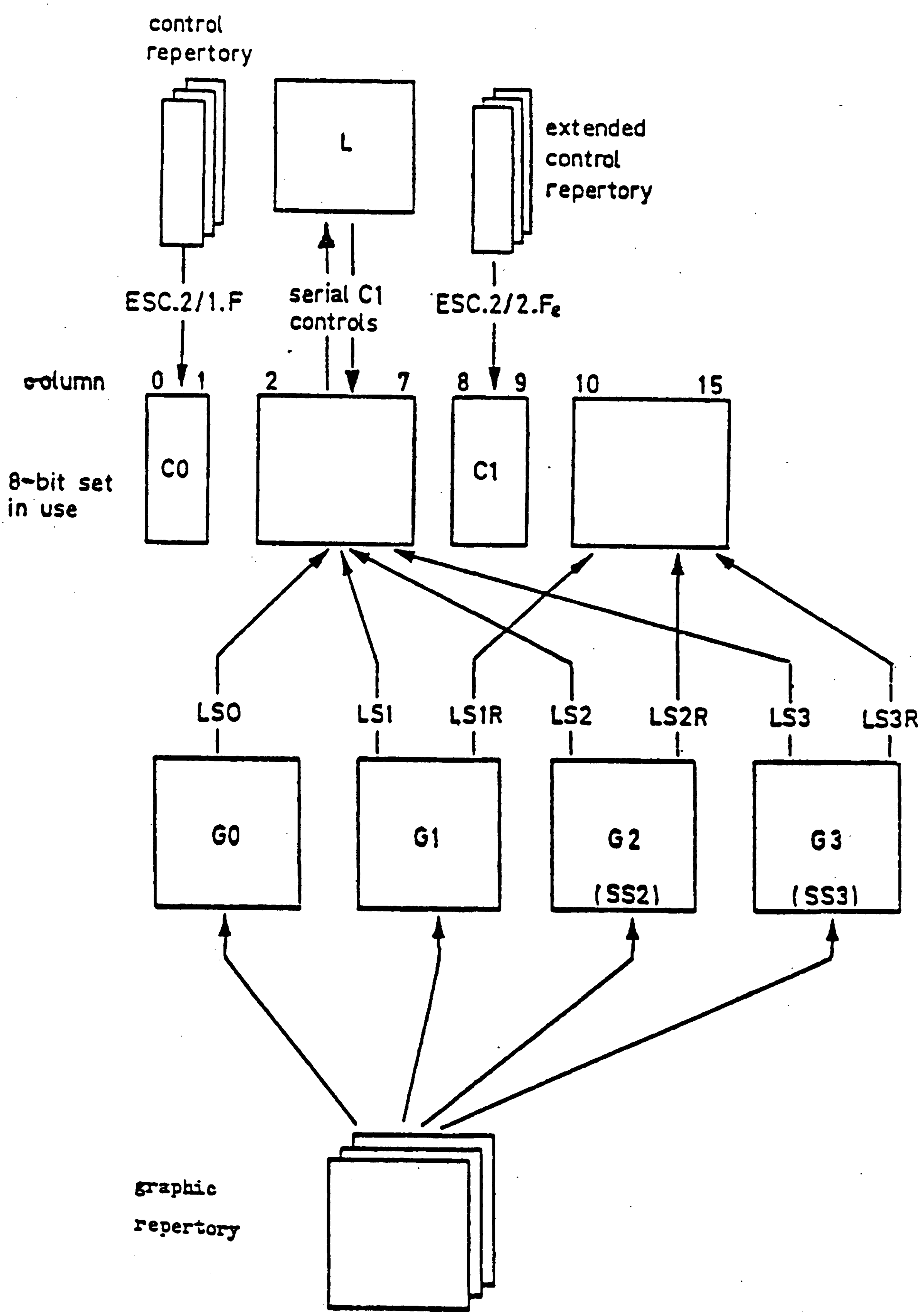


FIGURE 3 CODE EXTENSION IN AN 8-BIT ENVIRONMENT

* See section 3.1.2.

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.
```

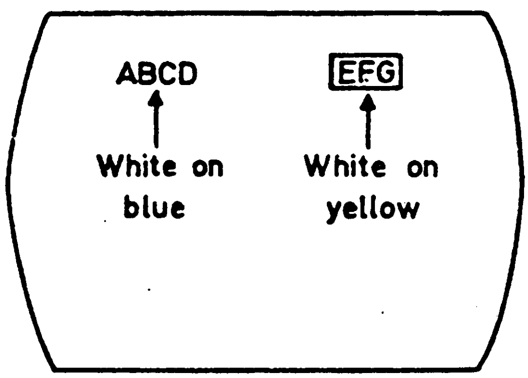
FIGURE 4   IDENTIFICATION SYSTEM

APPENDIX B

EXAMPLES OF TIME DEPENDENCY IN THE UNIFIED ALPHAMOSAIC MODEL

EXAMPLE 1

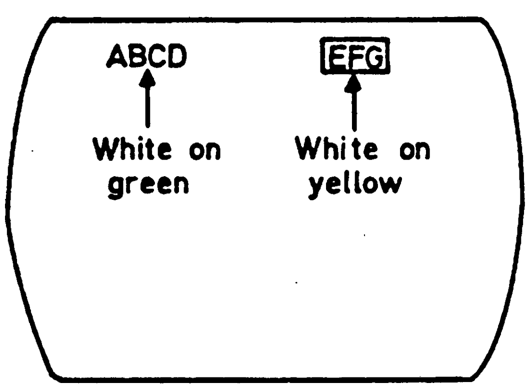
Full screen blue



Codes

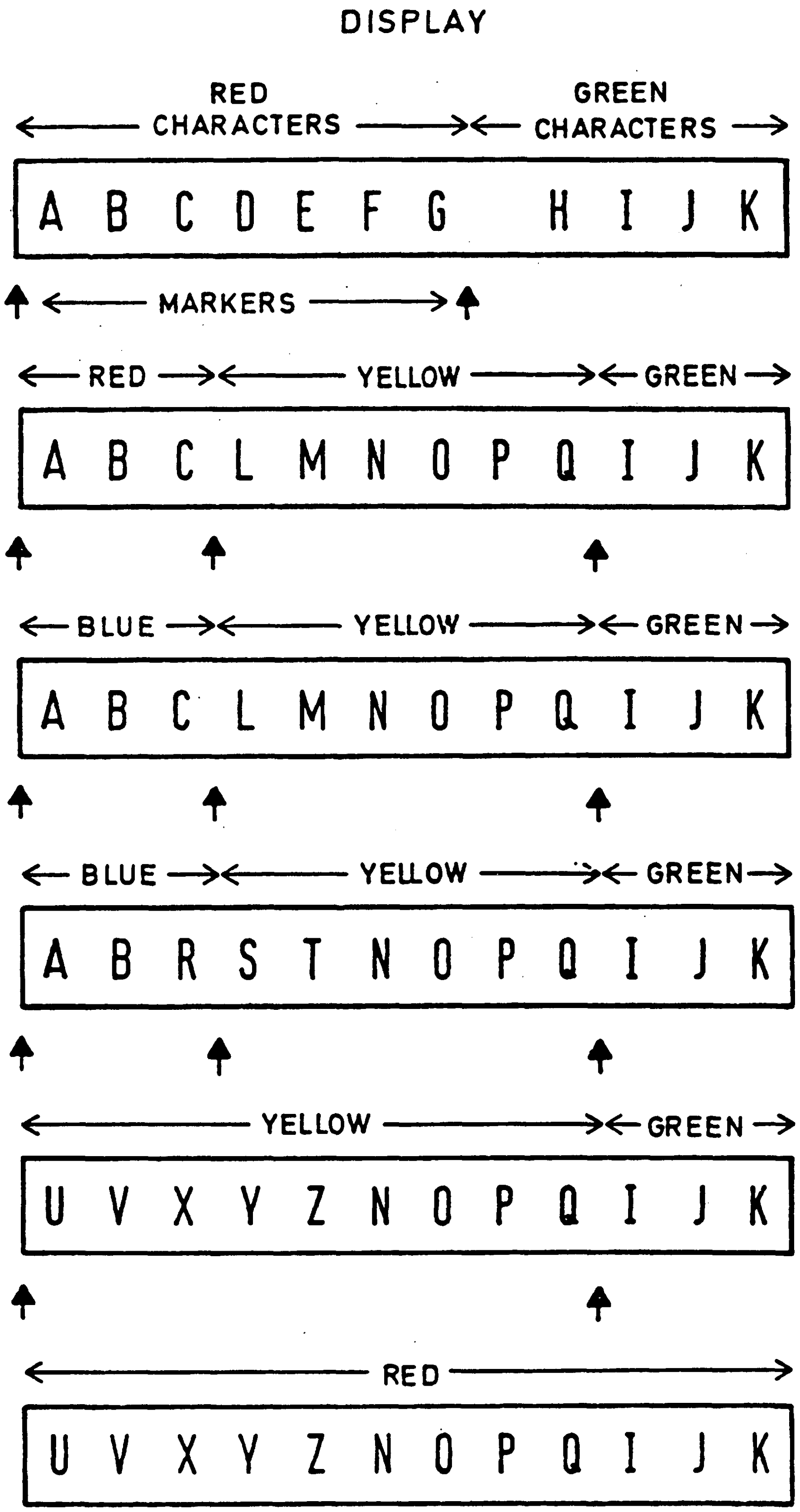
CS, full screen blue background, (transparent background), A, B, C,D yellow background, E,F,G.

Full screen green



Full screen green background.

EXAMPLE 2



SEQUENCE OF CODES.

CS
ESC,2/2,Fs
ESC,4/1,APB
A,B,C,D,E,F,G,
ESC,4/2
H,I,J,K

APR
APF,APF,APF
ESC,2/2,Fp
ESC,4/3
L,M,N,O,P,Q

APR
ESC,2/2,Fs
ESC,4/4

APF,
R,S,T

APR
ESC,2/2,Fp
U,V,X,Y,Z

Row attribute set to red foreground.

PART 2 - GEOMETRIC DISPLAYS

CONTENTS

List of Figures and Tables

1.0 INTRODUCTION

This part contains a detailed description of a geometric set. It specifies a set of primitives and their attributes with which a complete geometric set becomes available.

The specification is set up in such a way that it can be easily extended to cope with more enhanced facilities.

This part defines the geometric set in terms of:

a. Geometric primitives

b. Attributes of the geometric primitives

c. Display terminal characteristics

d. Coding of the primitives and the attributes.

This part is structured as follows:
Sections 1 to 4 give a general description of the geometric set. An overview of the specification is given in section 5 and sections 6 and 7 give the detailed specification. Appendix A contains a description of the B-spline curves.

2.0 SCOPE AND FIELD OF APPLICATION

This specification describes the geometric set for use in interactive videotex services. It is an extension of the alphamosaic set.

This specification is intended to be as closely compatible as possible with the recommendations and standards as defined in part 0 Chapter 2, as well as with ISO/DIS 7942 Information Processing - Graphical Kernel System (GKS).

With this specification it is possible to create a picture which can be displayed on a suitably equipped videotex terminal.

The pictures created with this specification are device independent. The device independency is achieved by handling coordinate data in two separate coordinate systems. All the functions and facilities offered in this specification are related to device independent data. The mapping of device independent data into device dependent data is performed in the videotex terminal.

This specification does not constrain any particular implementation of a videotex system.

3.0 DEFINITIONS

This chapter contains the definition of terms used in the specification of the geometric set. The definitions given in part 0 Chapter 3 also apply to this specification.

ASPECT RATIO:
The ratio of the width to the height of a rectangular area, such as window or 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.
Examples: Highlighting, colour, line thickness.

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.

BUNDLE TABLE:
An indexed table containing a set of attributes for each index.

CELL ARRAY:
A display element (graphic primitive) consisting of a rectangular grid of equal size rectangular cells, each having a single colour.

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

CLIPPING RECTANGLE:
A specified rectangular region of the normalized coordinate space which may be used when the picture is clipped during mapping to the device coordinates.

CONTROL:
A display element (graphic primitive) that can be used to change the characteristics of the display space.

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:
(Also graphic or videotex terminal)
A device on which display images can be represented.

DISPLAY ELEMENT:
(Also graphic primitive or primitive)
A basic graphic element that can be used to construct a display image.

DISPLAY IMAGE:
(Also picture)
A collection of display elements that are represented together on a display surface.

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

DISPLAY SURFACE:
(Also view surface)
In a display device, that medium on which display images may appear.

DOMAIN:
A CONTROL-sub-primitive that defines the accuracy (number of bits) of the coordinate data.

FILL AREA:
A SET-sub-primitive that defines attributes to be used in filling a closed boundary.

FILL AREA BUNDLE TABLE:
A table associating specific values of FILL AREA attributes with a fill area bundle index. In this geometric set, this table contains entries consisting of fill area colour, fill area interior style and fill area style index.

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

GRAPHIC CLASS:
(Also Class)
A set of display elements and attributes that forms a subset of the defined geometric set.

GRAPHIC OBJECT:
A display element (graphic primitive) that manipulates segments.

GRAPHIC PRIMITIVE:
(See display element)

GRAPHIC TERMINAL:
(See display device)

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

INFILL:
A display element (graphic primitive) that fills within a boundary with a specified interior style.

NORMALIZED DEVICE COORDINATE (NDC):
A coordinate specified in a device independent intermediate coordinate system, normalized to some range, typically 0 to 1.

OUTPUT PRIMITIVE:
A display element (graphic primitive) that actually generates (parts of) a display image. In this geometric set: POLYLINE, FILL AREA, POLYMARKER, CELL ARRAY, TEXT, INFILL AND GDP.

PICTURE:
(See display image)

PICTURE ELEMENT:
(See Pixel)

PIXEL:
(Also picture element)
The smallest element of a display surface that can be independently assigned a colour or intensity.

FILL AREA:
A display element (graphic primitive) consisting of a set of connected lines with a closed boundary.

POLYLINE:
A display element (graphic primitive) consisting of a set of connected lines.

POLYLINE BUNDLE TABLE:
A table associating specific values of POLYLINE attributes with a polyline bundle index. In this geometric set, this table contains entries consisting of polyline colour, line type and line width.

POLYMARKER:
A display element (graphic primitive) consisting of a set of locations to be indicated by a marker.

POLYMARKER BUNDLE TABLE:
A table associating specific values of POLYMARKER attributes with a polymarker bundle index. In this geometric set, this table contains entries consisting of polymarker colour, marker type and marker size.

PRIMITIVE:
(See display element)

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. In this geometric set 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. In this geometric set restricted to segments.

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

SPLINE:
A smooth curve, piecewise polynomial function, drawn through a series of control points. In this geometric set a uniform quadratic B-spline.

SUB-PRIMITIVE:
A display element (graphic primitive) that forms a part of a general display element.

TEXT:
A display element (graphic primitive) consisting of a character string.

TEXT BUNDLE TABLE:
A table associating specific values of TEXT attributes with a text bundle index. In this geometric set, this table contains entries consisting of text colour, character height and width, character spacing and text font and precision.

TRANSLATION:
The application of a constant displacement to the position of all or part of a display image. In this geometric set restricted to segments.

VIDEOTEX TERMINAL:
(See display device)

VIEWING TRANSFORMATION:
(See window/viewport transformation).

VIEWPORT:
A specified rectangular region of device coordinate space.

VIEW SURFACE:
(See display surface).

VIRTUAL SPACE:
A device independent area in which display images are defined.

WINDOW:
A specified part of virtual space, a rectangular region of the normalized device coordinate space.

3.1 Abbreviations

ASF	Aspect Source Flag
BGU	Basic Grid Unit
DC	Device Coordinates
DCC	Differential Chain Code
GDP	Generalized Drawing Primitive
NDC	Normalized Device Coordinates

4.0 GENERAL DESCRIPTION OF THE GEOMETRIC SET

The geometric set defined in this document specifies the functions and facilities that offer a mechanism to handle pictures in a videotex service. In a videotex service one can distinguish between the following basic elements:

a. The storage of the pictures in the videotex systems. This is outside the scope of this document.

b. The representation and structure of the display image.

c. The videotex terminal on which the display image can be represented.

This specification defines the above mentioned elements b and c.

The structure of a display image is described in an abstract manner. For this purpose two groups of basic elements are introduced. One being the GRAPHIC PRIMITIVES or DISPLAY ELEMENTS, the other being the ATTRIBUTES. The GRAPHIC PRIMITIVES are abstractions of basic actions a display device can perform, such as drawing lines. The ATTRIBUTES specify the characteristics of the PRIMITIVES on a display device, such as colour and line thickness.
Non geometric data, alphamosaic or photographic, will be handled according to the specifications for that sets.
The geometrical information (coordinates) contained in GRAPHIC PRIMITIVES and ATTRIBUTES can be subject to transformations. These transformations perform mappings between two coordinate systems, namely:

a. Normalized Device Coordinates (NDC), used to define a uniform coordinate system for all graphic devices.

b. Device Coordinates (DC), the actual coordinate system of the graphic device, representing its display space coordinates.

PRIMITIVES and ATTRIBUTES are mapped from NDC to NDC by segment transformations and from NDC to DC by display device transformations. With the two coordinate systems it is possible to handle the storage of pictures and the representation of pictures in an independent way.
The coordinates of the PRIMITIVES and ATTRIBUTES can be specified in the following two modes:

a. Direct mode, defining absolute coordinates.

b. Incremental mode, defining steps (increments) from one coordinate position to another.

The PRIMITIVES of the geometric set are independent of the current drawing point position. So for each PRIMITIVE and/or ATTRIBUTE that has a coordinate list as a parameter, the first coordinate is in Direct mode.

OUTPUT PRIMITIVES and their ATTRIBUTES may be grouped together in a segment. Segments are collections of display elements that can be manipulated and changed. Manipulation includes creation, renaming, deletion and insertion. Changing includes transforming, making a visible segment invisible, and highlighting a segment.

Two groups of ATTRIBUTES apply to the appearance of OUTPUT PRIMITIVES. One group of ATTRIBUTES specifying geometrical aspects, the other group specifying non-geometrical aspects. The non-geometrical aspects of PRIMITIVES are controlled by the primitive attributes in one of two ways. Either by specifying a single value for an ATTRIBUTE (called INDIVIDUAL specification) or by an index into a set of values (called BUNDLED specification). A set of values or representations is called a BUNDLE TABLE.

The PRIMITIVES and ATTRIBUTES are grouped in graphic Classes, Class C0 being no geometric set. The other Classes, numbered C1 to C4, are upwards compatible, where each lower numbered Class is a subset of the higher numbered Class. No additions will be made in Class C1, C2 or C3. New PRIMITIVES and ATTRIBUTES will be added in Class C4 or in higher Classes.

4.1 The graphic primitives

The following set of GRAPHIC PRIMITIVES or DISPLAY ELEMENTS are defined:

a. POLYLINE
Defining a set of connected lines by a sequence of coordinates.

b. FILL AREA
Defining a set of connected lines with a closed boundary that may be filled.

c. POLYMARKER
Defining a set of coordinates to be indicated by a marker.

d. CELL ARRAY
Defining a rectangular grid of equal size rectangular cells with individual colours.

e. TEXT
Defining a character string and the position of that string.

f. INFILL
Defining a coordinate from which the surrounding area will be filled with an interior style, until a boundary is reached.

g. GENERALIZED DRAWING PRIMITIVE (GDP)
Defining special geometric curves. It contains a set of sub-primitives.

h. SET
Defining characteristics that apply to the output primitives. It contains a set of sub-primitives.

i. CONTROL
Defining characteristics of the device space. It contains a set of sub-primitives.

j. GRAPHIC OBJECT
Defining operations on segments. It contains a set of sub-primitives.

The primitives: POLYLINE, FILL AREA, POLYMARKER, CELL ARRAY, TEXT and INFILL directly control the output to the display device. The GENERALIZED DRAWING PRIMITIVE, with its sub-primitives, directly control the drawing of special

geometric curves on the display surface.
The primitive SET, with its sub-primitives, control the attributes as used in the output generating primitives.
The primitive CONTROL, with its sub-primitives, allows for the managment of the device space characteristics.
With the GRAPHIC OBJECT primitive the segments can be handled.

4.1.1 Sub-primitives of GDP

The GENERALIZED DRAWING PRIMITIVE has the following sub-primitives:

a. ARC
Defining an open arc through the three given coordinates. The parameters may indicate multiple open arcs.

b. ARC-CHORD
Defining an arc closed by a chord through the three given coordinates. The parameters may indicate multiple arcs. The closed area may be filled.

c. ARC-PIE
Defining an arc closed by its pie sector defined by the arc centre through the three given coordinates. The parameters may indicate multiple arcs. The closed area may be filled.

d. CIRCLE
Defining a circle through the two given coordinates. The parameters may indicate multiple circles. The circle may be filled.

e. RECTANGLE
Defining a rectangle, based on the two given coordinates. The parameters may indicate multiple rectangles. The rectangle may be filled.

f. SPLINE
Defining an uniform quadratic B-spline through a given set of coordinates.

4.1.2 Sub-primitives of SET

The SET-primitive has the following sub-primitives:

1. POLYLINE COLOUR
Defining the INDIVIDUAL aspect of the colour to be used for the following connected lines.

2. LINE WIDTH
Defining the INDIVIDUAL aspect of the width of the line to be drawn.

3. LINE TYPE
Defining the INDIVIDUAL aspect of the type of the line to be drawn.

4. POLYLINE INDEX
Defining the entry in the polyline bundle table to be used in drawing polylines, used for BUNDLED aspects.

5. FILL AREA COLOUR
Defining the INDIVIDUAL aspect of the colour to be used in filling closed boundaries.

6. FILL AREA INTERIOR STYLE
Defining the INDIVIDUAL aspect of the interior style of a closed boundary.

7. FILL AREA STYLE INDEX
Defining the INDIVIDUAL aspect of the type of hatching or pattern used in filling a closed boundary.

8. FILL AREA PATTERN
Defining the INDIVIDUAL aspect for pattern filling a closed boundary.

9. PATTERN SIZE
Defining the size of the pattern to be used in filling closed boundaries, used in INDIVIDUAL as well as BUNDLED aspects.

10. PATTERN REFERENCE POINT
Defining the lower left corner of a pattern, used in INDIVIDUAL as well as BUNDLED aspects.

11. FILL AREA INDEX
Defining the entry in the fill area bundle table to be used in filling closed boundaries, used for BUNDLED aspects.

12. POLYMARKER COLOUR
Defining the INDIVIDUAL aspect of the colour of the markers.

13. MARKER TYPE
Defining the INDIVIDUAL aspect of the type of marker to be used in the POLYMARKER output primitive.

14. MARKER SIZE
Defining the INDIVIDUAL aspect of the size of the marker to be used in the POLYMARKER output primitive.

15. POLYMARKER INDEX
Defining the entry in the polymarker bundle table to be used in the POLYMARKER output primitive, used for BUNDLED aspects.

16. TEXT COLOUR
Defining the INDIVIDUAL aspect of the colour of the string generated with the TEXT primitive.

17. CHARACTER FORMAT
Defining the INDIVIDUAL aspect of the height and width of the characters to be generated in the TEXT-primitive.

18. CHARACTER UP VECTOR
Defining the direction in which the characters of a character-string, as specified in the TEXT-primitive, is to be generated.

19. TEXT PATH
Defining the writing direction of the string as defined in the TEXT-primitive.

20. CHARACTER SPACING
Defining the INDIVIDUAL aspect of how much additional space is to be inserted between characters as defined in the TEXT-primitive.

21. TEXT FONT AND PRECISION
Defining the INDIVIDUAL aspect of the font and precision to be used in generating strings with the TEXT-primitive.

22. TEXT ALIGNMENT
Defining the INDIVIDUAL positioning of the text string in relation to the text position.

23. TEXT INDEX
Defining the entry in the text bundle table to be used in generating strings, used for BUNDLED aspects.

24. POLYLINE ASF
Defining which of the polyline bundle attributes are used INDIVIDUAL or BUNDLED.

25. FILL AREA ASF
Defining which of the fill area bundle attributes are used INDIVIDUAL or BUNDLED.

26. POLYMARKER ASF
Defining which of the polymarker bundle attributes are used INDIVIDUAL or BUNDLED.

27. TEXT ASF
Defining which of the text bundle attributes are used INDIVIDUAL or BUNDLED.

28. POLYLINE REPRESENTATION
Defining the representation to be loaded in the specified entry of the polyline bundle table.

29. FILL AREA REPRESENTATION
Defining the representation to be loaded in the specified entry of the fill area bundle table.

30. POLYMARKER REPRESENTATION
Defining the representation to be loaded in the specified entry of the polymarker bundle table.

31. TEXT REPRESENTATION
Defining the representation to be loaded in the specified entry of the text bundle table.

32. PATTERN REPRESENTATION
Defining the representation to be loaded in the specified entry of the pattern table.

4.1.3 Sub-primitives of CONTROL

The CONTROL primitive has the following sub-primitives:

a. CLEAR
Defining the clearing of the display surface, attributes and segments and whether clipping is to be used or not.

b. WINDOW
Defining the rectangular part of the virtual space to be used.

c. VIEWPORT
Defining the rectangular region of device space to be used.

d. DOMAIN RING
Defining the accuracy that will be used in coordinate data and new ring sizes, used in incremental mode coordinates.

e. WAIT
Defining a wait time before the drawing of the next elements will continue.

f. DISPLAY PLANE
Defining which display plane will be used to write to and which one will be used to be displayed.

4.1.4 Sub-primitives of GRAPHIC OBJECT

The GRAPHIC OBJECT-primitive has the following sub-primitives:

a. CREATE
Defining the beginning of a named collection of graphic elements in a segment.

b. CLOSE
Defining the end of the collection of graphic elements in a segment.

c. RENAME
Defining a new name for a named segment.

d. DELETE
Defining the deletion of a previously created segment.

e. INSERT
Defining the copying of the graphic elements in the named segment and applying the specified transformation.

f. TRANSFORMATION
Defining the transformation to be used. In this geometric set restricted to segments.

g. HIGHLIGHT
Defining new visible attributes for a collection of graphic elements. In this geometric set restricted to segments.

h. VISIBILITY
Defining visibility or invisibility of a collection of graphic elements. In this geometric set restricted to segments.

4.2 The output primitives

The actual generation of display images is controlled by a group of PRIMITIVES. These group, called OUTPUT PRIMITIVES, contains: POLYLINE, FILL AREA, POLYMARKER, CELL ARRAY, TEXT, INFILL and GDP. Each of these output primitives has a set of ATTRIBUTES that control the appearance of the resulting drawing.

Some attributes of the OUTPUT PRIMITIVES can be specified in bundle tables. These tables contain a fixed number of entries. Each entry contains a value for the attributes in that table. The following tables are defined:

a. POLYLINE BUNDLE TABLE
Containing attributes used by the POLYLINE output primitive and the GDP sub-primitives: GDP-ARC and GDP-SPLINE, e.g. the GDP sub-primitives that do not generate closed boundaries.

b. FILL AREA BUNDLE TABLE
Containing attributes used by the output primitives that create closed boundaries: FILL AREA, INFILL, GDP-ARC CHORD, GDP-ARC PIE, GDP-CIRCLE and GDP-RECTANGLE.

c. POLYMARKER BUNDLE TABLE
Containing attributes used in the POLYMARKER output primitive.

d. TEXT BUNDLE TABLE
Containing attributes used in the TEXT output primitive.

e. PATTERN TABLE
This table contains pattern definitions, used when the fill area interior style is Pattern.

Attributes can either be selected from a bundle table (BUNDLED) or from individual specified values (INDIVIDUAL). This selection is made via the Aspect Source Flags (ASF). For each attribute in a bundle table an ASF exists.

The following paragraphs will describe the ATTRIBUTES and the bundle tables used for each of the OUTPUT PRIMITIVES.

4.2.1 POLYLINE attributes

The polyline bundle table contains three attributes per entry:

1. Polyline colour

2. Line width

3. Line type

The corresponding ASFs are: 'polyline colour' ASF, 'line width' ASF and 'line type' ASF.

The following ATTRIBUTES control the drawing of a set of connected lines:

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

- LINE WIDTH
  Determines the width of the line to be used. It is controlled with SET-LINE WIDTH and used if the 'line width' ASF is INDIVIDUAL.

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

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

- ASF
  Determines which of the attributes will be used from the polyline bundle table or from the individual specification. The ASFs are controlled with SET-POLYLINE ASF.

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

4.2.2 FILL AREA attributes

The fill area bundle table contains three attributes per entry:

1. Fill area colour

2. Interior style

3. Fill area style index

The corresponding ASFs are: 'fill area colour' ASF, 'fill area interior style' ASF and 'fill area style index' ASF.

The following ATTRIBUTES control the drawing of a set of connected lines with a closed boundary:

- FILL AREA COLOUR
  Determines the COLOUR in which the lines will be drawn. It is controlled with SET-FILL AREA COLOUR and used if the 'fill area colour' ASF is INDIVIDUAL.

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

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

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

- ASF
  Determines which of the attributes will be used from the fill area bundle table or from the individual specification. The ASFs are controlled with SET-FILL AREA ASF.

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

The patterns used in filling closed boundaries when the INTERIOR STYLE = Pattern may be either INDIVIDUAL or BUNDLED. The pattern table contains the following attributes:

1. Pattern dimension

2. Colour

The use of the attributes from the pattern table is controlled with the 'fill area interior style' and 'fill area style index' ASFs.

The following ATTRIBUTES control the specification of patterns:

- DIMENSION
  Defines the number of cells of the pattern in horizontal and vertical direction. The DIMENSION is defined with SET-FILL AREA PATTERN and used when the INTERIOR STYLE = Pattern and the 'fill area style index' ASF is INDIVIDUAL.

- COLOUR
  Determines a colour value for each of the defined cells. the COLOURS are defined with SET-FILL AREA PATTERN and used when the INTERIOR STYLE = Pattern and the 'fill area style index' ASF is INDIVIDUAL.

- PATTERN SIZE
  Determines the size of the rectangle in which the pattern cells are defined. The PATTERN SIZE is defined with SET-PATTERN SIZE and used when the selected (either via the fill area bundle or individually) INTERIOR STYLE = Pattern.

- REFERENCE POINT
  Determines the position of the lower left corner of the pattern rectangle. The REFERENCE POINT is defined with SET-PATTERN REFERENCE POINT and used when the selected (either via the fill area bundle or individually) INTERIOR STYLE = Pattern.

- REPRESENTATION
  Determines the attribute values to be loaded in the specified entry of the pattern table. The REPRESENTATION is specified with SET-PATTERN REPRESENTATION and contains the attributes: INDEX, DIMENSION and COLOUR.

4.2.3 POLYMARKER attributes

The polymarker bundle table contains three attributes per entry:

1. Polymarker colour
2. Marker type
3. Marker size

The corresponding ASFs are: 'polymarker colour' ASF, 'marker type' ASF and 'marker size' ASF.

The following ATTRIBUTES control the drawing of the markers:

- POLYMARKER COLOUR
  Determines the COLOUR to be used in drawing the centred markers. The COLOUR is selected with SET-POLYMARKER COLOUR 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 an x. 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 size is defined with SET-MARKER SIZE and is used when the 'marker size' ASF is INDIVIDUAL.

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

- ASF
  Determines which of the attributes will be used from the polymarker bundle table or from the individual specification. The ASFs are controlled with SET-POLYMARKER ASF.

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

In Class C1 the only available marker is a dot. The colour than used is the colour as selected with SET-POLYLINE COLOUR.

4.2.4 CELL ARRAY attributes

The attributes are not specified via a table. The CELL ARRAY output primitive contains the specification of its own attributes, except for the colour table, which is selected with SET-FILL AREA COLOUR.

The following ATTRIBUTES control the drawing of rectangular cells:

- PATTERN
  Determines the PATTERN for a rectangular area. The PATTERN is defined by a rectangular area, sub-divided in m by n evenly spaced cells. The PATTERN is specified with CELL ARRAY.

- COLOUR
  Determines the COLOUR of each cell as defined with PATTERN. For each cell a COLOUR is specified. CELL ARRAY specifies the COLOURS.

4.2.5 TEXT attributes

The text bundle table contains four attributes per entry:

1. Text colour
2. Character height and width
3. Character spacing
4. Text font and precision

The corresponding ASFs are: 'text colour' ASF, 'character height and width' ASF, 'character spacing' ASF and 'text font and precision' ASF.

The following ATTRIBUTES control the generation of text strings:

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

- CHARACTER HEIGHT
  Determines the height of characters to be generated. It specifies the nominal height of a capital letter character (see figure 4-1). SET-CHARACTER FORMAT specifies the CHARACTER HEIGHT and is used if the 'character height and width' ASF is INDIVIDUAL.

- CHARACTER WIDTH
  Determines the width of characters to be generated. It specifies the width of the capital letter X. If a text font is used with variable width characters, the new width of the other characters is calculated from the width expansion factor. The width expansion factor is the requested width divided by the width of the capital letter X. SET-CHARACTER FORMAT specifies the CHARACTER WIDTH and is used if the 'character height and width' ASF is INDIVIDUAL.

- CHARACTER UP VECTOR
  Determines the up direction of a character. The CHARACTER UP VECTOR is defined with SET-CHARACTER UP VECTOR.

- TEXT PATH
  Determines the writing direction, with respect to the CHARACTER UP VECTOR, of a text string: Right, Left, Up or Down. The TEXT PATH is selected with SET-TEXT PATH.

- 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 and is specified with SET-CHARACTER SPACING and is used if the 'character spacing' ASF is INDIVIDUAL.

- TEXT FONT
  Determines the TEXT FONT to be used in generating text strings. Each display device should support at least one TEXT FONT with value 0. SET-TEXT FONT AND PRECISION selects the TEXT FONT 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, Char 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.

- HORIZONTAL ALIGNMENT
  Determines the horizontal positioning of the text string in relation to the text position: Normal, Left, Centre or Right (see figure 4-1). 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 (see figure 4-1). The VERTICAL ALIGNMENT is specified with SET-TEXT ALIGNMENT.

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

- **ASF**
  Determines which of the attributes will be used from the text bundle table or from the individual specification. The ASFs are controlled with SET-TEXT ASF.

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

HORIZONTAL and VERTICAL ALIGNMENT 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 ALLIGNMENT
Right	(Left, Base)
Left	(Right, Base)
Up	(Centre, Base)
Down	(Centre, Top)

The characters in the text string can be defined in a 7-bit or 8-bit environment. This environment is determined in the alphamosaic set. The characters may be taken from the invoked G-set (not from columns 0 and 1) in a 7-bit environment, or from both invoked G-sets (not from columns 0, 1, 8 and 9) in an 8-bit environment.

Besides the characters from the in-use G-set, in a 7-bit environment, some control characters may be used as well, see table 4-1 (ESC= 1/11).

Abbreviation	Name	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

Table 4-1

Permitted control characters in a 7-bit environment

In an 8-bit environment some control characters may be used as well, see table 4-2 (ESC= 1/11).

Abbreviation	Name	Coding
LS0	Locking Shift 0	0/15
LS1	Loching 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

Table 4-2

Permitted control characters in an 8-bit environment

The contents of the character repertoire and the invocation and designation of the sets is fully described in Part 1.

Characters in the text string not from the invoked G-set or not from table 4-1, in a 7-bit environment, will be ignored. In an 8-bit environment characters not from the invoked G-sets or not from table 4-2 will be ignored.

The characters defined in a particular text font are display device dependent. In order to be able to describe the use of characters in the geometric set, the following general description is used.
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 cap line and a centre line, see figure 4-1.

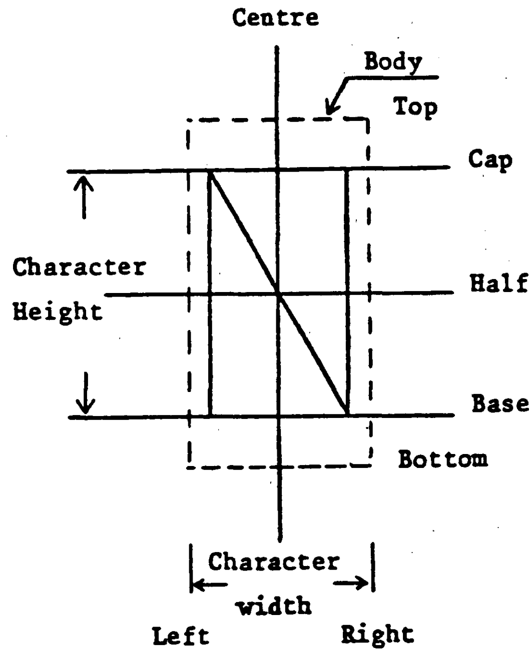


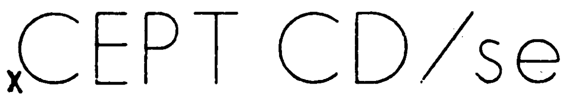
Figure 4-1

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 axes of the font coordinate system. The font baseline and the capline must be parallel to the x-axis of the font coordinate system and within the vertical extent of the body. The centre line is parallel to the y-axis 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 cap line. 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.

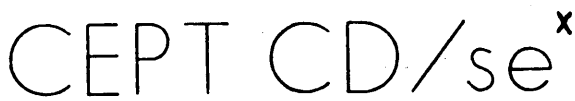
In general, the top limits of the bodies for a font will be identical with or very close to the typographical capline or ascender line and the bottom limit to the descender line. However, these and other details are purely for the use of the font designer. The intention is only that characters placed 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 4-2 and 4-3 are only inserted for clarification purposes. They show the effects of the different attributes on the display of the text: "CEPT CD/se". Changed attributes are underlined, coordinate values are expressed in BGU's based on an accuracy value of 1, 4 byte coordinate data.



Character height = 60; Character width = 50; Character up vector = (0,1);
Text = Right; Character spacing = 0; Text alignment = (Normal, Normal)

Character height = 40; Character width = 35; Character up vector = (0,1);
Text path = Right; Character spacing = 0; Text alignment = (Normal, Normal)



Character height = 60; Character width = 50; Character up vector = (0,1);
Text path = Right; Character spacing = 0; Text alignment = (Right, Top)

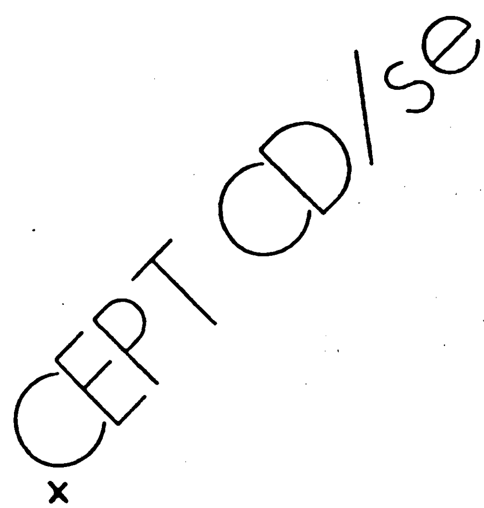
Character height = 60
Character width = 50
Character up vector = (-1,0)
Text path = Right
Character spacing = 0
Text alignment = (Normal, Normal)

Character height = 60
Character width = 50
Character up vector = (0,1)
Text path = Down
Character spacing = 0
Text alignment = (Normal, Normal)

X = text position

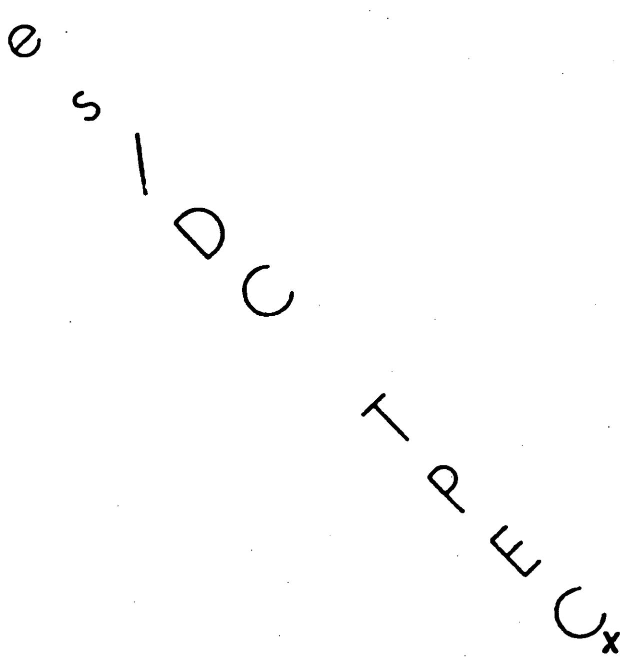
Figure 4-2

Effects of changes in individual text attributes



Character height = 80
Character width = 60
Character up vector = (-1,1)
Text path = Right
Character spacing = 0
Text alignment = (Normal, Normal)

Character height = 60
Character width = 50
Character up vector = (0,1)
Text path = Left
Character spacing = -10
Text alignment = (Normal, Normal)



Character height = 50
Character width = 40
Character up vector = (-4,3)
Text path = Up
Character spacing = 15
Text alignment = (Centre, Bottom)

Figure 4-3

Effects of combined changes in text attributes

4.2.6 INFILL attributes

The INFILL output primitive uses the ATTRIBUTES of fill area. For a description see FILL AREA attributes (4.2.2).

4.2.7 GDP attributes

The GDP sub-primitives that generate closed boundaries: GDP-ARC CHORD, GDP-ARC PIE, GDP-CIRCLE and GDP-RECTANGLE use the ATTRIBUTES of fill area. These are described in 4.2.2 under the FILL AREA attributes.

The GDP sub-primitives that do not generate closed boundaries: GDP-ARC and GDP-SPLINE, use the attributes of POLYLINE, for a description see 4.2.1.

4.3 Display device characteristics

A display device does not need to be able to support all defined Classes. If a display device supports a certain Class, it should support all primitives in that Class. Some display devices that support a certain Class may be able to handle primitives of a higher Class. In this case that display device can select the particular primitives. The encoding of primitives is independent of the Classes.

The following paragraphs describe the clearing of attributes and the attributes of two special primitives that control display device characteristics: CONTROL-WAIT and CONTROL-DISPLAY PLANE.

4.3.1 Display device clear attributes

The following ATTRIBUTES allow for the clearing of the display device characteristics:

- CLEAR DISPLAY SPACE
  Specifies whether the display space must be cleared or not. It needs only to clear the geometric picture. CLEAR DISPLAY SPACE is specified with CONTROL-CLEAR.

- CLEAR ATTRIBUTES
  Specifies whether the geometric attributes must be reset to their default values or not. CLEAR ATTRIBUTES is specified with CONTROL-CLEAR.

- CLEAR SEGMENTS
  Specifies whether all segments, their names and contents, must be deleted or not. CLEAR SEGMENTS is specified with CONTROL-CLEAR.

The geometric CLEAR should not interfere with the other sets, e.g. alphamosaic and photographic. When leaving or entering the geometric set no implicit clearing or resetting to default values is performed.

4.3.2 WAIT attributes

The only ATTRIBUTE is:

- WAIT TIME
  Determines the WAIT TIME before the drawing of the next part of a picture. The WAIT TIME is defined with CONTROL-WAIT.

4.3.3 DISPLAY PLANE attributes

The following ATTRIBUTES are available to select display planes:

- DISPLAY PLANE TO WRITE TO
  Determines the display plane to which the next PRIMITIVES will be applied. It does not need to be the plane that is currently selected to display the new created drawings. CONTROL-DISPLAY PLANE controls the selection.

- DISPLAY PLANE TO DISPLAY
  Determines the display plane that is made visible. It does not need to be the plane to which the next PRIMITIVES will be applied. CONTROL-DISPLAY PLANE controls the selection.

4.4 Coordinates, window and viewport

The geometric set uses two coordinate systems:

a. Normalized Device Coordinates (NDC), used to define a uniform coordinate system for all graphic devices.

b. Device Coordinates (DC), the actual coordinate system of the graphic device, representing the display space coordinates.

All coordinates of geometric pictures, stored in a videotex system, are in NDC, in the range from 0 (inclusive) up to 1 (non-inclusive) in x- and y-direction. The accuracy of coordinate data can be specified in three steps. This allows different pictures to be encoded with different accuracy of coordinate data and even allowing different accuracy of coordinate data in one picture. The mapping from NDC to DC is done in the display device. It possibly performs a translation and an equal scaling.

The geometric set allows for the selection of a specific rectangular part of the NDC space (window). The window rectangle is parallel to the NDC coordinate axes. The default window is the full NDC range.

A specific rectangular area of the display surface can be selected (viewport). This area, specified in NDC, defines a rectangle in DC where the actual picture will be drawn. The viewport rectangle is parallel to the DC coordinate axes. The default viewport is the full display space.

The drawing of parts of a picture outside a clipping rectangle can be removed by using clipping.

Some display devices may not be able to display the full NDC range, e.g. raster scan devices often have a 4:3 aspect ratio. For this type of devices the window will have a default size of 0 (inclusive) to 1 (non-inclusive) in the horizontal direction. In the vertical direction the window size is 0 (inclusive) to 0.75 (non-inclusive). If the aspect ratio of a certain display device is not 1:1, each device will perform automatic clipping at the window boundaries as derived from the aspect ratio. This automatic window clipping cannot be controlled.

The coordinates can be specified in two different modes:

a. Direct mode, defining absolute coordinates in NDC

b. Incremental mode, defining steps (increments) from one coordinate position to another.

The accuracy of Direct mode coordinates as well as the characteristics of the Incremental mode can be controlled.

The following paragraphs will describe the ATTRIBUTES that control the coordinates, the window, viewport and clipping rectangle.

4.4.1 Coordinate attributes

The following ATTRIBUTES control the coordinate data:

- ACCURACY VALUE
  Determines the number of bytes and so the number of bits used in encoding the Direct mode coordinate data. The ACCURACY VALUE may be 0, 1 or 2 specifying 3, 4 or 5 bytes coordinate data or an accuracy of 1/256, 1/2048 respectively 1/16384. The ACCURACY VALUE is defined with CONTROL-DOMAIN RING.

- RING SIZE
  Determines the size of the basic Ring as used in Incremental mode coordinate data. The RING SIZE is defined with CONTROL-DOMAIN RING.

- ANGULAR RESOLUTION FACTOR
  Determines the number of points on a Ring. The ANGULAR RESOLUTION FACTOR may be 0, 1, 2 or 3. CONTROL-DOMAIN RING specifies the ANGULAR RESOLUTION FACTOR.

- TRANSPARENCY
  Determines whether the Incremental mode coordinate data is packed in 8-bit or in 6-bit. The TRANSPARENCY is specified in the first byte of the Incremental mode coordinate data.

A detailed description of the coordinate data is given in chapter 6.

4.4.2 Window, viewport and clipping rectangle attributes

The following ATTRIBUTES control the window, viewport and clipping rectangle:

- WINDOW
  Determines a rectangular area of the virtual space that will be displayed. The WINDOW is defined with CONTROL-WINDOW.

- VIEWPORT
  Determines a rectangular area of the display surface, where the display image will be drawn. The VIEWPORT is defined with CONTROL-VIEWPORT.

- CLIPPING RECTANGLE
  Determines a rectangular area of the display surface which may be used to clip the display image if the clipping is on. The CLIPPING RECTANGLE is defined with CONTROL-CLIPPING RECTANGLE.
  The CLIPPING RECTANGLE cannot be transformed.

- CLIPPING
  Specifies whether CLIPPING, removing parts of display elements that lie outside the defined clipping rectangle, is applicable or not. CONTROL-CLEAR specifies the CLIPPING.

The transformation involved in mapping the window into the viewport ensures equal scaling on each axis, by:

a. Preserving the aspect ratio.

b. Mapping the lower left-hand corner of the window to the lower left-hand corner of the viewport.

So, space will be left unused at the top or right side of the viewport, if the aspect ratios of the window and viewport are different.

The automatic window clipping in display devices with a display surface with an aspect ratio that is not 1:1, cannot be controlled.

4.5 Segments

OUTPUT PRIMITIVES may be grouped in segments as well as being handled outside segments. Each segment is identified with a unique name, in the range 0 to 15 or in the range 0 to 1023. Segments can be:

a. Transformed

b. Made Visible or Invisible

c. Highlighted or not

d. Deleted

e. Renamed

f. Inserted

g. Nested

A segment starts after the GRAPHIC OBJECT-CREATE. All following output primitives belong to that segment until the first GRAPHIC OBJECT-CLOSE. All non output primitives are not stored in the segment but immediately applied. Primitives in a segment cannot be modified nor can they be added to or deleted from a segment. However, geometrical transformations, changes of the attributes used from within a segment are possible.

Each segment is stored in the display device. It can be deleted by GRAPHIC OBJECT-DELETE. It can be overwritten by a GRAPHIC OBJECT-CREATE with the same name. All segments in the display device can be deleted, their names and contents, with the CLEAR SEGMENTS attribute of CONTROL-CLEAR.

A segment may contain a GRAPHIC OBJECT-INSERT. But this primitive may not contain a TRANSFORMATION MATRIX attribute. If this, nested, segment needs to be transformed, a TRANSFORMATION MATRIX for this particular named segment must be defined with GRAPHIC OBJECT-TRANSFORMATION.

4.5.1 Segment attributes

The following ATTRIBUTES are possible in the manipulation of segments:

- TRANSFORMATION
  Determines, when a segment is inserted, whether a segment should be transformed or not. TRANSFORMATION includes: Translation, Rotation, Scaling or any combination of these basic transformations. GRAPHIC OBJECT-INSERT specifies the TRANSFORMATION.

- TRANSFORMATION MATRIX
  Specifies a TRANSFORMATION MATRIX to be applied to all primitives in a named segment, only when the segment is inserted. A TRANSFORMATION MATRIX can be specified with: GRAPHIC OBJECT-TRANSFORMATION or GRAPHIC OBJECT-INSERT.

- RESET MATRIX
  Specifies whether the TRANSFORMATION MATRIX for a named segment should be reset to its default, identity, value or not. RESET MATRIX is specified with GRAPHIC OBJECT-TRANSFORMATION.

- HIGHLIGHT
  Determines whether a named segment should be highlighted or displayed normal. HIGHLIGHT is not applied to nested segments, only to the named segment. HIGHLIGHT is specified with GRAPHIC OBJECT-HIGHLIGHT.

- VISIBILITY
  Determines whether a named segment is displayed or not.. VISIBILITY is not applied to nested segments, only to the named segment. VISIBILITY can be specified with: GRAPHIC OBJECT-VISIBILITY or GRAPHIC OBJECT-CREATE.

Transformation of segments takes place before any clipping. The clipping, if it is on, is done against the clipping rectangle as defined when the segment is displayed. So all clipping rectangles in an inserted segment are ignored. Segment transformations are not concatenated, e.g. a transformation is always applied to the named segment as it was created.

Transformation is done on the coordinates of the PRIMITIVES, Direct and Incremental-mode coordinates. Especially scaling of CIRCLES and ARCS can have different effects depending on the scaling and the specification of the coordinates of the CIRCLE or ARC.

If a segment contains a GRAPHIC OBJECT-INSERT, this primitive may not have a TRANSFORMATION MATRIX attribute, if it has a TRANSFORMATION MATRIX attribute it is ignored. A transformation of the nested segment can be specified with GRAPHIC OBJECT-TRANSFORMATION. The TRANSFORMATION MATRIX of the outer segment is not applied to the nested segment.

A TRANSFORMATION MATRIX defined, for a specific segment, with GRAPHIC OBJECT-TRANSFORMATION is not immediately applied. The matrix is stored in the display device and applied when the segment is inserted. Only one transformation is applied to a segment at any one time. The ordering between the two methods of using TRANSFORMATION MATRICES is:

1. If with GRAPHIC OBJECT-INSERT a matrix is defined, this will be applied.

2. If no matrix is defined with GRAPHIC OBJECT-INSERT, the matrix as defined for that segment with GRAPHIC OBJECT-TRANSFORMATION is applied.

3. If no matrix is defined with GRAPHIC OBJECT-INSERT and the matrix defined for that segment contains the default, identity, matrix, the segment is not transformed.

4.6 Error handling

Each PRIMITIVE with its ATTRIBUTES and possible coordinate data will appear in a specified format. The possible forms of each PRIMITIVE are defined in this specification. The following rules will apply if a display device detects something wrong in the geometric data:

- If a value is requested that is not defined, the default value will be used, e.g. an accuracy value of 3 will default to a value of 1.

- If a PRIMITIVE is received that is not supported, the whole primitive will be ignored.

- A PRIMITIVE that is not defined will be ignored.

- A PRIMITIVE that contains ATTRIBUTES and/or coordinate data not according to this specification, will be ignored.

The encoding of PRIMITIVES and ATTRIBUTES is set up in such a way that the skipping of erroneous PRIMITIVES is always possible. New PRIMITIVES can easily be detected. Each PRIMITIVE or sub-PRIMITIVE byte has a 0 in bit b7, while each ATTRIBUTE or coordinate byte will have a 1 in bit b7. The only exceptions to these rules are:

1. The characters in the text string of the TEXT-primitive are taken from the invoked G-set or from table 4-1 (in a 7-bit environment) or the invoked G-sets or from table 4-2 (in an 8-bit environment).

2. The DELETE character (7/15) identifies the end of a string, as defined in the TEXT-primitive. So a Class C1 display device should be able to detect this character, if it wants to process some primitives from a higher Class.

3. The end of Incremental mode coordinate data is identified by the <End of Block> code as defined in the Huffman Code table (table 6-4). In the 6-bit version, the normal rules apply.

5.0 OVERVIEW OF THE GEOMETRIC SET

5.1 Description of the primitives

This section of the specification will contain a detailed description of all PRIMITIVES, their sub-primitives and attributes if applicable. Each PRIMITIVE will be described in a separate paragraph. These paragraphs all have the same basic format:

The paragraph name is the <PRIMITIVE-name>

Class: The class of the primitive.

Parameters: The parameters/attributes of this primitive, containing the name, the coordinate system to be used and the type and range.

Effect: A description of the effect of this primitive.

A list of the defined classes and the distribution of primitives over the classes is given in 5.2.

5.1.1 POLYLINE-primitive

Class: C1

Parameters: - Coordinate list in NDC; number of coordinates >= 2

Effect: A sequence of connected straight lines is generated from the first point and ending at the last point.
The lines are drawn according to the current setting of attributes as specified with the SET-primitive.

5.1.2 FILL AREA-primitive

Class: C2

Parameters: - Coordinate list in NDC; number of coordinates >=3

Effect: A sequence of connected straight lines is generated from the first point to the second, to the third and so on to the last point. After the last point a line is generated to the first.
The drawing of the lines, the filling of the closed boundary etc. is controlled by the SET-primitive.

In Class C1 this primitive is ignored.

5.1.3 POLYMARKER-primitive

Class: C1/C2

Parameters: - Coordinate list in NDC; one or more coordinates

Effect: A sequence of markers is drawn centred at each coordinate in the <coordinate list>. If the marker is completely within the clipping area, the whole marker is drawn. If a part of the marker is outside the clipping area, the marker is drawn only if the marker position is inside the clipping area.

The default of this primitive in the Class C1 is the drawing of a point. The colour to be used is than according to the colour specified with the SET-POLYLINE COLOUR-primitive.

5.1.4 CELL ARRAY-primitive

Class: C2

Parameters:
- Rectangular area defined in NDC by xmin, xmax, ymin, ymax
- Dimension of area, m by n evenly spaced cells
- Entry number in CLUT; (0,1,...,7)

Effect: A rectangle, aligned with NDC-axes, defined by the points (xmin, ymin) and (xmax, ymax) is subdivided in m by n evenly spaced cells. The width of each cell is: (xmax-xmin)/m and the height is: (ymax-ymin)/n. Each cell will have the colour as specified by the corresponding <entry number in CLUT> parameter. For each cell there is one <entry number in CLUT> parameter. The total number of <entry number in CLUT> parameters is m*n. The <entry number in CLUT> values are mapped row by row to the cells (each row running from xmin to xmax). The rows are ordered from ymax to ymin.

The colour table (CLUT) used is the last selected colour table, e.g. selected with the last SET-FILL AREA COLOUR-primitive.

The default <entry number in CLUT> is 7 (white).

The rectangular grid is subject to all transformations, potentially transforming the rectangular cells into parallelograms. If part of a transformed cell is outside the clipping rectangle, the transformed cell is partially clipped. Mapping the transformed cells onto pixels of a raster display is performed according to the following rules:

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

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

In Class C1 this primitive is ignored.

5.1.5 TEXT-primitive

Class: C2

Parameters:
- Coordinate list in NDC; 1 coordinate
- Character string

Effect: The string specified by <character string> is generated at the position specified by <coordinate list>. The <character string> is drawn according to the attributes controlled by the SET-primitive.

The character codes as specified in the <character string> will be taken from the currently invoked G-set or from table 4-1 in a 7-bit environment or from both invoked G-sets or table 4-2 in an 8-bit environment.

In Class C1 this primitive is ignored.

5.1.6 INFILL-primitive

Class: C4

Parameters:
- Coordinate list in NDC; 1 coordinate

Effect: Starting at the specified coordinate, the surrounding area is filled with the interior style as specified by the SET-primitive. This filling continues until an area is found with a colour different from the colour of the starting coordinate.

In Class C1, C2 and C3 this primitive is ignored.

5.1.7 GENERALIZED DRAWING-primitive

This primitive will be described in terms of its sub-primitives.

5.1.7.1 GDP-ARC-primitive

Class: C2

Parameters: - Coordinate list in NDC; number of coordinates is a multiple of 3

Effect: An ARC is drawn from the first coordinate, through the specified intermediate coordinate, to the specified ending coordinate. If the first and last coordinate are identical, a CIRCLE is drawn through the specified coordinates and has a diameter equal to the distance from the first to the intermediate coordinate. The resulting CIRCLE is not considered to be a closed area, so it does not have an interior style. If the three coordinates are colinear, a straight LINE through the three coordinates is drawn.

The drawing of the ARC is controlled by the SET-primitive.

Each set of three coordinates indicates an ARC.

In Class C1 this primitive is ignored.

5.1.7.2 GDP-ARC CHORD-primitive

Class: C2

Parameters: - Coordinate list in NDC; number of coordinates is a multiple of 3

Effect: An ARC is drawn from the first coordinate, through the specified intermediate coordinate, to the specified ending coordinate. The chord from the first coordinate to the third coordinate is drawn, and the area defined by the ARC and the chord is filled with the interior style specified in a previously issued SET-primitive. If the first and last coordinate are identical a CIRCLE is drawn through the specified coordinates and has a diameter equal to the distance from the first to the intermediate coordinate. The CIRCLE is filled with the interior style specified in a previously issued SET-primitive. If the three coordinates are colinear, a straight LINE through the three coordinates is drawn.

The drawing of the ARC is controlled by the SET-primitive.
Each set of three coordinates indicates an ARC.

In Class C1 this primitive is ignored.

5.1.7.3 GDP-ARC PIE-primitive

Class: C2

Parameters: - Coordinate list in NDC; number of coordinates is a multiple of 3

Effect: An ARC is drawn from the first coordinate, through the specified intermediate coordinate, to the specified ending coordinate. The pie sector defined by the ARC centre, the specified starting coordinate, and the ending coordinate is drawn. The area defined by the ARC and the pie sector is filled with the interior style specified in a previously issued SET-primitive.
If the first and last coordinate are identical a CIRCLE is drawn through the specified coordinates and has a diameter equal to the distance from the first to the intermediate coordinate. The CIRCLE is filled with the interior style specified in a previously issued SET-primitive.
If the three coordinates are colinear, a straight LINE through the three coordinates is drawn.

The drawing of the ARC is controlled by the SET-primitive.
Each set of three coordinates indicates an ARC.

In Class C1 this primitive is ignored.

5.1.7.4 GDP-CIRCLE-primitive

Class: C2

Parameters: - Coordinate list in NDC; number of coordinates is a multiple of 2

Effect: A CIRCLE is drawn at the specified centre with a radius equal to the distance between the two points. The first coordinate of each pair is the centre coordinate, the second coordinate is a peripheral point. If the second coordinate of a pair is 0, a dot is drawn. The CIRCLE has an interior style as specified in a previously issued SET-primitive.

The drawing of the CIRCLE is controlled by the SET-primitive.

Each set of two coordinates indicates a CIRCLE.

In Class C1 this primitive is ignored.

5.1.7.5 GDP-RECTANGLE-primitive

Class: C2

Parameters: - Coordinate list in NDC; number of coordinates is a multiple of 2

Effect: A RECTANGLE is drawn with one corner at the first coordinate and the opposite corner at the second coordinate. The RECTANGLE has an interior style as specified in a previously issued SET-primitive.

The drawing of the RECTANGLE is controlled by the SET-primitive.

Each set of two coordinates indicates a RECTANGLE.

In Class C1 this primitive is ignored.

5.1.7.6 GDP-SPLINE-primitive

Class: C2

Parameters: - Coordinate list in NDC; number of coordinates >=3

Effect: A smooth curve is drawn based on the specified coordinates. This curve is known as a uniform quadratic B-spline.
For a short description see appendix A.

The drawing of the curve is controlled by the SET-primitive.

In Class C1 this primitive is ignored.

5.1.8 SET-primitive

This primitive will be described in terms of its sub-primitives.

5.1.8.1 SET-POLYLINE COLOUR-primitive

Class: C1

Parameters: - Colour table; CLUT1, CLUT2, CLUT3, CLUT4
- Entry number in CLUT; (0,1,...,7)

Effect: The current polyline colour is changed to the specified colour. This value is used for the display of subsequent POLYLINE output primitives, created when the 'polyline colour' ASF is INDIVIDUAL. This value does not affect the display of subsequent POLYLINE output primitives, created when the 'polyline colour' ASF is BUNDLED.
The use of redefinable colours is according to the specification in Part 5.

By default the eight colours defined in the alphamosaic C1 set are available.

The default <entry number in CLUT> is 7 (white).

5.1.8.2 SET-LINE WIDTH-primitive

Class: C1

Parameters: - Line width; ymax

Effect: The current line width is changed to the specified <line width>. This value is used for the display of subsequent POLYLINE output primitives, created when the 'line width' ASF is INDIVIDUAL. This value does not affect the display of subsequent POLYLINE output primitives, created when the 'line width' ASF is BUNDLED.

The default <line width> is equal to the Basic Grid Unit (=ymax).

5.1.8.3 SET-LINE TYPE-primitive

Class: C2

Parameters: - Line type; (0,1,2,3)

Effect: The current line type is changed to the specified line type. This value is used for the display of subsequent POLYLINE output primitives, created when the 'line type' ASF is INDIVIDUAL. This value does not affect the display of subsequent POLYLINE output primitives, created when the 'line type' ASF is BUNDLED. There are four defined line types:

Value	Line type
0	Solid
1	Dashed
2	Dotted
3	Dashed-dotted
>=4	Reserved for future use

The default <line type> is solid. In Class C1 only solid lines are supported.

The format of the "dashes" and "dots" in the <line type> is device dependent. "Dashing" in small increments may default to a solid line.

In Class C1 this primitive is ignored.

5.1.8.4 SET-POLYLINE INDEX-primitive

Class: C3

Parameters: - Polyline index; (0,1,...,7)

Effect: The current polyline index is set to the specified value. This value is used when creating subsequent POLYLINE output primitives.

The default <polyline index> is 0. If an empty entry of the polyline bundle table is specified, the default will be used. Entry 0 of the polyline bundle table contains by default the INDIVIDUAL values.

In Classes C1 and C2 this primitive is ignored.

5.1.8.5 SET-FILL AREA COLOUR-primitive

Class: C2

Parameters: - Colour table; CLUT1, CLUT2, CLUT3, CLUT4
- Entry number in CLUT; (0,1,...,7)

Effect: The current fill area colour is changed to the specified colour. This value is used in filling closed boundaries, when the 'fill area colour' ASF is INDIVIDUAL. This value does not affect the filling of closed boundaries, when the 'fill area colour' ASF is BUNDLED.
The use of redefinable colours is according to the specification in Part 5.

By default the eight colours defined in the alphamosaic C1 set are available.

The default <entry number in CLUT> is 7 (white).

In Class C1 this primitive is ignored.

5.1.8.6 SET-FILL AREA INTERIOR STYLE-primitive

Class: C2

Parameters: - Interior style; (0,1,2,3)

Effect: The current interior style is changed to the specified interior style. This value is used in filling closed boundaries, when the 'fill area interior style' ASF is INDIVIDUAL. This value does not affect the filling of closed boundaries, when the 'fill area interior style' ASF is BUNDLED.
The interior style defines how a closed boundary should be filled:

a. HOLLOW is <interior style> = 0
Only a solid closed boundary is drawn, no filling.

b. SOLID is <interior style> = 1
The closed boundary is solid filled.

c. HATCH is <interior style> = 2
The closed boundary is filled with the selected hatch style.

d. PATTERN is <interior style> = 3
The closed boundary is filled with the selected pattern.

Other values of <interior style> are reserved for future use.

The default <interior style> is HOLLOW.

In Class C1 this primitive is ignored.

5.1.8.7 SET-FILL AREA STYLE INDEX-primitive

Class: C2

Parameters: - Fill area style index; (0,1,...,5)

Effect: The current fill area style index is changed to the specified one. This value is used in filling closed boundaries, when the 'fill area style index' ASF is INDIVIDUAL. This value does not affect the filling of closed boundaries, when the 'fill area style index' ASF is BUNDLED.

For interior styles HOLLOW and SOLID, the <fill area style index> is unused. For interior style PATTERN, it is a pointer into the pattern table. For interior style HATCH, the value of <fill area style index> is according to the following:

Value	Hatch style
0	Vertical lines
1	Horizontal lines
2	45 degree lines
3	-45 degree lines
4	Crossed lines, vertical and horizontal
5	Crossed lines, 45 degrees and -45 degrees
>=6	Reserved for future use

The default <hatch style> is 0 (vertical lines).

In Class C1 this primitive is ignored.

5.1.8.8 SET-FILL AREA PATTERN-primitive

Class: C2

Parameters:
- Dimension m by n
- Entry number in CLUT; (0,1,...,7)

Effect: The current pattern dimension and cell colours are changed to the specified ones. This pattern is used when the fill area interior style is PATTERN, and the 'fill area interior style' ASF is INDUVIDUAL. This value does not affect the filling of closed boundaries, when the 'fill area interior style' ASF is BUNDLED.

The <pattern style> is defined in a rectangle, aligned with NDC-axes, by the points (xmin,ymin) and (xmax,ymax). The points are defined with the SET-PATTERN SIZE-primitive. This rectangle is subdivided in m by n evenly spaced cells. The width of each cell is: (xmax-xmin)/m and the height is: (ymax-ymin)/n. Each cell will have the colour as specified by the corresponding <entry number in CLUT> parameter. For each cell there is one <entry number in CLUT> parameter. The total number of <entry number in CLUT> parameters is m*n.
The so defined pattern is replicated row by row starting at the highest left corner of the closed boundary until the interior is covered.

The default pattern is the drawing of the closed boundary line, in the colour as specified with the SET-FILL AREA COLOUR-primitive.

The colour table (CLUT) used for the <pattern style> parameter is the <colour table> as specified with the SET-FILL AREA COLOUR-primitive.

The default <entry number in CLUT> for the <pattern style> parameter is 7 (white).

In Class C1 this primitive is ignored.

5.1.8.9 SET-PATTERN SIZE-primitive

Class: C2

Parameters:
- Pattern size with xmin, xmax, ymin, ymax; in NDC

Effect: The current pattern size is set to the specified <pattern size>. When the currently selected (either via the fill area bundle or individually, depending on the corresponding ASF) interior style is PATTERN, this value is used, where possible, in conjunction with the current pattern reference point for displaying the FILL AREA output primitives.

The default values are: <xmin> and <ymin>=0 and <xmax> and <ymax>=1, or equal to the aspect ratio of the display surface.

In Class C1 this primitive is ignored.

5.1.8.10 SET-PATTERN REFERENCE POINT-primitive

Class: C2

Parameters: - Reference point; in NDC

Effect: The current pattern reference point is set to the specified <reference point>. When the currently selected (either via the fill area bundle or individually, depending on the corresponding ASF) fill area style is PATTERN, this value is used, where possible, in conjunction with the current pattern size for displaying the FILL AREA output primitives.

The <reference point> is the lower left corner of the rectangular grid.

The default value is 0, e.g. the lower left corner of the display surface.

In Class C1 this primitive is ignored.

5.1.8.11 SET-FILL AREA INDEX-primitive

Class: C3

Parameters: - Fill area index; (0,1,...,7)

Effect: The current fill area index is set to the specified <fill area index>. This value is used when creating subsequent closed boundaries like fill areas.

The default <fill area index> is 0. If an empty entry of the fill area bundle table is specified, the default will be used. Entry 0 of the fill area bundle table contains by default the INDIVIDUAL values.

In Class C1 and C2 this primitive is ignored.

5.1.8.12 SET-POLYMARKER COLOUR-primitive

Class: C2

Parameters: - Colour table; CLUT1, CLUT2, CLUT3, CLUT4
- Entry number in CLUT; (0,1,...,7)

Effect: The current polymarker colour is changed to the specified colour. This value is used for the display of subsequent POLYMARKER output primitives, created when the 'polymarker colour' ASF is INDIVIDUAL. This value does not affect the display of subsequent

POLYMARKER output primitives, created when the 'polymarker colour' ASF is BUNDLED.

The use of redefinable colours is according to the specification in Part 5.

By default the eight colours defined in the alphamosaic C1 set are available.

The default <entry number in CLUT> is 7 (white).

In Class C1 this primitive is ignored.

5.1.8.13 SET-MARKER TYPE-primitive

Class: C2

Parameters: - Marker type; (0,1,2,3,4)

Effect: The current marker type is changed to the specified one. This value is used for the display of subsequent POLYMARKER output primitives, created when the 'marker type' ASF is INDIVIDUAL. This value does not affect the display of subsequent POLYMARKER output primitives, created when the 'marker type' ASF is BUNDLED.

The <marker type> defines the type of the marker to be used, to draw centred symbols according to the following list:

Value	Type
0	.
1	+
2	*
3	o
4	x
>=5	Reserved for future use

<marker type> = 0 is always displayed as the smallest displayable dot. The default <marker type> = 2 (*).

In Class C1 this primitive is ignored.

5.1.8.14 SET-MARKER SIZE-primitive

Class: C2

Parameters: - Marker size

Effect: The current marker size is changed to the specified one. This value is used for the display of subsequent POLYMARKER output primitives, created when the 'marker size' ASF is INDIVIDUAL. This value does not affect the display of subsequent POLYMARKER output primitives, created when the 'marker size' ASF is BUNDLED.

The <marker size> is specified in Basic Grid Units. The default <marker size> is the normal size of the marker in the alphamosaic set.

In Class C1 this primitive is ignored.

5.1.8.15 SET-POLYMARKER INDEX-primitive

Class: C3

Parameters: - Polymarker index; (0,1,...,7)

Effect: The current polymarker index is set to the specified <polymarker index>. This value is used when creating subsequent POLYMARKER output primitives.

The default <polymarker index> is 0. If an empty entry of the polymarker bundle table is specified, the default will be used. Entry 0 of the polymarker bundle table contains by default the INDIVIDUAL values.

In Class C1 and C2 this primitive is ignored.

5.1.8.16 SET-TEXT COLOUR-primitive

Class: C2

Parameters: - Colour table; CLUT1, CLUT2, CLUT3, CLUT4
- Entry number in CLUT; (0,1,...,7)

Effect: The current text colour is changed to the specified colour. This value is used for the display of subsequent TEXT output primitives, created when the 'text colour' ASF is INDIVIDUAL. This value does not affect the display of subsequent TEXT output primitives, created when the 'text colour' ASF is BUNDLED.

The use of redefinable colours is according to the specification in Part 5.

By default the eight colours defined in the alphamosaic C1 set are available.

The default <entry number in CLUT> is 7 (white).

In Class C1 this primitive is ignored.

5.1.8.17 SET-CHARACTER FORMAT-primitive

Class: C2

Parameters: - Character height in NDC
- Character width in NDC

Effect: The current character format is set to the specified value. This value is used for the display of subsequent TEXT output primitives, created when the 'character height and width' ASF is INDIVIDUAL. This value does not affect the display of subsequent TEXT output primitives, created when the 'character height and width' ASF is BUNDLED.

The <character height> defines the height of the character to be displayed with the TEXT-primitive. The <character width> defines the width. Both apply to the size of an X (capital letter x).

The default <character height> and <character width> is the normal size character in the alphamosaic set.

In Class C1 this primitive is ignored.

5.1.8.18 SET-CHARACTER UP VECTOR-primitive

Class: C2

Parameters: - Character up vector in NDC (x,y)

Effect: The current character up vector is set to the specified value. The <character up vector> is used in displaying character strings with the TEXT-primitive.

The default <character up vector> is (0,1).

In Class C1 this primitive is ignored.

5.1.8.19 SET-TEXT PATH-primitive

Class: C2

Parameters: - Path indicator; (0,1,2,3)

Effect: The current text path is set to the specified path. The text path, with respect to the character up vector, is used in displaying character strings with the TEXT-primitive.

The following <path indicators> are defined:

Value	Type
0	Right
1	Left
2	Up
3	Down
>=4	Reserved for future use

The default <path indicator> = 0 (Right).

In Class C1 this primitive is ignored.

5.1.8.20 SET-CHARACTER SPACING-primitive

Class: C2

Parameters: - Character spacing

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

The <character spacing> defines the spacing between characters in the text path direction. This space may be positive or negative.

The default <character spacing> is 0.

In Class C1 this primitive is ignored.

5.1.8.21 SET-TEXT FONT AND PRECISION-primitive

Class: C3

Parameters: - Text font
- Text precision; String, Char, Stroke (0,1,2)

Effect: The current text font and precision attributes are changed to the specified ones. This value is used for the display of subsequent TEXT output primitives, created when the 'text font and precision' ASF is INDIVIDUAL. This value does not affect the display of subsequent TEXT output primitives, created when the 'text font and precision' ASF is BUNDLED.

The <text font> parameter defines the text font to be used. The default value is 0 for <text font>. Each display device will support at least one text font, with value 0. If a non existent text font is specified, it will default to the text font with value 0.

The <text precision> parameter determines the accuracy with which the character string in the TEXT-primitive is drawn. The possible values of this parameter are:

a. String is <text precision> = 0
The character string in the TEXT-primitive is positioned by placing the first character of that string at the given starting point. Clipping is done in an implementation dependent way.

b. Char is <text precision> = 1
Individual characters of the character string in the TEXT-primitive in the requested or default text font are positioned as calculated from the starting point, the CHARACTER FORMAT, the TEXT PATH and the CHARACTER SPACING. The writing direction is defined by the CHARACTER UP VECTOR and the CHARACTER PATH. For the representation of each individual character, the attributes CHARACTER FORMAT and the up direction of the CHARACTER UP VECTOR are evaluated as closely as possible in an implementation dependent way. Clipping is done at least on a character body by character body basis.

c. Stroke is <text precision> = 2
The character string in the TEXT-primitive in the requested or default text font is displayed at the text starting point by applying all text attributes. The character string is clipped exactly at the clipping rectangle.

Other values of <text precision> are reserved for future use.

The default <text precision> = 0 (String).

In Class C1 and C2 this primitive is ignored.

5.1.8.22 SET-TEXT ALIGNMENT-primitive

Class: C2

Parameters:
- Horizontal alignment; (0,1,2,3)
- Vertical alignment; (0,1,2,3,4,5)

Effect: The current text alignment is set to the specified value. This value is used when creating subsequent TEXT output primitives.

The following <horizontal alignment> values are defined:

Value	Horizontal Alignment
0	Normal
1	Left
2	Centre
3	Right
>=4	Reserved for future use

The default <horizontal alignment> = 0 (Normal).

The following <vertical alignment> values are defined:

Value	Vertical Alignment
0	Normal
1	Top
2	Cap
3	Half
4	Base
5	Bottom
>=6	Reserved for future use

The default <vertical alignment> = 0 (Normal).

In Class C1 this primitive is ignored.

5.1.8.23 SET-TEXT INDEX-primitive

Class: C3

Parameters: - Text index; (0,1,...,7)

Effect: The current text index is set to the specified <text index>. This value is used when creating subsequent TEXT output primitives.

The default <text index> is 0. If an empty entry of the text bundle table is specified, the default will be used. Entry 0 of the text bundle table contains by default the INDIVIDUAL values.

In Class C1 and C2 this primitive is ignored.

5.1.8.24 SET-POLYLINE ASF-primitive

Class: C3

Parameters:
- Polyline colour ASF; BUNDLED or INDIVIDUAL
- Line width ASF; BUNDLED or INDIVIDUAL
- Line type ASF; BUNDLED or INDIVIDUAL

Effect: The Aspect Source Flags (ASFs) of the polyline bundle attributes are set according to the specified values.

The default value for the ASFs is INDIVIDUAL.

In Class C1 and C2 this primitive is ignored.

5.1.8.25 SET-FILL AREA ASF-primitive

Class: C3

Parameters:
- Fill area colour ASF; BUNDLED or INDIVIDUAL
- Fill area interior style ASF; BUNDLED or INDIVIDUAL
- Fill area style index ASF; BUNDLED or INDIVIDUAL

Effect: The Aspect Source Flags (ASFs) of the fill area bundle attributes are set according to the specified values.

The default value for the ASFs is INDIVIDUAL.

In Class C1 and C2 this primitive is ignored.

5.1.8.26 SET-POLYMARKER ASF-primitive

Class: C3

Parameters:
- Polymarker colour ASF; BUNDLED or INDIVIDUAL
- Marker type ASF; BUNDLED or INDIVIDUAL
- Marker size ASF; BUNDLED or INDIVIDUAL

Effect: The Aspect Source Flags (ASFs) of the polymarker bundle attributes are set according to the specified values.

The default value for the ASFs is INDIVIDUAL.

In Class C1 and C2 this primitive is ignored.

5.1.8.27 SET-TEXT ASF-primitive

Class: C3

Parameters:
- Text colour ASF; BUNDLED or INDIVIDUAL
- Character height and width ASF; BUNDLED or INDIVIDUAL
- Character spacing ASF; BUNDLED or INDIVIDUAL
- Text font and precision ASF; BUNDLED or INDIVIDUAL

Effect: The Aspect Source Flags (ASFs) of the text bundle attributes are set according to the specified values.

The default value for the ASFs is INDIVIDUAL.

In Class C1 and C2 this primitive is ignored.

5.1.8.28 SET-POLYLINE REPRESENTATION-primitive

Class: C3

Parameters:
- Polyline index; (0,1,...,7)
- Line type; (0,1,2,3)
- Polyline colour table; CLUT1, CLUT2, CLUT3, CLUT4
- Entry number in CLUT; (0,1,...,7)
- Line width; ymax

Effect: In the polyline bundle table, the <polyline index> is associated with the specified parameters.

The <line type> parameter defines the 'line type' of the specified bundle table entry, with the defined values:

Value	Line type
0	Solid
1	Dashed
2	Dotted
3	Dashed-dotted
>=4	Reserved for future use

The <polyline colour table> and <entry number in CLUT> parameters define the 'polyline colour' of the specified bundle table entry.

The <line width> parameter defines the 'line width' of the specified bundle table entry.

When polylines are displayed, the <polyline index> refers to an entry in the polyline bundle table. If polylines are displayed with a <polyline index> that is not present, <polyline index> 0 is used. Which of the aspects in the entry are used depends on the setting of the corresponding ASFs.

The default contents of entry 0 of the polyline bundle table contains the INDIVIDUAL values of the polyline attributes.

In Class C1 and C2 this primitive is ignored.

5.1.8.29 SET-FILL AREA REPRESENTATION-primitive

Class: C3

Parameters:
- Fill area index; (0,1,...,7)
- Fill area colour table; CLUT1, CLUT2, CLUT3, CLUT4
- Entry number in CLUT; (0,1,...,7)
- Interior style; (0,1,2,3)
- Fill area style index; (0,1,...,5)

Effect: In the fill area bundle table, the <fill area index> is associated with the specified parameters.

The <fill area colour table> and <entry number in CLUT>

parameters define the 'fill area colour' of the specified bundle table entry.

The <interior style> parameter defines 'fill area interior style' of the specified bundle table entry, with the defined values:

Value	Interior style
0	Hollow
1	Solid
2	Hatch
3	Pattern
>=4	Reserved for future use

The <fill area style index> parameter defines the 'fill area style index' of the specified bundle table entry.

When closed boundaries are displayed, the <fill area index> refers to an entry in the fill area bundle table. If closed boundaries are displayed with a <fill area index> that is not present, <fill area index> 0 is used. Which of the aspects in the entry are used depends on the setting of the corresponding ASFs.

The default contents of entry 0 of the fill area bundle table contains the INDIVIDUAL values of the fill area attributes.

In Class C1 and C2 this primitive is ignored.

5.1.8.30 SET-POLYMARKER REPRESENTATION-primitive

Class: C3

Parameters:
- Polymarker index; (0,1,...,7)
- Marker type; (0,1,2,3,4)
- Polymarker colour table; CLUT1, CLUT2, CLUT3, CLUT4
- Entry number in CLUT; (0,1,...,7)
- Marker size

Effect: In the polymarker bundle table, the <polymarker index> is associated with the specified parameters.

The <marker type> defines the 'marker type' of the specified bundle table entry, with the following values:

Value	Type
0	.
1	+
2	*
3	o
4	x
>=5	Reserved for future use

The <polymarker colour table> and <entry number in CLUT> parameters define the 'polymarker colour' of the specified bundle table entry.

The <marker size> defines the 'marker size' entry of the specified bundle table entry.

When polymarkers are displayed, the <polymarker index> refers to an entry in the polymarker bundle table. If polymarkers are displayed with a <polymarker index> that is not present, <polymarker index> 0 is used. Which of the aspects in the entry are used depends on the setting of the corresponding ASFs.

The default contents of entry 0 of the polymarker bundle table contains the INDIVIDUAL values of the polymarker attributes.

In Class C1 and C2 this primitive is ignored.

5.1.8.31 SET-TEXT REPRESENTATION-primitive

Class: C3

Parameters:
- Text index; (0,1,...,7)
- Text colour table; CLUT1, CLUT2, CLUT3, CLUT4
- Entry number in CLUT; (0,1,...,7)
- Character height in NDC
- Character width in NDC
- Character spacing
- Text font
- Text precision; STRING, CHAR, STROKE (0,1,2)

Effect: In the text bundle, the <text index> is associated with the specified parameters.

The <text colour table> and <entry number in CLUT> parameters define the 'text colour' of the specified bundle table entry.

The <character height> defines the height of the character to be displayed with the TEXT-primitive. The <character width> defines the width. Both apply to the size of an X (capital letter x) and define the 'character height and width' of the specified bundle table entry.

The <character spacing> defines the spacing between characters in the text path direction. This space may be positive or negative. It defines the 'character spacing' of the specified bundle table entry.

The <text font> parameter defines the 'text font' of the specified entry.

The <text precision> parameter defines the 'text precision' of the specified entry. The possible values of this parameter are:

Value	Text precision
0	String
1	Char
2	Stroke
>=3	Reserved for future use

The <text font> and <text precision> parameters define the 'text font and precision' of the specified bundle table entry.

When text is displayed, the <text index> refers to an entry in the text bundle table. If text is displayed with a <text index> that is not present, <text index> 0 is used. Which of the aspects in the entry are used depends on the setting of the corresponding ASFs.

The default contents of entry 0 of the text bundle table contains the INDIVIDUAL values of the text attributes.

In Class C1 and C2 this primitive is ignored.

5.1.8.32 SET-PATTERN REPRESENTATION-primitive

Class: C3

Parameters:
- Pattern index; (0,1,...,7)
- Dimension m by n
- Entry number in CLUT (m*n values); (0,1,...,7)

Effect: In the pattern table, the <pattern index> is associated with the specified parameters.

The <dimension> defines the 'pattern dimension' of the specified table entry. In horizontal direction m-cells and vertically n-cells.

The <entry number in CLUT> parameters (m*n), define the colour of each cell and the 'colour' values for the specified table entry.

The colour table (CLUT) used, is the <colour table> as specified with the SET-FILL AREA COLOUR-primitive.

When closed boundaries are displayed, if the currently selected interior style is PATTERN, the currently selected style index refers to an entry in the pattern table. If closed boundaries are displayed with a <pattern index> that is not present, <pattern index> 0 is used.

The default contents of entry 0 of the pattern table contains the INDIVIDUAL values of the pattern attributes.

In Class C1 and C2 this primitive is ignored.

5.1.9 CONTROL-primitive

This primitive will be described in terms of its sub-primitives.

5.1.9.1 CONTROL-CLEAR-primitive

Class: C1/C3

Parameters:
- Clear display space; Yes or No
- Clear attributes; Yes or No
- Clear segments; Yes or No
- Clipping; Clip-on or Clip-off

Effect: If <clear display space> is yes, the display space is cleared, it does not contain a display image.

The attributes as defined by the SET-primitive are reset to their default values when <clear attributes> is yes. Only attributes of the geometric set can be controlled.

The contents of all already stored segments is cleared and their names deleted when <clear segments> is yes.

The control of clipping, removing parts of display elements that lie outside the defined clipping rectangle, is done with the <clipping> parameter.

The default value for <clipping> is clip-off.

The clearing of segments in Class C1 and C2 will have no effect. The clearing of attributes that are not defined in a particular class will have no effect.

The clipping is off in Class C1.

5.1.9.2 CONTROL-WINDOW-primitive

Class: C2

Parameters:
- Window limits in NDC; xmin<xmax and ymin<ymax

Effect: The value range specifies part of the virtual space use for the display of images.

The default values are: <xmin> and <ymin>=0 <xmax> and <ymax>=1, or equal to the aspect ratio of the display surface.

In Class C1 this primitive is ignored.

5.1.9.3 CONTROL-VIEWPORT-primitive

Class: C2

Parameters: - Viewport limits in NDC; xmin<xmax and ymin<ymax

Effect: The value range specifies a rectangular region of the device space.

The default values are: <xmin> and <ymin>=0 <xmax> and <ymax>=1, or equal to the aspect ratio of the display surface.

In Class C1 this primitive is ignored.

5.1.9.4 CONTROL-CLIPPING RECTANGLE-primitive

Class: C2

Parameters: - Clipping rectangle limits in NDC; xmin<xmax and ymin<ymax

Effect: The value range specifies a rectangular region of the device space.

The default values are: <xmin> and <ymin>=0 <xmax> and <ymax>=1, or equal to the aspect ratio of the display surface.

In Class C1 this primitive is ignored.

5.1.9.5 CONTROL-DOMAIN RING-primitive

Class: C1/C3

Parameters:
- Accuracy value; (0,1,2)
- Angular resolution factor; (0,1,2,3)
- Ring size; in Basic Grid Units

Effect: The accuracy of the coordinate data is changed to the specified value. The value of the <accuracy value> is according to the following table:

Value	number of bytes for each coordinate
0	3 bytes
1	4 bytes
2	5 bytes
>=3	Reserved for future use

The default <accuracy value> is 1 (4 bytes).

The default <ring size> is 8 and the default <angular resolution factor> is 0.

The basic Ring used in the Incremental coordinate mode is changed to the given <ring size> (in Basic Grid Units), the <angular resolution factor> is changed to the specified factor.

This primitive may have a variable set of parameters:

a. Two parameters
   - <accuracy value>
   - <angular resolution factor>

b. All listed parameters

In Class C1 and C2 the parameters <angular resolution factor> and <ring size> will be ignored.

5.1.9.6 CONTROL-WAIT-primitive

Class: C2

Parameters: - Wait time; in units of 0.1 seconds

Effect: The drawing of the following graphic elements will be delayed for the given time. A second CONTROL-WAIT-primitive will cancel the previous wait time and the new one applied.

In Class C1 this primitive is ignored.

5.1.9.7 CONTROL-DISPLAY PLANE-primitive

Class: C4

Parameters: - Display plane number to write to
- Display plane number to display

Effect: Selecting the display plane to which the following graphic elements will be applied and which display plane will be visible.

The default value for both <write> and <display> is 0.

For Class C1, C2 and C3 this will default to the only available plane, so this primitive is ignored.

5.1.10 GRAPHIC OBJECT-primitive

This primitive will be described in terms of its sub-primitives.

5.1.10.1 GRAPHIC OBJECT-CREATE-primitive

Class: C3

Parameters:
- Segment name
- Visibility; Visible or Invisible

Effect: This primitive opens a segment with the name <segment name>. All following graphic primitives belong to this segment until the first GRAPHIC OBJECT-CLOSE primitive.

The <visibility> controls whether the following primitives will be displayed (Visible) or not (Invisible). It is not applied to nested segments.
The default <visibility> is Visible.

In Class C1 and C2 this primitive will be ignored.

5.1.10.2 GRAPHIC OBJECT-CLOSE-primtive

Class: C3

Parameters:
- None

Effect: No further display elements will be added to the previously open segment. If no segment was open, this primitive will have no effect.

In Class C1 and C2 this primitive will be ignored.

5.1.10.3 GRAPHIC OBJECT-RENAME-primitive

Class: C3

Parameters:
- Old segment name
- New segment name

Effect: The segment with the name <old segment name> will get the name <new segment name>. If the <old segment name> is the open segment this segment will be renamed to <new segment name>.

In Class C1 and C2 this primitive will be ignored.

5.1.10.4 GRAPHIC OBJECT-DELETE-primitive

Class: C3

Parameters: - Segment name

Effect: The segment and the <segment name> are deleted from the display device storage. If the named segment does not exist, this primitive will have no effect.

In Class C1 and C2 this primitive will be ignored.

5.1.10.5 GRAPHIC OBJECT-INSERT-primitive

Class: C3

Parameters:
- Segment name
- Transformation; Yes or No
- Transformation matrix

Effect: After transformation of the segment <segment name>, if applicable, the display elements of the segment are drawn on the display surface. If the segment <segment name> does not exist, this primitive will have no effect.

The default value for <transformation> is No.

The coordinates in the segment will be transformed by applying the following matrix multiplication to them:

$$\begin{vmatrix} X1 \\ \\ Y1 \end{vmatrix} = \begin{vmatrix} M11 & M12 & M13 \\ & & \\ M21 & M22 & M23 \end{vmatrix} * \begin{vmatrix} X \\ Y \\ 1 \end{vmatrix}$$

The original coordinates are (X,Y), the transformed coordinates are (X1, Y1) both in NDC. The values M13 en M23 are in NDC, the other values are unitless. The segment transformation (conceptually) takes place in NDC space. The segment transformation is not cumulative, i.e. it always applies to the segment as originally created. The <transformation> is not applied to nested segments.

The default value for the matrix elements M11 and M22 = 1 and for the others 0.

For geometric attributes which are vectors (e.g. CHARACTER UP VECTOR) the values M13 and M23 are ignored.

This primitive may have a variable set of parameters:

a. Two parameters
   - <segment name>
   - <transformation>

b. All listed parameters

In Class C1 and C2 this primitive will be ignored.

5.1.10.6 GRAPHIC OBJECT-TRANSFORMATION-primitive

Class: C3

Parameters:
- Segment name
- Reset matrix; Yes or No
- Transformation matrix

Effect: The <transformation matrix> is stored in the display device, marked as to be used for the segment <segment name>.

If <reset matrix> is Yes, the stored transformation matrix is reset to its default value.

When the segment with the name <segment name> is displayed, the coordinates of its primitives will be transformed by applying the following matrix multiplication to them:

$$\begin{vmatrix} X1 \\ \\ Y1 \end{vmatrix} = \begin{vmatrix} M11 & M12 & M13 \\ & & \\ M21 & M22 & M23 \end{vmatrix} * \begin{vmatrix} X \\ Y \\ 1 \end{vmatrix}$$

The original coordinates are (X,Y), the transformed coordinates are (X1, Y1) both in NDC. The values M13 en M23 are in NDC, the other values are unitless. The segment transformation (conceptually) takes place in NDC space. The segment transformation is not cumulative, i.e. it always applies to the segment as originally created. The <transformation> is not applied to nested segments.

The default value for the matrix elements M11 and M22 = 1 and for the others 0.

For geometric attributes which are vectors (e.g. CHARACTER UP VECTOR) the values M13 and M23 are ignored.

This primitive may have a variable set of parameters:

a. Two parameters
   - <segment name>
   - <reset matrix> = Yes

b. All listed parameters

In Class C1 and C2 the default will be the only supported <transformation matrix>.

In Class C1 and C2 this primitive will be ignored.

5.1.10.7 GRAPHIC OBJECT-HIGHLIGHT-primitive

Class: C3

Parameters: - Segment name
- Highlight; Highlighted or Normal

Effect: The current highlight status is changed to the specified status. The highlighting will be applied to all the elements in the segment <segment name>, not to nested segments.

The default <highlight> is Normal.

In Class C1 and C2 this primitive is ignored.

5.1.10.8 GRAPHIC OBJECT-VISIBILITY-primitive

Class: C3

Parameters: - Segment name
- Visibility; Visible or Invisible

Effect: The current visibility status is changed to the specified status. The visibility will be applied to all the elements in the segment <segment name>, not to nested segments.

The default <visibility> is Visible.

In Class C1 and C2 this primitive is ignored.

5.2 Relation between Classes and Primitives

The PRIMITIVES defined for the geometric set are divided in four Classes. The Classes are numbered C1, C2, C3 and C4. Each higher Class, C3 is higher than C2, fully contains and supports the PRIMITIVES in the lower Class(es), so the Classes are upwards compatible. In this scheme, Class C0 can be considered as no geometric set.
If future extensions are needed, they can be grouped in new (higher) Classes or in Class C4, so no additions will be made to Class C1, C2 or C3. With this Class scheme the new extensions will not influence already existing PRIMITIVES. The relation between Class and PRIMITIVE is given in table 5-1. In this table an "L" indicates a limited functionality in that Class, while an "F" indicates full functionality.

Primitive name	Class			
	C1	C2	C3	C4
POLYLINE	F			
FILL AREA		F		
POLYMARKER	L	F		
CELL ARRAY		F		
TEXT		F		
INFILL				F
GDP-ARC		F		
GDP-ARC-CHORD		F		
GDP-ARC-PIE		F		
GDP-CIRCLE		F		
GDP-RECTANGLE		F		
GDP-SPLINE		F		
SET-POLYLINE COLOUR	F			
SET-LINE WIDTH	F			
SET-LINE TYPE		F		
SET-POLYLINE INDEX			F	
SET-FILL AREA COLOUR		F		
SET-FILL AREA INTERIOR STYLE		F		
SET-FILL AREA STYLE INDEX		F		
SET-FILL AREA PATTERN		F		
SET-PATTERN SIZE		F		
SET-PATTERN REFERENCE POINT		F		
SET-FILL AREA INDEX			F	
SET-POLYMARKER CLOUR		F		
SET-MARKER TYPE		F		
SET-MARKER SIZE		F		
SET-POLYMARKER INDEX			F	
SET-TEXT COLOUR		F		
SET-CHARACTER FORMAT		F		
SET-CHARACTER UP VECTOR		F		
SET-TEXT PATH		F		
SET-CHARACTER SPACING		F		
SET-TEXT FONT AND PRECISION		F		
SET-TEXT ALIGNMENT		F		
SET-TEXT INDEX			F	
SET-POLYLINE ASF			F	
SET-FILL AREA ASF			F	
SET-POLYMARKER ASF			F	
SET-TEXT ASF			F	
SET-POLYLINE REPRESENTATION			F	
SET-POLYMARKER REPRESENTATION			F	
SET-FILL AREA REPRESENTATION			F	
SET-TEXT REPRESENTATION			F	
SET-PATTERN REPRESENTATION			F	

TABLE 5-1

Relation between Class and Primitive

L = Limited functionality; F = Full functionality
continued on next page

Primitive name	Class			
	C1	C2	C3	C4
CONTROL-CLEAR	L		F	
CONTROL-WINDOW		F		
CONTROL-VIEWPORT		F		
CONTROL-CLIPPING RECTANGLE		F		
CONTROL-DOMAIN RING	L		F	
CONTROL-WAIT		F		
CONTROL-DISPLAY PLANE				F
GRAPHIC OBJECT-CREATE			F	
GRAPHIC OBJECT-CLOSE			F	
GRAPHIC OBJECT-RENAME			F	
GRAPHIC OBJECT-DELETE			F	
GRAPHIC OBJECT-INSERT			F	
GRAPHIC OBJECT-TRANSFORMATION			F	
GRAPHIC OBJECT-HIGHLIGHT			F	
GRAPHIC OBJECT-VISIBILITY			F	

TABLE 5-1

Relation between Class and Primitive

L= Limited functionality; F= Full functionality

continued from previous page

6.0 ENCODING PRINCIPLES OF THE GEOMETRIC SET

The encoding of the geometric set is independent of the encoding of other sets, e.g. alphamosaic and photographic set. The geometric set is selected using the PPCI (Presentation Protocol Control Information) sequence US 3/0 <y>. US is the UNIT SEPARATOR control and is coded 1/15. The <y> indicates the highest Class of PRIMITIVES that is embedded in the geometric data following the PPCI. The currently defined values are given in table 6-1.

<y> code	highest Class
2/1	C1
2/2	C2
2/3	C3
2/4	C4

Table 6-1

Relation between <y> code and Class

After this PPCI all data is regarded as geometric data. Selecting of other sets is only allowed after complete PRIMITIVES and their ATTRIBUTES. This means that after a PRIMITIVE code, the encoding is transparant regarding other sets.

The PPCI sequence US 3/1 is reserved for selecting a three dimensional (3D) geometric set.

The encoding of PRIMITIVES and their ATTRIBUTES is done in a 7 bit environment. Bit 8 of each codeword may be used for parity.

The encoding of the geometric set allows for future extensions. This is achieved by encoding the PRIMITIVES and their ATTRIBUTES separately.

The rest of this chapter will deal with the detailed encoding principles of:

a. The PRIMITIVES and the SUB-PRIMITIVES

b. The ATTRIBUTES

c. The COORDINATES in Direct and Incremental mode

6.1 Encoding principles of PRIMITIVES

The PRIMITIVES are encoded in one or two bytes. The PRIMITIVES that have no SUB-PRIMITIVES are encoded in one byte. The others are encoded in two bytes, the first byte indicating the PRIMITIVE, the second byte indicating the SUB-PRIMITIVE.

The structure of the PRIMITIVE byte is given in figure 6-1.

```
              b8  b7  b6  b5  b4  b3  b2  b1
             ---------------------------------
             ! - ! 0 ! 1 !   !   !   !   !   !
             ---------------------------------
                 \_____/   !  \_______________/
                    !      !          !
Primitive flag _____!      !          !__________ Primitive code
                           !
                           !____________________ Sub-Primitive flag
```

Figure 6-1

8 bit PRIMITIVE Byte

For single byte PRIMITIVES the SUB-PRIMITIVE flag is always 0. If the SUB-PRIMITIVE flag (bit 5) is 1, this PRIMITIVE has SUB-PRIMITIVES. The second byte than has the same format except that the SUB-PRIMITIVE flag has no meaning.

The encoding of the SUB-PRIMITIVES is done for each set of SUB-PRIMITIVES so the SUB-PRIMITIVE encoding is related to the previous PRIMITIVE byte.

6.2 Encoding principles of ATTRIBUTES

Although coordinates are ATTRIBUTES, they will not be described in this paragraph. Because of their special nature the next paragraph will deal with this ATTRIBUTE type.

The general structure of the ATTRIBUTE byte is given in figure 6-2.

```
              b8  b7  b6  b5  b4  b3  b2  b1
             ---------------------------------
             ! - ! 1 !   !   !   !   !   !   !
             ---------------------------------
                     !  \_______________________/
                     !             !
Attribute flag_______!             !_______________ Attribute code
```

Figure 6-2

8 bit ATTRIBUTE byte

The ATTRIBUTES can be of various types. The following types and encoding principles are defined:

a. Coordinate list
   Describes one or more coordinates, for the encoding see next paragraph.

b. Two byte values
   With this type of ATTRIBUTES a value greater than 63 is described, e.g. dimension and wait time. The format of this type is given in figure 6-3.

```
              b8  b7  b6  b5  b4  b3  b2  b1
             --------------------------------
First byte   ¦ - ¦ 1 ¦   ¦   ¦   ¦   ¦   ¦   ¦  Most significant byte
             --------------------------------

              b8  b7  b6  b5  b4  b3  b2  b1
             --------------------------------
Second byte  ¦ - ¦ 1 ¦   ¦   ¦   ¦   ¦   ¦   ¦  Least significant byte
             --------------------------------
```

Figure 6-3

Two byte value encoding

This type of ATTRIBUTE always occupies two bytes even if the most significant byte is 0. The value in this two bytes is an unsigned integer value in the range 0 to 4095.

c. String
This ATTRIBUTE type describes a character string as used in the TEXT-primitive. The characters in the string may only be taken from the invoked G-set or from table 4-1. In an 8-bit alphamosaic environment the characters in the character string may be taken from both invoked G-sets or from table 4-2. In this situation bit 8, see figure 6-4, forms a part of the character.
The format of this type is not according to the general ATTRIBUTE byte structure as given in figure 6-2. The format of the String type ATTRIBUTE is given in figure 6-4.

```
 b8  b7  b6  b5  b4  b3  b2  b1
--------------------------------
¦(-)¦   ¦   ¦   ¦   ¦   ¦   ¦   ¦
--------------------------------
    \__________________________/
                 ¦
                 ¦_______________ Character
```

Figure 6-4

String encoding

The end of the string is reached as soon as a DELETE character (7/15) is detected. The DELETE character will not be displayed.

d. Name
This ATTRIBUTE type describes a segment name, as used in the GRAPHIC OBJECT-primitives. This type exists in two forms, a short and long form. The format of both forms is given in figure 6-5.

```
SHORT FORM

                b8  b7  b6  b5  b4  b3  b2  b1
               ---------------------------------
               ¦ - ¦ 1 ¦   ¦ 0 ¦   ¦   ¦   ¦   ¦
               ---------------------------------
     Flag ____________¦   ¦  \______________/
                          ¦          ¦
Long/short _______________¦          ¦_________ Segment name
indicator

LONG FORM

                b8  b7  b6  b5  b4  b3  b2  b1
               ---------------------------------
               ¦ - ¦ 1 ¦   ¦ 1 ¦   ¦   ¦   ¦   ¦
               ---------------------------------
     Flag ____________¦   ¦  \______________/
                          ¦          ¦
Long/short _______________¦          ¦_________ Most significant part
indicator                                       of segment name

                b8  b7  b6  b5  b4  b3  b2  b1
               ---------------------------------
               ¦ - ¦ 1 ¦   ¦   ¦   ¦   ¦   ¦   ¦
               ---------------------------------
                       \______________________/
                                  ¦
                                  ¦____________ Least significant part
                                                of segment name
```

Figure 6-5

Name and flag encoding

If more than 16 (0 up to 15) names are required, the long form of the segment name should be used. The short form of the name is indicated by a long/short indicator = 0, the long form with a value of 1 for this indicator. So the name is in the range 0 to 15 in short form and in the range 0 to 1023 in long form.

e. Flags
This ATTRIBUTE type defines an ON/OFF ATTRIBUTE like VISIBILITY ON or OFF. The position of the flag bit is given in figure 6-5.

f. Two bit values
This ATTRIBUTE type is used for ATTRIBUTES with a value in the range 0 to 3. The format of the two bit value type is given in figure 6-6.

```
            b8  b7  b6  b5  b4  b3  b2  b1
           --------------------------------
           | - | 1 |   |   |   |   |   |   |
           --------------------------------
                    \_____/ \______________/
                       |           |
Two bit  ______________|           |__________ Four bit values
values
```

Figure 6-6

Two and four bit value encoding

g. Three bit values
This ATTRIBUTE type is used for ATTRIBUTES with a value in the range 0 to 7. The format of the three bit value type is given in figure 6-7.

```
              b8  b7  b6  b5  b4  b3  b2  b1
             --------------------------------
             | - | 1 |   |   |   |   |   |   |
             --------------------------------
                      \_________/ \_________/
                           |          |
First three  ______________|          |________ Second three
bit value                                       bit value
```

Figure 6-7

Three bit value encoding

h. Four bit values
This ATTRIBUTE type is used for ATTRIBUTES with a value in the range 0 to 15. The format is given in figure 6-6.

i. Six bit values
This ATTRIBUTE type is used for ATTRIBUTES with a value in the range 0 to 63. The format is given in figure 6-8.

```
 b8  b7  b6  b5  b4  b3  b2  b1
--------------------------------
| - | 1 |   |   |   |   |   |   |
--------------------------------
         \_____________________/
                    |
                    |______________ Six bit values
```

Figure 6-8

Six bit value encoding

j. Matrix elements
This ATTRIBUTE type describes the elements of a transformation matrix. The format is given in figure 6-9.

```
 b8  b7  b6  b5  b4  b3  b2  b1
 --------------------------------
 | - | 1 | S | X | X | S | Y | Y |   First byte
 --------------------------------
           |___________|___________  Sign
 b8  b7  b6  b5  b4  b3  b2  b1
 --------------------------------
 | - | 1 | X | X | X | Y | Y | Y |   Second byte
 --------------------------------

 b8  b7  b6  b5  b4  b3  b2  b1
 --------------------------------
 | - | 1 | X | X | X | Y | Y | Y |   N-th byte
 --------------------------------
```

Figure 6-9

Matrix element encoding

The number of bytes in this ATTRIBUTE type is dependent on the accuracy value as defined with the CONTROL-DOMAIN RING-primitive. The number of bytes (N) follows from:

N = 2 * (<accuracy value> + 3)

The bits denoted with S (bits b6 and b3 of the first byte) are Sign bits. If S=0 the Sign is positive, if S=1 the Sign is negative.
This type defines two fractional numbers, denoted by X respectively Y. The fractional units are based on the BGU. The actual fractional number specified with this type is formed by all X- respectively Y-bits divided by the BGU.

k. Size values
This ATTRIBUTE type is used to describe ATTRIBUTES that specify a size or format such as the size of the brush and the format of a character. The format is given in figure 6-10.

```
b8  b7  b6  b5  b4  b3  b2  b1
--------------------------------
¦ - ¦ 1 ¦ S ¦ X ¦ X ¦ S ¦ Y ¦ Y ¦   First byte
--------------------------------
        ¦___________¦___________  Sign

b8  b7  b6  b5  b4  b3  b2  b1
--------------------------------
¦ - ¦ 1 ¦ X ¦ X ¦ X ¦ Y ¦ Y ¦ Y ¦   Second byte
--------------------------------

b8  b7  b6  b5  b4  b3  b2  b1
--------------------------------
¦ - ¦ 1 ¦ X ¦ X ¦ X ¦ Y ¦ Y ¦ Y ¦   N-th byte
--------------------------------
```

Figure 6-10

Size value encoding

The number of bytes (N) in this ATTRIBUTE type is dependent on the accuracy value as defined with the CONTROL-DOMAIN RING-primitive and is as given in table 6-2.
The bits denoted with S (bits b6 and b3 of the first byte) are Sign bits. If S=0 the Sign is positive, if S=1 the Sign is negative.
The x- and y-values are encoded in the same way as for Direct mode coordinates, e.g. as fractional units based on the BGU.

In order to be able to distinguish between the different types of ATTRIBUTES they will appear in a, PRIMITIVE dependent, predefined order.

The encoding of the PRIMITIVES and the SUB-PRIMITIVES with their ATTRIBUTES is given in the next chapter.

6.3 Encoding principles of coordinates

The geometric set specified in this document allows two different modes of coordinates:

a. Direct mode, defining absolute coordinates.

b. Incremental mode, defining steps (increments) from one coordinate position to another.

The encoding principles for each of these modes will be described in the following paragraphs.
The encoding of coordinate data is according to the general structure as given in figure 6-2. The general encoding of the first byte of coordinate data is given in figure 6-11.

```
                b8  b7  b6  b5  b4  b3  b2  b1
               --------------------------------
               ¦ - ¦ 1 ¦   ¦   ¦   ¦   ¦   ¦   ¦
               --------------------------------
                     ¦   ¦           ¦__________ Sub-Mode indicator
                     ¦   ¦
Attribute  __________¦   ¦______________________ Mode indicator
flag
```

Figure 6-11

General encoding of the first byte of coordinate data

6.3.1 Encoding principles for Direct mode

The coordinates used in constructing display images are expressed in a device independent coordinate system, the Normalized Device Coordinates (NDC). The NDC coordinates are in the range 0 (inclusive) to 1 (non-inclusive) for both x and y coordinates. The actual coordinates are expressed in fractional units, based on the Basic Grid Unit (BGU). The BGU is a binary fraction that identifies the minimal accuracy of coordinates. The value of the BGU is controlled with the CONTROL-DOMAIN RING-primitive. The encoding format of Direct mode coordinates is given in figure 6-12.

```
                b8  b7  b6  b5  b4  b3  b2  b1
               --------------------------------
               ¦ - ¦ 1 ¦ 0 ¦ X ¦ X ¦ 0 ¦ Y ¦ Y ¦   First byte
               --------------------------------
                     ¦   ¦           ¦__________ Sub-Mode indicator
                     ¦   ¦
Attribute  __________¦   ¦______________________ Direct Mode
flag

                b8  b7  b6  b5  b4  b3  b2  b1
               --------------------------------
               ¦ - ¦ 1 ¦ X ¦ X ¦ X ¦ Y ¦ Y ¦ Y ¦   Second byte
               --------------------------------

                b8  b7  b6  b5  b4  b3  b2  b1
               --------------------------------
               ¦ - ¦ 1 ¦ X ¦ X ¦ X ¦ Y ¦ Y ¦ Y ¦   N-th byte
               --------------------------------
```

Figure 6-12

Direct mode coordinate encoding

The first byte of Direct mode coordinate data is identified with the Mode and Sub-Mode indicator equal to 0. The number of bytes for Direct mode coordinate data is controlled with the CONTROL-DOMAIN RING-primitive. The most significant part of the coordinate data is in the first byte. The least significant part is in the n-th byte.
In table 6-2 the relation between the accuracy parameter of the CONTROL-DOMAIN RING-primitive, the number of bytes for Direct mode coordinate data, the number of coordinate bits and the BGU is given.

Accuracy value	Number of bytes	Number of bits	BGU
0	3	8	1/256
1	4	11	1/2048
2	5	14	1/16384

TABLE 6-2

Relation between Direct mode coordinate values

6.3.2 Encoding principles for Incremental mode

The Incremental mode as defined in this specification is 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 previous identified point. The first centre point is encoded in Direct mode.
A Ring is characterized by its Radius (R in Basic Grid Units), its Angular resolution (by a factor p) and its Direction (D). The maximum number of points on a Ring is 8R. The actual number of points on a Ring with a given Angular resolution factor p follows from:

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

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 6-13 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 6-13

Some example Rings with point numbering

The Direction of a Ring is identified by the position of the point with number 0. 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 0 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 consequtive 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 6-14, 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 6-13. The Direction (position of the point with number 0) is identified by D.

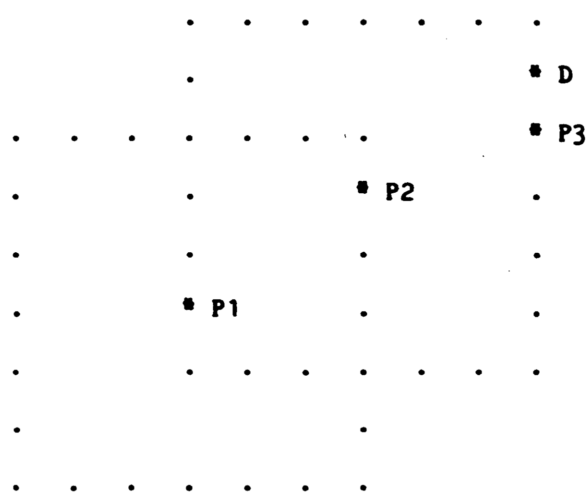


Figure 6-14

Change of direction with R=3

The basic Radius of the Ring, as used in the Incremental mode, is dependent on the BGU. Table 6-3 gives the relation between the Accuracy value, the default basic Radius and the BGU.

Accuracy value	Basic Radius	BGU
0	R=1	1/256
1	R=8	1/2048
2	R=64	1/16384

TABLE 6-3

Relation between Accuracy and basic Radius

The basic Radius as defined in table 6-3 may be changed to any value with the CONTROL-DOMAIN RING-primitive. With this primitive one can also change the Angular resolution factor p. The default value for p=0 and p can only be 0, 1, 2 or 3.

The encoding used in Incremental mode makes use of the DCC property by using variable length code-words (Huffman Code). The encoding also allows chahging of the Radius and the Angular resolution factor. The Radius can have a value of R, 2R, 3R or 4R, where R is the defined Radius. The Angular resolution factor p can be 0, 1, 2, or 3.
The Huffman Code table used in the Incremental mode is a fixed length table. To allow the encoding of more points on a Ring two Escapes are defined. With these Escapes the points outside the Huffman Code table can be addressed. The end of the Incremental mode data is signalled by an End of Block value in the Huffman Code table.
The fixed Huffman Code table is given in table 6-4.

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	Escape 1
10	1111111110	Escape 2
10	1111111111	End of Block

TABLE 6-4

Huffman Code table for Incremental mode

The <End of Block> code from the Huffman Code table identifies the end of the Incremental mode data. Remaining bits in the last Incremental mode data byte have no meaning, they will be ignored.
The escape codes <Escape 1> and <Escape 2> are used to extend the addressable number of points, e.g. points outside the range -20 to 19. The code <Escape 1> adds +20 or -20 to the following code depending on the sign of that following point. The code <Escape 2> adds +40 or -40 to the following code, depending on the sign. The escape codes can follow each other in any desired order. Example 6-1 demonstrates some possible combinations, [n] is a point number.

```
             <Escape 1> [ 1]  =  point number 21
             <Escape 1> [-1]  =  point number -21
             <Escape 2> [14]  =  point number 54
             <Escape 2> [-12] =  point number -52
<Escape 1>   <Escape 2> [  6] = point number (20+40+6) = 66
<Escape 2>   <Escape 1> [-18] = point number (-40+-20+-18) = -78
```

Example 6-1

The use of Escape codes in the Incremental mode

The codes C1 up to C6 are used to change the parameters that define the Ring to be used. The function of these codes is as follows:

a. C1
Change the Ring parameters, R and p, to the next higher value e.g. if the Radius is R, the next higher is 2R, if p=0 the next higher value is 1. The values of R are taken from the range: R, 2R, 3R and 4R. The values of p are taken from the range: 0, 1, 2 and 3. R cannot become greater than 4R and p cannot become greater than 4. For example if the current Ring Radius is 4R and the current p=3, the code <C1> has no effect.

b. C2
Change the Ring parameters, R and p, to the next lower value. The effect of the code <C2> is the inverse of code <C1>. R cannot become smaller than R and p cannot become smaller than 0. For example if the current Radius is R and the current p=0, the code <C2> has no effect.

c. C3
Change the Ring Radius R to the next higher value. The code <C3> has no effect if the current Radius = 4R.

d. C4
Change the Angular resolution factor p to the next higher value. The code <C4> has no effect if the current p = 3.

e. C5
Change the Ring Radius R to the next lower value. The code <C5> has no effect if the current Radius = R.

f. C6
Change the Angular resolution factor p to the next lower value. The code <C6> has no effect if the current p = 0.

The Incremental mode can be used in two different versions:

a. 6-bit version

b. 8-bit version

The following sections will describe the encoding of the versions.

6.3.2.1 <u>Encoding principles for the 6-bit Incremental mode version</u>

In the 6-bit Incremental mode version all data bytes are encoded according to the general structure as given in figure 6-2. The encoding format of the 6-bit Incremental mode version is given in figure 6-15.

```
                  b8  b7  b6  b5  b4  b3  b2  b1
                 --------------------------------
                 | - | 1 | 1 | 1 | 0 | 0 | 0 | 0 |   First byte
                 --------------------------------
Attribute  ________|   |   |____________________   6-bit version
flag                   |
                       |________________________   Incremental mode

                  b8  b7  b6  b5  b4  b3  b2  b1
                 --------------------------------
                 | - | 1 | I | I | I | I | I | I |   Other bytes
                 --------------------------------
                       \_______________________/
                                  |
                                  |_____________   Incremental mode data
```

Figure 6-15

6-bit Incremental mode version encoding

The bits b4, b3, b2 and b1 of the first byte are reserved for future use and now set to 0.

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 least significant part is packed in the second byte. If the remaining part does not fit in the second byte a third byte is used 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.

6.3.2.2 Encoding principles for the 8-bit Incremental mode version

In the 8-bit Incremental mode version the encoding of the Incremental data is as efficient as possible. In this version all 8-bits of each byte are used to encode the Incremental data. This implies that the encoding in this version is not according to the general structure as given in figure 6-2. The encoding format of this version is given in figure 6-16.

```
                  b8  b7  b6  b5  b4  b3  b2  b1
                  --------------------------------
                  | - | 1 | 1 | 0 | 0 | 0 | 0 | 0 |   First byte
                  --------------------------------
Attribute  ________|   |   |______________________   8-bit version
flag                   |
                       |__________________________   Incremental mode

                  b8  b7  b6  b5  b4  b3  b2  b1
                  --------------------------------
                  | I | I | I | I | I | I | I | I |   Other bytes
                  --------------------------------
                  \______________________________/
                               |
                               |__________________   Incremental mode data
```

Figure 6-16

8-bit Incremental mode version encoding

The bits b4, b3, b2 and b1 of the first byte are reserved for future use and now set to 0.

The Incremental mode data in the 8-bit version requires transparent transmission. For a description of this transparent transmission see Part 7.

The end of Incremental mode data is identified by the <End of Block> code from the Huffman Code table (table 6-4). Remaining bits in the last byte, after the <End of Block> code, are ignored.

The variable length code-words are packed in consecutive bits of a byte. If a code word does not fit in one byte, the most significant part is packed in the first byte, the least significant part is packed in the second byte (see figure 6-16).

6.3.3 Combining coordinate modes

Within a PRIMITIVE the ATTRIBUTES identifying a coordinate list may combine data of each of the two different modes for coordinate data.
The first byte of a group of coordinate bytes identifies the mode of that group, the mode indicator in figure 6-11. The number of bytes for Direct mode coordinate data is determined by the CONTROL-DOMAIN RING-primitive, e.g. 3, 4 or 5 bytes. The end of coordinate (incremental) data in the Incremental mode is determined by the <End of Block> code from the Huffman Code table (table 6-4).
The first byte after a group of coordinate data bytes can be:

a. Another group of coordinate data bytes in Direct or Incremental mode.

b. Another PRIMITIVE byte.

c. A PPCI selecting another set, e.g. alphamosaic or photographic or another Class.

7.0 GEOMETRIC SET ENCODING

The actual encoding of the geometric set will be described in two parts. The first part will describe the encoding of the PRIMTIVES and the SUB-PRIMITIVES. In the second part, the full encoding of the ATTRIBUTES of each PRIMITIVE and SUB-PRIMITIVE will be given. The first part will be described in the next paragraph. The second part will have a separate paragraph for each PRIMITIVE or SUB-PRIMITIVE.

7.1 PRIMITIVE and SUB-PRIMITIVE encoding

The encoding of the PRIMITIVES is given in table 7-1. Bit b8 may be used for Parity and is not shown in this table.

Primitive flag b7	b6	Sub-Primitive flag b5	Primitive Code b4	b3	b2	b1	Primitive
0	1	0	0	0	0	0	POLYLINE
0	1	0	0	0	0	1	FILL AREA
0	1	0	0	0	1	0	POLYMARKER
0	1	0	0	0	1	1	CELL ARRAY
0	1	0	0	1	0	0	TEXT
0	1	0	0	1	0	1	INFILL
0	1	0	0	1	1	0	\
.	.	.	.	.	.	.	> Reserved
0	1	0	1	1	1	1	/
0	1	1	0	0	0	0	GDP
0	1	1	0	0	0	1	SET
0	1	1	0	0	1	0	CONTROL
0	1	1	0	0	1	1	GRAPHIC OBJECT
0	1	1	0	1	0	0	\
.	.	.	.	.	.	.	> Reserved
0	1	1	1	1	1	1	/

TABLE 7-1

Encoding of PRIMITIVES

The tables 7-2, 7-3, 7-4 and 7-5 give the encoding for the SUB-PRIMITIVES of GDP, SET, CONTROL and GRAPHIC OBJECT respectively. Bit b8 may be used for Parity and is not shown in this tables.

Primitive flag b7	b6	Sub-Primitive code b5	b4	b3	b2	b1	SUB-PRIMITIVE of GDP
0	1	0	0	0	0	0	ARC
0	[illegible]	0	0	0	0	1	ARC-CHORD
0	[illegible]	0	0	0	1	0	ARC-PIE
0	[illegible]	0	0	0	1	1	CIRCLE
0	[illegible]	0	0	1	0	0	RECTANGLE
0	1	0	0	1	0	1	SPLINE
0	1	0	0	1	1	0	\
.	.	.	.	.	.	.	> Reserved
0	1	1	1	1	1	1	/

TABLE 7-2

Encoding of GDP SUB-PRIMITIVES

Primitive flag b7	b6	Sub-Primitive code b5	b4	b3	b2	b1	SUB-PRIMITIVE of SET
0	1	0	0	0	0	0	POLYLINE COLOUR
0	1	0	0	0	0	1	LINE WIDTH
0	1	0	0	0	1	0	LINE TYPE
0	1	0	0	0	1	1	POLYLINE INDEX
0	1	0	0	1	0	0	FILL AREA COLOUR
0	1	0	0	1	0	1	FILL AREA INTERIOR STYLE
0	1	0	0	1	1	0	FILL AREA STYLE INDEX
0	1	0	0	1	1	1	FILL AREA PATTERN
0	1	0	1	0	0	0	PATTERN SIZE
0	1	0	1	0	0	1	PATTERN REFERENCE POINT
0	1	0	1	0	1	0	FILL AREA INDEX
0	1	0	1	0	1	1	POLYMARKER COLOUR
0	1	0	1	1	0	0	MARKER TYPE
0	1	0	1	1	0	1	MARKER SIZE
0	1	0	1	1	1	0	POLYMARKER INDEX
0	1	0	1	1	1	1	TEXT COLOUR
0	1	1	0	0	0	0	CHARACTER FORMAT
0	1	1	0	0	0	1	CHARACTER UP VECTOR
0	1	1	0	0	1	0	TEXT PATH
0	1	1	0	0	1	1	CHARACTER SPACING
0	1	1	0	1	0	0	TEXT FONT AND PRECISION
0	1	1	0	1	0	1	TEXT ALIGNMENT
0	1	1	0	1	1	0	TEXT INDEX
0	1	1	0	1	1	1	POLYLINE ASF
0	1	1	1	0	0	0	FILL AREA ASF
0	1	1	1	0	0	1	POLYMARKER ASF
0	1	1	1	0	1	0	TEXT ASF
0	1	1	1	0	1	1	POLYLINE REPRESENTATION
0	1	1	1	1	0	0	FILL AREA REPRESENTATION
0	1	1	1	1	0	1	POLYMARKER REPRESENTATION
0	1	1	1	1	1	0	TEXT REPRESENTATION
0	1	1	1	1	1	1	PATTERN REPRESENTATION

TABLE 7-3

Encoding of SET SUB-PRIMITIVES

Primitive flag b7 b6		Sub-Primitive code b5	b4	b3	b2	b1	SUB-PRIMITIVE of CONTROL
0	1	0	0	0	0	0	CLEAR
0	1	0	0	0	0	1	WINDOW
0	1	0	0	0	1	0	VIEWPORT
0	1	0	0	0	1	1	CLIPPING RECTANGLE
0	1	0	0	1	0	0	DOMAIN RING
0	1	0	0	1	0	1	WAIT
0	1	0	0	1	1	0	DISPLAY PLANE
0	1	0	0	1	1	1	\
.	.	.	.	.	.	.	> Reserved
0	1	1	1	1	1	1	/

TABLE 7-4

Encoding of CONTROL SUB-PRIMITIVES

Primitive flag b7 b6		Sub-Primitive code b5	b4	b3	b2	61	SUB-PRIMITIVE of GRAPHIC OBJECT
0	1	0	0	0	0	0	CREATE
0	1	0	0	0	0	1	CLOSE
0	1	0	0	0	1	0	RENAME
0	1	0	0	0	1	1	DELETE
0	1	0	0	1	0	0	INSERT
0	1	0	0	1	0	1	TRANSFORMATION
0	1	0	0	1	1	0	HIGHLIGHT
0	1	0	0	1	1	1	VISIBILITY
0	1	0	1	0	0	0	\
.	.	.	.	.	.	.	> Reserved
0	1	1	1	1	1	1	/

TABLE 7-5

Encoding of GRAPHIC OBJECT SUB-PRIMITIVES

7.2 ATTRIBUTE encoding

The encoding of the PRIMITIVES, SUB-PRIMITIVES and their ATTRIBUTES will be given in the next paragraphs. Each paragraph will specify the full encoding of a particular PRIMITIVE or SUB-PRIMITIVE and the related ATTRIBUTES. The specification will have a general form. The heading of the paragraph gives the PRIMITIVE or SUB-PRIMITIVE name.

Than follows the encoding of the PRIMITIVE or SUB-PRIMITIVE. The bits in these bytes are numbered b8 on the left to b1 on the right. Bit b8 may be used for parity, denoted with -.

b8	b7	b6	b5	b4	b3	b2	b1	
-	0	1						PRIMITIVE encoding

b8	b7	b6	b5	b4	b3	b2	b1	
-	0	1						SUB-PRIMITIVE encoding if applicable

Then follows the following basic format:

Parameters: The parameters/attributes of this PRIMITIVE, including the type and encoding.

Remarks: Constraints on the use of the parameters, the value ranges etcetera.

If a PRIMITIVE has a <coordinate list> as ATTRIBUTE it will be mentioned. The <coordinate list> can contain coordinates in Direct or Incremental mode. The Incremental mode data can be in a 6-bit or an 8-bit version. The encoding of the coordinates is as specified in the concerned paragraphs.

7.2.1 Encoding of POLYLINE

-	0	1	0	0	0	0	0	POLYLINE-primitive

Parameters: - Coordinate list
In Direct or Incremental mode
2 or more coordinates

Remarks: Coordinates of different modes may be combined.

The first coordinate of the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list> point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.2 Encoding of FILL AREA

```
---------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 0 | 1 |      FILL AREA-primitive
---------------------------------
```

Parameters: - Coordinate list
In Direct or Incremental mode
3 or more coordinates

Remarks: Coordinates of different modes may be combined.

The first coordinate of the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list>, point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.3 Encoding of POLYMARKER

```
---------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 1 | 0 |      POLYMARKER-primitive
---------------------------------
```

Parameters: - Coordinate list
In Direct or Incremental mode
1 or more coordinates

Remarks: Coordinates of different modes may be combined.

The first coordinate in the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list>, point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.4 Encoding of CELL ARRAY

```
---------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 1 | 1 |      CELL ARRAY-primitive
---------------------------------
```

Parameters: - Coordinate list
In Direct mode
The <coordinate list> contains exactly two coordinates;
(xmin, ymin) and (xmax, ymax)

- Two byte value, m

```
---------------------------------
| - | 1 | m | m | m | m | m | m |
---------------------------------

---------------------------------
| - | 1 | m | m | m | m | m | m |
---------------------------------
```

- Two byte value, n

```
---------------------------------
| - | 1 | n | n | n | n | n | n |
---------------------------------

---------------------------------
| - | 1 | n | n | n | n | n | n |
---------------------------------
```

- Entry number in CLUT
  In three-bit-values
  One value for each cell

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
        \__________/\__________/
             |          |________ Second colour
             |
             |___________________ First colour
```

Remarks: The value <xmin> is the smallest absolute x-value in the coordinate list, as is <ymin> the smallest absolute y-value. The values <xmax> and <ymax> are the largest absolute x- respectively y-value in the coordinate list.

For each cell a colour is defined with a <entry number in CLUT> parameter. The number of colours is m*n, if this is odd the last <entry number in CLUT> parameter has no meaning.

The colour table (CLUT) used is the last selected colour table, e.g. selected with the last SET-COLOUR-primitive.

7.2.5 Encoding of TEXT

```
---------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 0 | 0 |     TEXT-primitive
---------------------------------
```

Parameters:
- Coordinate list
  In Direct mode
  The <coordinate list> contains exactly one coordinate
- Character string

Encoded according to the String ATTRIBUTE type

```
 b8  b7  b6  b5  b4  b3  b2  b1
---------------------------------
|(-)|   |   |   |   |   |   |   |
---------------------------------
    \___________________________/
                 |
                 |_______________ Character
```

Remarks: The end of the <character string> is determined with the DELETE character (7/15). The characters in the string may only be taken from the G-set or from table 4-1 in a 7 bit environment. In an 8-bit environment they may be taken from the G-sets or from table 4-2.

7.2.6 Encoding of INFILL

```
---------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 0 | 1 |      INFILL-primitive
---------------------------------
```

Parameters: - Coordinate list
In Direct mode
The <coordinate list> contains exactly one coordinate

Remarks: None

7.2.7 Encoding of GDP-ARC

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 0 |      GDP-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 0 | 0 |      ARC-sub-primitive
---------------------------------
```

Parameters: - Coordinate list
In Direct or Incremental mode
The number of coordinates is a multiple of 3

Remarks: Each set of 3 coordinates indicates an ARC.

Coordinates of different modes may be combined.

The first coordinate of the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list>, point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.8 Encoding of GDP-ARC CHORD

-	0	1	1	0	0	0	0	GDP-primitive

-	0	1	0	0	0	0	1	ARC CHORD-sub-primitive

Parameters: - Coordinate list
In Direct or Incremental mode
The number of coordinates is a multiple of 3

Remarks: Each set of 3 coordinates indicates an ARC CHORD.

Coordinates of different modes may be combined.

The first coordinate of the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list>, point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.9 Encoding of GDP-ARC PIE

-	0	1	1	0	0	0	0	GDP-primitive

-	0	1	0	0	0	1	0	ARC PIE-sub-primitive

Parameters: - Coordinate list
In Direct or Incremental mode
The number of coordinates is a multiple of 3

Remarks: Each set of 3 coordinates indicates an ARC PIE.

Coordinates of different modes may be combined.

The first coordinate of the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list>, point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.10 Encoding of GDP-CIRCLE

-	0	1	1	0	0	0	0	GDP-primitive

-	0	1	0	0	0	1	1	CIRCLE-sub-primitive

Parameters: - Coordinate list
In Direct or Incremental mode
The number of coordinates is a multiple of 2

Remarks: Each set of 2 coordinates indicates a CIRCLE.

Coordinates of different modes may be combined.

The first coordinate of the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list>, point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.11 Encoding of GDP-RECTANGLE

-	0	1	1	0	0	0	0	GDP-primitive

-	0	1	0	0	1	0	0	RECTANGLE-sub-primitive

Parameters: - Coordinate list
In Direct or Incremental mode
The number of coordinates is a multiple of 2

Remarks: Each set of 2 coordinates indicates a RECTANGLE.

Coordinates of different modes may be combined.

The first coordinate of the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list>, point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.12 Encoding of GDP-SPLINE

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 0 |      GDP-primitive
-------------------------------
-------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 0 | 1 |      SPLINE-sub-primitive
-------------------------------
```

Parameters: - Coordinate list
In Direct or Incremental mode
3 or more coordinates

Remarks: Coordinates of different modes may be combined.

The first coordinate of the <coordinate list> must be in Direct mode.

If Incremental mode is used in the second coordinate of the <coordinate list>, point number 0 of the initial Ring, the Direction, is on the positive x-axis, while the cartesian axes are drawn through the initial Ring centre point.

7.2.13 Encoding of SET-POLYLINE COLOUR

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 0 | 0 |      POLYLINE COLOUR-sub-primitive
-------------------------------
```

Parameters: - Colour value
In two three-bit-values
Colour table and entry number

```
-------------------------------
| - | 1 |   |   |   |   |   |   |
-------------------------------
        \__________/\__________/
              |           |________ Entry number in CLUT
              |
              |____________________ Colour table
```

Remarks: The <colour table> is in the range 0 to 3. The other values are reserved for future use.

7.2.14 Encoding of SET-LINE WIDTH

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 0 | 1 |     LINE WIDTH-sub-primitive
-------------------------------
```

Parameters: - Line width
In a Size value

```
-------------------------------
| - | 1 | 0 | 0 | 0 | 0 | Y | Y |     First byte
-------------------------------

-------------------------------
| - | 1 | 0 | 0 | 0 | Y | Y | Y |     Second byte
-------------------------------

-------------------------------
| - | 1 | 0 | 0 | 0 | Y | Y | Y |     N-th byte
-------------------------------
```

Remarks: The <line width> defines the width of the line in Basic Grid Units. If the Y-value is exactly 0 (all bits equal 0) the default size is used.
The other bits of the <line width> are for future use, now set to 0.

The number of bytes (N) for encoding the <line width> depends on the accuracy value as defined by the CONTROL-DOMAIN RING-primitive (see table 6-2).

7.2.15 Encoding of SET-LINE TYPE

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 1 | 0 |     LINE TYPE-sub-primitive
-------------------------------
```

Parameters: - Line type
In one three-bit-value

```
-------------------------------
| - | 1 |   |   |   | 0 | 0 | 0 |
-------------------------------
         \___________/
               |_________________ Line type
```

Remarks: The <line type> is in the range 0 to 3. The other values are reserved for future use. The second three-bit-value is for future use, now set to 0.

7.2.16 Encoding of SET-POLYLINE INDEX

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 1 | 1 |      POLYLINE INDEX-sub-primitive
-------------------------------
```

Parameters: - Index
In one three-bit-value

```
-------------------------------
| - | 1 |   |   |   | 0 | 0 | 0 |
-------------------------------
        \___________/
              |__________________ Index
```

Remarks: The second three-bit-value is for future use, now set to 0.

7.2.17 Encoding of SET-FILL AREA COLOUR

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 0 | 0 |      FILL AREA COLOUR-sub-primitive
-------------------------------
```

Parameters: - Colour value
In two three-bit-values
Colour table and entry number

```
-------------------------------
| - | 1 |   |   |   |   |   |   |
-------------------------------
        \___________/\___________/
              |            |________ Entry number in CLUT
              |
              |__________________ Colour table
```

Remarks: The <colour table> is in the range 0 to 3. The other values are reserved for future use.

7.2.18 Encoding of SET-FILL AREA INTERIOR STYLE

```
---------------------------------
¦ - ¦ 0 ¦ 1 ¦ 1 ¦ 0 ¦ 0 ¦ 0 ¦ 1 ¦      SET-primitive
---------------------------------

---------------------------------
¦ - ¦ 0 ¦ 1 ¦ 0 ¦ 0 ¦ 1 ¦ 0 ¦ 1 ¦      FILL AREA INTERIOR STYLE-sub-primitive
---------------------------------
```

Parameters: - Interior style
In one three-bit-value

```
---------------------------------
¦ - ¦ 1 ¦   ¦   ¦   ¦ 0 ¦ 0 ¦ 0 ¦
---------------------------------
        \___________/
              ¦____________________ Interior style
```

Remarks: The <interior style> is in the range 0 to 3. The other values are reserved for future use. The second three-bit-value is for future use, now set to 0.

7.2.19 Encoding of SET-FILL AREA STYLE INDEX

```
---------------------------------
¦ - ¦ 0 ¦ 1 ¦ 1 ¦ 0 ¦ 0 ¦ 0 ¦ 1 ¦      SET-primitive
---------------------------------

---------------------------------
¦ - ¦ 0 ¦ 1 ¦ 0 ¦ 0 ¦ 1 ¦ 1 ¦ 0 ¦      FILL AREA STYLE INDEX-sub-primitive
---------------------------------
```

Parameters: - Style index
In one three-bit-value
Hatch style or Pattern index

```
---------------------------------
¦ - ¦ 1 ¦   ¦   ¦   ¦ 0 ¦ 0 ¦ 0 ¦
---------------------------------
        \___________/
              ¦____________________ Style index
```

Remarks: The <style index> is not used for interior styles HOLLOW and SOLID. For interior style PATTERN it is in the range 0 to 7. For interior style HATCH it is in the range 0 to 5. The second three-bit-value is for future use, now set to 0.

7.2.20 Encoding of SET-FILL AREA PATTERN

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 1 | 1 |      FILL AREA PATTERN-sub-primitive
---------------------------------
```

Parameters: - Dimension
defined with:

- Two byte value, m

```
---------------------------------
| - | 1 | m | m | m | m | m | m |
---------------------------------

---------------------------------
| - | 1 | m | m | m | m | m | m |
---------------------------------
```

- Two byte value, n

```
---------------------------------
| - | 1 | n | n | n | n | n | n |
---------------------------------

---------------------------------
| - | 1 | n | n | n | n | n | n |
---------------------------------
```

- Entry number in CLUT
In three-bit-values
One value for each cell

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
         \__________/\__________/
               |          |________ Second colour
               |
               |___________________ First colour
```

Remarks: For each cell in the pattern style a colour is defined with one <entry number in CLUT> parameter.
The number of <entry number in CLUT> parameters is m*n, if the number is odd the last <entry number in CLUT> parameter has no meaning.

7.2.21 Encoding of SET-PATTERN SIZE

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 0 | 1 | 0 | 0 | 0 |     PATTERN SIZE-sub-primitive
---------------------------------
```

Parameters: - Coordinate list
In Direct mode
The <coordinate list> contains exactly two coordinates;
(xmin, ymin) and (xmax, ymax)

Remarks: The value <xmin> in the <coordinate list> is the smallest absolute x-value in both coordinates as is <ymin> the smallest absolute y-value. The value <xmax> and <ymax> are the largest absolute x- respectively y-value in the coordinates.

7.2.22 Encoding of SET-PATTERN REFERENCE POINT

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 0 | 1 | 0 | 0 | 1 |     PATTERN REFERENCE POINT-sub-primitive
---------------------------------
```

Parameters: - Coordinate list
In Direct mode
The <coordinate list> contains exactly one coordinate

Remarks: None

7.2.23 Encoding of SET-FILL AREA INDEX

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 0 | 1 | 0 | 1 | 0 |     FILL AREA INDEX-sub-primitive
---------------------------------
```

Parameters: - Index
In one three-bit-value

```
-------------------------------
| - | 1 |   |   |   | 0 | 0 | 0 |
-------------------------------
       \__________/
            |_________________ Index
```

Remarks: The second three-bit-value is for future use, now set to 0.

7.2.24 <u>Encoding of SET-POLYMARKER COLOUR</u>

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 1 | 0 | 1 | 1 |     POLYMARKER COLOUR-sub-primitive
-------------------------------
```

Parameters: - Colour value
In two three-bit-values
Colour table and entry number

```
-------------------------------
| - | 1 |   |   |   |   |   |   |
-------------------------------
       \__________/\__________/
            |          |_______ Entry number in CLUT
            |
            |_________________ Colour table
```

Remarks: The <colour table> is in the range 0 to 3. The other values are reserved for future use.

7.2.25 <u>Encoding of SET-MARKER TYPE</u>

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 1 | 1 | 0 | 0 |     MARKER TYPE-sub-primitive
-------------------------------
```

Parameters: - Marker type
In one three-bit-value

```
-------------------------------
| - | 1 |   |   |   | 0 | 0 | 0 |
-------------------------------
       \__________/
            |_________________ Marker type
```

Remarks: The <marker type> is in the range 0 to 4. The other values are reserved for future use. The second three-bit-value is for future use, now set to 0.

7.2.26 Encoding of SET-MARKER SIZE

-	0	1	1	0	0	0	1	SET-primitive

-	0	1	0	1	1	0	1	MARKER SIZE-sub-primitive

Parameters: - Marker size
In a Size value

-	1	0	X	X	0	Y	Y	First byte

-	1	X	X	X	Y	Y	Y	Second byte

-	1	X	X	X	Y	Y	Y	N-th byte

Remarks: The x-value of the <marker size> defines the width of the marker. The y-value defines the height, both in Basic Grid Units. If the X- and Y-values are exactly 0 (all bits equal 0) the default size is used.

The number of bytes (N) for encoding the <marker size> depends on the accuracy value as defined by the CONTROL-DOMAIN RING-primitive (see table 6-2).

7.2.27 Encoding of SET-POLYMARKER INDEX

-	0	1	1	0	0	0	1	SET-primitive

-	0	1	0	1	1	1	0	POLYMARKER INDEX-sub-primitive

Parameters: - Index
In one three-bit-value

```
            -------------------------------
            | - | 1 |   |   | 0 | 0 | 0 |
            -------------------------------
                   \__________/
                        |__________________ Index
```

Remarks: The second three-bit-value is for future use, now set to 0.

7.2.28 Encoding of SET-TEXT COLOUR

```
  -------------------------------
  | - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
  -------------------------------

  -------------------------------
  | - | 0 | 1 | 0 | 1 | 1 | 1 | 1 |      TEXT COLOUR-sub-primitive
  -------------------------------
```

Parameters: - Colour value
In two three-bit-values
Colour table and entry number

```
            -------------------------------
            | - | 1 |   |   |   |   |   |   |
            -------------------------------
                   \__________/\__________/
                        |           |________ Entry number in CLUT
                        |
                        |____________________ Colour table
```

Remarks: The <colour table> is in the range 0 to 3. The other values are reserved for future use.

7.2.29 Encoding of SET-CHARACTER FORMAT

```
  -------------------------------
  | - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
  -------------------------------

  -------------------------------
  | - | 0 | 1 | 1 | 0 | 0 | 0 | 0 |      CHARACTER FORMAT-sub-primitive
  -------------------------------
```

Parameters: - Character height and width
In a Size value

```
-------------------------------------
| - | 1 | 0 | X | X | 0 | Y | Y |    First byte
-------------------------------------

-------------------------------------
| - | 1 | X | X | X | Y | Y | Y |    Second byte
-------------------------------------

-------------------------------------
| - | 1 | X | X | X | Y | Y | Y |    N-th byte
-------------------------------------
```

Remarks: The x-value in the <character height and width> defines the character width. The y-value defines the character height, both in Basic Grid Units.
The, alphamosaic, default width and height is indicated by an x- and y-value that is exactly 0 (all bits equal 0).

The number of bytes (N) for encoding the <character height and width> depends on the accuracy value as defined by the CONTROL-DOMAIN RING-primitive (see table 6-2).

7.2.30 Encoding of SET-CHARACTER UP VECTOR

```
-------------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |    SET-primitive
-------------------------------------

-------------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |    CHARACTER UP VECTOR-sub-primitive
-------------------------------------
```

Parameters: - Up vector
In a Size value

```
-------------------------------------
| - | 1 | S | X | X | S | Y | Y |    First byte
-------------------------------------
        |___________|____________    Sign

-------------------------------------
| - | 1 | X | X | X | Y | Y | Y |    Second byte
-------------------------------------

-------------------------------------
| - | 1 | X | X | X | Y | Y | Y |    N-th byte
-------------------------------------
```

Remarks: The <up vector> defines a vector indicating the up direction of the characters.

The Sign bits (S) may indicate a positive as well as a negative value.

An <up vector> of 0 (all x and y bits 0) gives the default vector.

The number of bytes (N) for encoding the <up vector> depends on the accuracy value as defined by the CONTROL-DOMAIN RING-primitive (see table 6-2).

7.2.31 Encoding of SET-TEXT PATH

```
 ------------------------------
 ¦ - ¦ 0 ¦ 1 ¦ 1 ¦ 0 ¦ 0 ¦ 0 ¦ 1 ¦      SET-primitive
 ------------------------------

 ------------------------------
 ¦ - ¦ 0 ¦ 1 ¦ 1 ¦ 0 ¦ 0 ¦ 1 ¦ 0 ¦      TEXT PATH-sub-primitive
 ------------------------------
```

Parameters: - Path indicator
In a three bit value

```
 ------------------------------
 ¦ - ¦ 1 ¦   ¦   ¦   ¦ 0 ¦ 0 ¦ 0 ¦
 ------------------------------
         \___________/
               ¦___________________ Path indicator
```

Remarks: The <path indicator> is in the range 0 to 3. The other values are for future use. The second three-bit-value is for future use, now set to 0.

7.2.32 Encoding of SET-CHARACTER SPACING

```
 ------------------------------
 ¦ - ¦ 0 ¦ 1 ¦ 1 ¦ 0 ¦ 0 ¦ 0 ¦ 1 ¦      SET-primitive
 ------------------------------

 ------------------------------
 ¦ - ¦ 0 ¦ 1 ¦ 1 ¦ 0 ¦ 0 ¦ 1 ¦ 1 ¦      CHARACTER SPACING-sub-primitive
 ------------------------------
```

Parameters: - Character spacing
In a Size value

```
-------------------------------
| - | 1 | S | X | X | 0 | 0 | 0 |    First byte
-------------------------------
          |________________________  Sign

-------------------------------
| - | 1 | X | X | X | 0 | 0 | 0 |    Second byte
-------------------------------

-------------------------------
| - | 1 | X | X | X | 0 | 0 | 0 |    N-th byte
-------------------------------
```

Remarks: The x-value of <character spacing> defines the space between characters, the inter-character spacing, in Basic Grid Units. The Sign bit (S) may indicate a positive as well as a negative space. The other bits of the <character spacing> are reserved for future use, now set to 0.

The number of bytes (N) for encoding the <character spacing> depends on the accuracy value as defined by the CONTROL-DOMAIN RING-primitive (see table 6-2).

7.2.33 Encoding of SET-TEXT FONT AND PRECISION

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |    SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 1 | 0 | 1 | 0 | 0 |    TEXT FONT AND PRECISION-sub-primitive
-------------------------------
```

Parameters:
- Text font
  In a four-bit-value
- Text precision
  In a two-bit-value

```
-------------------------------
| - | 1 |   |   |   |   |   |   |
-------------------------------
        \_______/\_____________/
            |          |__________  Text font
            |
            |_____________________  Text precision
```

Remarks: The <text font> is in the range 0 to 15.

The <text precision> is in the range 0 to 2. Value 3 is reserved for future use.

7.2.34 Encoding of SET-TEXT ALIGNMENT

```
--------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
--------------------------------

--------------------------------
| - | 0 | 1 | 1 | 0 | 1 | 0 | 1 |     TEXT ALIGNMENT-sub-primitive
--------------------------------
```

Parameters: - Horizontal alignment
In a three-bit-value

- Vertical alignment
In a three-bit-value

```
--------------------------------
| - | 1 |   |   |   |   |   |   |
--------------------------------
        \__________/\__________/
             |           |________ Vertical alignment
             |
             |____________________ Horizontal alignment
```

Remarks: The <horizontal alignment> is in the range 0 to 3. The other values are reserved for future use.

The <vertical alignment> is in the range 0 to 5. The other values are for future use.

7.2.35 Encoding of SET-TEXT INDEX

```
--------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
--------------------------------

--------------------------------
| - | 0 | 1 | 1 | 0 | 1 | 1 | 0 |     TEXT INDEX-sub-primitive
--------------------------------
```

Parameters: - Index
In one three-bit-value

```
--------------------------------
| - | 1 |   |   |   | 0 | 0 | 0 |
--------------------------------
        \__________/
             |____________________ Index
```

Remarks: The second three-bit-value is for future use, now set to 0.

7.2.36 Encoding of SET-POLYLINE ASF

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 1 | 0 | 1 | 1 | 1 |      POLYLINE ASF-sub-primitive
-------------------------------
```

Parameters:
- Polyline colour ASF
  In a flag
- Line width ASF
  In a flag
- Line type ASF
  In a flag

```
-------------------------------
| - | 1 |   |   |   | 0 | 0 | 0 |
-------------------------------
          |   |   |________________ Line type ASF
          |   |
          |   |____________________ Line width ASF
          |
          |________________________ Polyline colour ASF
```

Remarks: The undefined bits, b3, b2 and b1, are reserved for future use, now set to 0.

A value of 0 for a bit indicates INDIVIDUAL, a value of 1 indicates BUNDLED.

7.2.37 Encoding of SET-FILL AREA ASF

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 1 | 1 | 0 | 0 | 0 |      FILL AREA ASF-sub-primitive
-------------------------------
```

Parameters:
- Fill area colour ASF
  In a flag
- Fill area interior style ASF
  In a flag
- Fill area style index ASF
  In a flag

```
-------------------------------
| - | 1 |   |   | 0 | 0 | 0 |
-------------------------------
          |   |   |______________ Fill area style index ASF
          |   |
          |   |__________________ Fill area interior style ASF
          |
          |______________________ Fill area colour ASF
```

Remarks: The undefined bits, b3, b2 and b1, are reserved for future use, now set to 0.

A value of 0 for a bit indicates INDIVIDUAL, a value of 1 indicates BUNDLED.

7.2.38 Encoding of SET-POLYMARKER ASF

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |     SET-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 1 | 1 | 0 | 0 | 1 |     POLYMARKER ASF-sub-primitive
-------------------------------
```

Parameters:
- Polymarker colour ASF
  In a flag
- Marker type ASF
  In a flag
- Marker size ASF
  In a flag

```
-------------------------------
| - | 1 |   |   | 0 | 0 | 0 |
-------------------------------
          |   |   |______________ Marker size ASF
          |   |
          |   |__________________ Marker type ASF
          |
          |______________________ Polymarker colour ASF
```

Remarks: The undefined bits, b3, b2 and b1, are reserved for future use, now set to 0.

A value of 0 for a bit indicates INDIVIDUAL, a value of 1 indicates BUNDLED.

7.2.39 Encoding of SET-TEXT ASF

```
------------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
------------------------------------

------------------------------------
| - | 0 | 1 | 1 | 1 | 0 | 1 | 0 |      TEXT ASF-sub-primitive
------------------------------------
```

Parameters:
- Text colour ASF
  In a flag
- Character height and width ASF
  In a flag
- Character spacing ASF
  In a flag
- Text font and precision ASF
  In a flag

```
------------------------------------
| - | 1 |   |   |   |   | 0 | 0 |
------------------------------------
          |   |   |   |__________ Text font and precision ASF
          |   |   |
          |   |   |______________ Character spacing ASF
          |   |
          |   |__________________ Character height and width ASF
          |
          |______________________ Text colour ASF
```

Remarks: The undefined bits, b2 and b1, are reserved for future use, now set to 0.

A value of 0 for a bit indicates INDIVIDUAL, a value of 1 indicates BUNDLED.

7.2.40 Encoding of SET-POLYLINE REPRESENTATION

```
------------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
------------------------------------

------------------------------------
| - | 0 | 1 | 1 | 1 | 0 | 1 | 1 |      POLYLINE REPRESENTATION-sub-primitive
------------------------------------
```

Parameters:
- Polyline index
  In a three-bit-value
- Line type
  In a three-bit-value

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
       \___________^___________/
              |           |________ Line type
              |
              |____________________ Polyline index
```

- Colour value
  In two three-bit-values
  Colour table and entry number

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
       \___________/\__________/
              |           |________ Entry number in CLUT
              |
              |____________________ Colour table
```

- Line width
  In a size value

```
---------------------------------
| - | 1 | 0 | 0 | 0 | 0 | Y | Y |   First byte
---------------------------------

---------------------------------
| - | 1 | 0 | 0 | 0 | Y | Y | Y |   Second byte
---------------------------------

---------------------------------
| - | 1 | 0 | 0 | 0 | Y | Y | Y |   N-th byte
---------------------------------
```

Remarks:

The <line type> is in the range 0 to 3. The other values are reserved for future use.

The <colour table> is in the range 0 to 3. The other values are for future use.

The undefined bits of <line width> are for future use, now set to 0.

The number of bytes (N) for encoding the <line width> depends on the accuracy value as defined by the CONTROL-DOMAIN RING-primitive (see table 6-2).

7.2.41 Encoding of SET-FILL AREA REPRESENTATION

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 1 | 1 | 1 | 0 | 0 |      FILL AREA REPRESENTATION-sub-primitive
---------------------------------
```

Parameters: - Fill area index
In a three-bit-value

```
---------------------------------
| - | 1 |   |   |   | 0 | 0 | 0 |
---------------------------------
        \___________/
              |____________________ Fill area index
```

- Colour value
In two three-bit-values
Colour table and entry number

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
        \___________/\___________/
              |            |________ Entry number in CLUT
              |
              |____________________ Colour table
```

- Interior style
In a three-bit value

- Style index
In a three-bit-value

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
        \___________/\___________/
              |            |________ Style index
              |
              |____________________ Interior style
```

Remarks: The second three-bit-value in the index byte is reserved for future use, now set to 0.

The <colour table> is in the range 0 to 3. The other values are reserved for future use.

The <interior style> is in the range 0 to 3. The other values are reserved for future use.

The <style index> is not used for interior style HOLLOW and SOLID. For interior style PATTERN it is in the range 0 to 7. For interior style HATCH it is in the range 0 to 5.

7.2.42 Encoding of SET-POLYMARKER REPRESENTATION

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 1 | 1 | 1 | 0 | 1 |      POLYMARKER REPRESENTATION-sub-primitive
---------------------------------
```

Parameters: - Polymarker index
In a three-bit-value

- Marker type
In a three-bit-value

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
        \___________/\__________/
              |            |________ Marker type
              |
              |_____________________ Polymarker index
```

- Colour value
In two three-bit-values
Colour table and entry number

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
        \___________^___________/
              |            |________ Entry number in CLUT
              |
              |_____________________ Colour table
```

- Marker size
In a size value

```
---------------------------------
| - | 1 | 0 | X | X | 0 | Y | Y |     First byte
---------------------------------

---------------------------------
| - | 1 | X | X | X | Y | Y | Y |     Second byte
---------------------------------

---------------------------------
| - | 1 | X | X | X | Y | Y | Y |     N-th byte
---------------------------------
```

Remarks: The <marker type> is in the range 0 to 4. The other values are reserved for future use.

The <colour table> is in the range 0 to 3. The other values are for future use.

The x-value of the <marker size> defines the width of the marker. The y-value defines the height, both in Basic Grid Units. A <marker size> of 0 (all x and y bits 0) gives the default size.

The number of bytes (N) for encoding the <marker size> depends on the accuracy value as defined by the CONTROL-DOMAIN RING-primitive (see table 6-2).

7.2.43 Encoding of SET-TEXT REPRESENTATION

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 1 | 1 | 1 | 1 | 0 |      TEXT REPRESENTATION-sub-primitive
---------------------------------
```

Parameters: - Text index
In a three-bit-value

```
---------------------------------
| - | 1 |   |   |   | 0 | 0 | 0 |
---------------------------------
        \___________/
              |____________________ Text index
```

- Colour value
In two three-bit-values
Colour table and entry number

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
        \___________/\__________/
              |            |________ Entry number in CLUT
              |
              |____________________ Colour table
```

- Text font
In a four-bit-value

- Text precision
In a two bit value

```
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
        \_______/\______________/
            |           |__________ Text font
            |
            |______________________ Text precision
```

Remarks: The second three-bit-value in the index byte is reserved for future use, now set to 0.

The <colour table> is in the range 0 to 3. The other values are reserved for future use.

The <text font> is in the range 0 to 15.

The <text precision> is in the range 0 to 2. Value 3 is reserved for future use.

7.2.44 Encoding of SET-PATTERN REPRESENTATION

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 0 | 1 |      SET-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 1 | 1 | 1 | 1 | 1 |      PATTERN-sub-primitive
---------------------------------
```

Parameters: - Pattern index
In a three-bit-value

```
---------------------------------
| - | 1 |   |   | 0 | 0 | 0 |
---------------------------------
        \___________/
              |____________________ Pattern index
```

- Dimension
defined with:

- Two byte value, m

```
---------------------------------
| - | 1 | m | m | m | m | m | m |
---------------------------------

---------------------------------
| - | 1 | m | m | m | m | m | m |
---------------------------------
```

- Two byte value, n

```
---------------------------------
| - | 1 | n | n | n | n | n | n |
---------------------------------

---------------------------------
| - | 1 | n | n | n | n | n | n |
---------------------------------
```

- Entry number in CLUT
In three-bit-values
One value for each cell

```
 -------------------------------
 | - | 1 |   |   |   |   |   |   |
 -------------------------------
        \__________^__________/
             |         |_________ Second colour
             |
             |___________________ First colour
```

Remarks: The second three-bit-value of the index byte is reserved for future use, now set to 0.

For each cell in the pattern style a colour is defined with one <entry number in CLUT> parameter.
The number of <entry number in CLUT> parameters is m*n, if the number is odd the last <entry number in CLUT> parameter has no meaning.

7.2.45 Encoding of CONTROL-CLEAR

```
 -------------------------------
 | - | 0 | 1 | 1 | 0 | 0 | 1 | 0 |      CONTROL-primitive
 -------------------------------

 -------------------------------
 | - | 0 | 1 | 0 | 0 | 0 | 0 | 0 |      CLEAR-sub-primitive
 -------------------------------
```

Parameters:
- Clear display space
  In a flag
- Clear attributes
  In a flag
- Clear segments
  In a flag
- Clipping flag
  In a flag

```
 -------------------------------
 | - | 1 |   |   |   |   | 0 | 0 |
 -------------------------------
         |   |   |   |___________ Clipping flag
         |   |   |
         |   |   |_______________ Clear segments
         |   |
         |   |___________________ Clear attributes
         |
         |_______________________ Clear display space
```

Remarks: The undefined bits, b2 and b1, are reserved for future use, now set to 0.

A value of 0 for a bit indicates NO clearing. A value of 1 indicates the request to perform the required clearing.

If the <clipping flag> = 0 clipping is OFF, if the <clipping flag> = 1 clipping is ON.

7.2.46 Encoding of CONTROL-WINDOW

-	0	1	1	0	0	1	0	CONTROL-primitive

-	0	1	0	0	0	0	1	WINDOW-sub-primitive

Parameters: - Coordinate list
In Direct mode
The <coordinate list> contains exactly two coordinates, (xmin,ymin) and (xmax,ymax).

Remarks: None

7.2.47 Encoding of CONTROL-VIEWPORT

-	0	1	1	0	0	1	0	CONTROL-primitive

-	0	1	0	0	0	1	0	VIEWPORT-sub-primitive

Parameters: - Coordinate list
In Direct mode
The <coordinate list> contains exactly two coordinates, (xmin,ymin) and (xmax,ymax).

Remarks: None

7.2.48 Encoding of CONTROL-CLIPPING RECTANGLE

-	0	1	1	0	0	1	0	CONTROL-primitive

-	0	1	0	0	0	1	1	CLIPPING RECTANGLE-sub-primitive

Parameters: - Coordinate list
In Direct mode
The <coordinate list> contains exactly two coordinates, (xmin,ymin) and (xmax,ymax).

Remarks: None

7.2.49 Encoding of CONTROL-DOMAIN RING

```
-------------------------------
¦ - ¦ 0 ¦ 1 ¦ 1 ¦ 0 ¦ 0 ¦ 1 ¦ 0 ¦     CONTROL-primitive
-------------------------------

-------------------------------
¦ - ¦ 0 ¦ 1 ¦ 0 ¦ 0 ¦ 1 ¦ 0 ¦ 0 ¦     DOMAIN RING-sub-primitive
-------------------------------
```

Parameters: - Accuracy value
In a three-bit-value

- Angular resolution factor
In a three-bit-value

```
-------------------------------
¦ - ¦ 1 ¦   ¦   ¦   ¦   ¦   ¦   ¦
-------------------------------
        \__________^__________/
             ¦          ¦_______ Angular resolution factor
             ¦
             ¦__________________ Accuracy value
```

- Ring size
In a six-bit-value

```
-------------------------------
¦ - ¦ 1 ¦   ¦   ¦   ¦   ¦   ¦   ¦
-------------------------------
        \_____________________/
                   ¦_____________ Ring size
```

Remarks: This primitive may be used in the following forms defined by its parameters:

a. Two parameters
- <accuracy value>
- <angular resolution factor>

b. All parameters
In the order defined

The <accuracy value> is in the range 0 to 2. The other values are reserved for future use.

The <angular resolution factor> is in the range 0 to 3. The other values are reserved for future use.

The <ring size> is in the range 1 to 63. Value 0 is used to switch to the default basic <ring size> for the defined domain as specified in table 6-3.

7.2.50 Encoding of CONTROL-WAIT

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 1 | 0 |      CONTROL-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 0 | 1 |      WAIT-sub-primitive
-------------------------------
```

Parameters: - Wait time
In a two-byte value

```
-------------------------------
| - | 1 | t | t | t | t | t | t |
-------------------------------

-------------------------------
| - | 1 | t | t | t | t | t | t |
-------------------------------
```

Remarks: The <wait time> is in units of 0.1 seconds, so it can be in the range 0 to 409.5 seconds.

7.2.51 Encoding of CONTROL-DISPLAY PLANE

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 1 | 0 |      CONTROL-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 1 | 0 |      DISPLAY PLANE-sub-primitive
-------------------------------
```

Parameters: - Display plane number to write to
In a three-bit-value

- Display plane number to display
In a three-bit-value

```
-------------------------------
| - | 1 |   |   |   |   |   |   |
-------------------------------
        \__________/\_________/
             |          |_______ Display plane to display
             |
             |__________________ Display plane to write to
```

Remarks: The <display plane numbers> are in the range 0 to 6. Display plane number 7 is reserved for future use.

7.2.52 Encoding of GRAPHIC OBJECT-CREATE

```
 -------------------------------
 | - | 0 | 1 | 1 | 0 | 0 | 1 | 1 |      GRAPHIC OBJECT-primitive
 -------------------------------

 -------------------------------
 | - | 0 | 1 | 0 | 0 | 0 | 0 | 0 |      CREATE-sub-primitive
 -------------------------------
```

Parameters: - Segment name
In long or short form

- Visibility
In a flag

SHORT FORM

```
                 -------------------------------
                 | - | 1 |   | 0 |   |   |   |   |
                 -------------------------------
Visible=0  ____________|       \______________/
Invisible=1                            |
                                       |________  Segment name
```

LONG FORM

```
                 -------------------------------
                 | - | 1 |   | 1 |   |   |   |   |
                 -------------------------------
Visible=0  ____________|       \______________/
Invisible=1                            |
                                       |________  Segment name
                                       |
                          _____________|_______
                         /                     \
                 -------------------------------
                 | - | 1 |   |   |   |   |   |   |
                 -------------------------------
```

Remarks: - The short form is used for <segment names> in the range 0 to 15, the long form for the range 0 to 1023.

7.2.53 Encoding of GRAPHIC OBJECT-CLOSE

```
 -------------------------------
 | - | 0 | 1 | 1 | 0 | 0 | 1 | 1 |      GRAPHIC OBJECT-primitive
 -------------------------------

 -------------------------------
 | - | 0 | 1 | 0 | 0 | 0 | 0 | 1 |      CLOSE-sub-primitive
 -------------------------------
```

Parameters: - None

Remarks: None

7.2.54 Encoding of GRAPHIC OBJECT-RENAME

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 1 | 1 |      GRAPHIC OBJECT-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 1 | 0 |      RENAME-sub-primitive
---------------------------------
```

Parameters: - Old segment name
In long or short form

SHORT FORM

```
---------------------------------
| - | 1 | 0 | 0 |   |   |   |   |
---------------------------------
                 \_____________/
                        |
                        |_________ Old segment name
```

LONG FORM

```
---------------------------------
| - | 1 | 0 | 1 |   |   |   |   |
---------------------------------
                 \_____________/
                        |
                        |_________ Old segment name
                        |
         _______________|_______
        /                       \
---------------------------------
| - | 1 |   |   |   |   |   |   |
---------------------------------
```

- New segment name
In long or short form

SHORT FORM

```
---------------------------------
| - | 1 | 0 | 0 |   |   |   |   |
---------------------------------
                 \_____________/
                        |
                        |_________ New segment name
```

LONG FORM

```
-----------------------------------------
| - | 1 | 0 | 1 |   |   |   |   |
-----------------------------------------
                 \_______________/
                         |
                         |__________ New segment name
                         |
                         |
        _________________|__________
       /                            \
-----------------------------------------
| - | 1 |   |   |   |   |   |   |
-----------------------------------------
```

Remarks: The flag bit (bit 6 in the first name byte) is reserved for future use, now set to 0.

The short form is used for <segment names> in the range 0 to 15, the long form for the range 0 to 1023.

7.2.55 Encoding of GRAPHIC OBJECT-DELETE

```
-----------------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 1 | 1 |     GRAPHIC OBJECT-primitive
-----------------------------------------

-----------------------------------------
| - | 0 | 1 | 0 | 0 | 0 | 1 | 1 |     DELETE-sub-primitive
-----------------------------------------
```

Parameters: - Segment name
In long or short form

SHORT FORM

```
-----------------------------------------
| - | 1 | 0 | 0 |   |   |   |   |
-----------------------------------------
                 \_______________/
                         |
                         |__________ Segment name
```

LONG FORM

```
-------------------------------
| - | 1 | 0 | 1 |   |   |   |   |
-------------------------------
                 \_____________/
                        |
                        |_________  Segment name
                        |
          ______________|_______
         /                      \
-------------------------------
| - | 1 |   |   |   |   |   |   |
-------------------------------
```

Remarks: The flag bit (bit 6 in the first name byte) is reserved for future use, now set to 0.

The short form is used for <segment names> in the range 0 to 15, the long form for the range 0 to 1023.

7.2.56 Encoding of GRAPHIC OBJECT-INSERT

```
---------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 1 | 1 |     GRAPHIC OBJECT-primitive
---------------------------------

---------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 0 | 0 |     INSERT-sub-primitive
---------------------------------
```

Parameters:
- Segment name
  In long or short form
- Transformation
  In a flag

SHORT FORM

```
                      ---------------------------------
                      | - | 1 |   | 0 |   |   |   |   |
                      ---------------------------------
Transformation  _______|        \_______________/
                                        |
                                        |_________  Segment name
```

LONG FORM

```
                 ------------------------------
                 | - | 1 |   | 1 |   |   |   |   |
                 ------------------------------
Transformation ________|     \_______________/
                                    |
                                    |________ Segment name
                                    |
                         ___________|_______
                        /                   \
                 ------------------------------
                 | - | 1 |   |   |   |   |   |   |
                 ------------------------------
```

- Transformation matrix
  In matrix elements
  It contains exactly three matrix elements

```
---------------------------------
| - | 1 | S | X | X | S | Y | Y |    First byte
---------------------------------

---------------------------------
| - | 1 | X | X | X | Y | Y | Y |    Second byte
---------------------------------

---------------------------------
| - | 1 | X | X | X | Y | Y | Y |    N-th byte
---------------------------------
```

Remarks: This primitive may be used in the following forms defined by its parameters:

a. Two parameters
   <segment name>
   <transformation flag>

b. All parameters
   In the order defined

The short form is used for <segment names> in the range 0 to 15, the long form for the range 0 to 1023.

If the <transformation flag> = 0, no-transformation is performed. If the <transformation flag> = 1, the transformation, as specified in this primitive or if absent as specified with the GRAPHIC OBJECT-TRANSFORMATION primitive, is performed.

The <transformation matrix> is encoded in N-bytes per two matrix elements. N is dependent on the accuracy value (N = 2 * (<accuracy value> + 3)), so 3 * N bytes are specified. The first matrix element defines M11 and M21 in respectively the X and Y-bits. The second matrix element defines M12 and M22 in respectively the X and Y-bits. The third matrix element defines M13 and M23 in respectively the X and Y-bits.

7.2.57 Encoding of GRAPHIC OBJECT-TRANSFORMATION

```
-------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 1 | 1 |     GRAPHIC OBJECT-primitive
-------------------------------

-------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 0 | 1 |     TRANSFORMATION-sub-primitive
-------------------------------
```

Parameters: - Segment name
In long or short form

- Reset matrix
In a flag

<u>SHORT FORM</u>

```
                 -------------------------------
                 | - | 1 |   | 0 |   |   |   |   |
                 -------------------------------
Reset matrix  ________|        \_____________/
                                      |
                                      |________ Segment name
```

<u>LONG FORM</u>

```
                 -------------------------------
                 | - | 1 |   | 1 |   |   |   |   |
                 -------------------------------
Reset matrix  ________|        \_____________/
                                      |
                                      |________ Segment name
                                      |
                                      |
                        ______________|_______
                       /                      \
                 -------------------------------
                 | - | 1 |   |   |   |   |   |   |
                 -------------------------------
```

- Transformation matrix
In matrix elements
It contains exactly three matrix elements

```
-------------------------------
| - | 1 | S | X | X | S | Y | Y |     First byte
-------------------------------

-------------------------------
| - | 1 | X | X | X | Y | Y | Y |     Second byte
-------------------------------

-------------------------------
| - | 1 | X | X | X | Y | Y | Y |     N-th byte
-------------------------------
```

Remarks: This primitive may be used in the following forms defined by its parameters:

a. Two parameters
<segment name>
<reset matrix flag>
Only if <reset matrix flag> = 1

b. All parameters
In the order defined

The short form is used for <segment names> in the range 0 to 15, the long form for the range 0 to 1023.

If <reset matrix flag> = 0, the stored matrix for the named segment is not reset, but replaced by the specified one. If <reset matrix flag> = 1, the stored matrix is reset to the default value.

The <transformation matrix> is encoded in N-bytes per two matrix elements. N is dependent on the accuracy value (N = 2 * (<accuracy value> + 3)), so 3 * N bytes are specified. The first matrix element defines M11 and M21 in respectively the X and Y-bits. The second matrix element defines M12 and M22 in respectively the X and Y-bits. The third matrix element defines M13 and M23 in respectively the X and Y-bits.

7.2.58 Encoding of GRAPHIC OBJECT-HIGHLIGHT

```
 ---------------------------------
 | - | 0 | 1 | 1 | 0 | 0 | 1 | 1 |     GRAPHIC OBJECT-primitive
 ---------------------------------

 ---------------------------------
 | - | 0 | 1 | 0 | 0 | 1 | 1 | 0 |     HIGHLIGHT-sub-primitive
 ---------------------------------
```

Parameters: - Segment name
In long or short form

- Highlight
In a flag

SHORT FORM

```
                 ---------------------------------
                 | - | 1 |   | 0 |   |   |   |   |
                 ---------------------------------
 Normal:0 ____________|       \_______________/
Highlight:1                           |
                                      |_________ Segment name
```

LONG FORM

```
                      -----------------------------------------
                      | - | 1 |   | 1 |   |   |   |   |
                      -----------------------------------------
 Normal=0 ____________|        \_______________/
Highlight=1                            |
                                       |________ Segment name
                                       |
                        _______________|________
                       /                        \
                      -----------------------------------------
                      | - | 1 |   |   |   |   |   |   |
                      -----------------------------------------
```

Remarks: The short form is used for <segment names> in the range 0 to 15, the long form for the range 0 to 1023.

7.2.59 Encoding of GRAPHIC OBJECT-VISIBILITY

```
-----------------------------------------
| - | 0 | 1 | 1 | 0 | 0 | 1 | 1 |      GRAPHIC OBJECT-primitive
-----------------------------------------

-----------------------------------------
| - | 0 | 1 | 0 | 0 | 1 | 1 | 1 |      VISIBILITY-sub-primitive
-----------------------------------------
```

Parameters:

- Segment name
  In long or short form

- Visibility
  In a flag

SHORT FORM

```
                      -----------------------------------------
                      | - | 1 |   | 0 |   |   |   |   |
                      -----------------------------------------
Visible=0 ____________|        \_______________/
Invisible=1                            |
                                       |________ Segment name
```

LONG FORM

```
                    --------------------------------
                    ¦ - ¦ 1 ¦    ¦ 1 ¦   ¦   ¦   ¦   ¦
                    --------------------------------
Visible=0 ____________¦      \_______________/
Invisible=1                           ¦
                                      ¦_________ Segment name
                                      ¦
                             _________¦_______
                            /                 \
                    --------------------------------
                    ¦ - ¦ 1 ¦   ¦   ¦   ¦   ¦   ¦   ¦
                    --------------------------------
```

Remarks: The short form is used for <segment names> in the range 0 to 15, the long form for the range 0 to 1023.

APPENDIX A

SHORT NOTE ON B-SPLINE CURVES

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

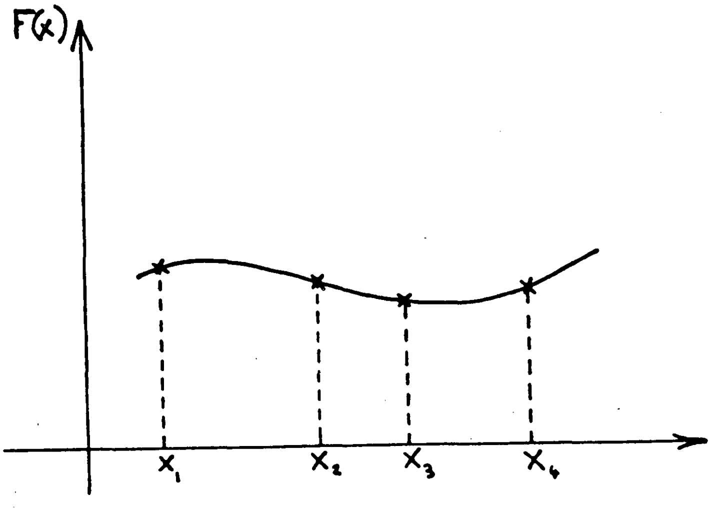


Figure A-1

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 uniform B-splines are used, that are B-splines for which the breakpoints are equally spaced. In the two-dimensional space, a B-spline curve is defined as:

$$P(t) = \sum_{i=1}^{n} P_i N_{im}(t)$$

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

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

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

The corresponding knots are:

$$\frac{P_i + P_{i+1}}{2} \quad \text{and} \quad \frac{P_{i+1} + P_{i+2}}{2}$$

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

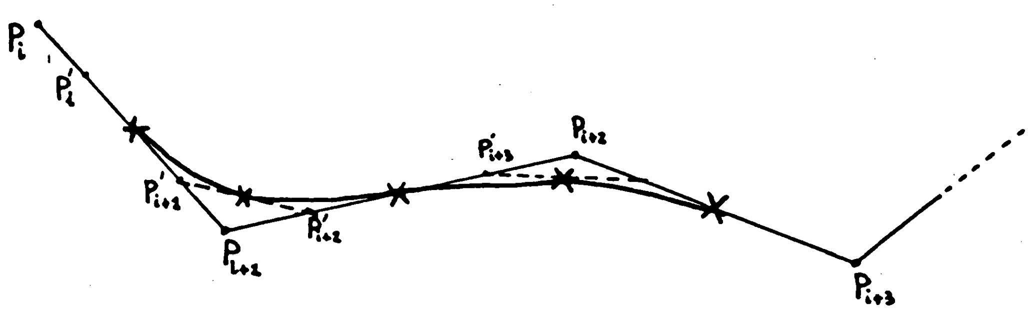


Figure A-2

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

We can apply this theory to quadratic B-splines by replacing Pi, Pi+1 and Pi+2 by the set of four points: P'i, P'i+1, P'i+2 and P'i+3 given by:

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

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

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

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

The new guiding points will produce the same curve as the former ones but they introduce a supplementary knot:

$$\frac{P'_{i+1} + P'_{i+2}}{2}$$

Thus the original curve segment has been divided into two parts. Furthermore the new guiding points are closer to the curve than the former ones (see figure A-2).

By simply repeating this procedure, until the curve segments reach the size of a pixel, the spline curve can be drawn. Only very simple integer arithmetic is needed at each sub-division step (addition and shift). An algorithm of this type is given in [2]. Note that in this algorithm, the given end-points of the curve are no guiding points but knots.

The coordinates as specified in the GDP SPLINE-primitive will be considered as guiding points of a uniform quadratic B-spline curve. The curve can thus easily be generated using the above mentioned sub-division technique.

[1] LANE J.M., R.F. RIESENFELD. "A theoretical Development for the Computer Generation and Display of Piecewise Polynomial Surfaces". I.E.E.E. Trans. on P.A.M.I. - Vol. PAMI - 2, No 1 (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.

PART 3 - PHOTOGRAPHIC DISPLAYS

CONTENTS

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).

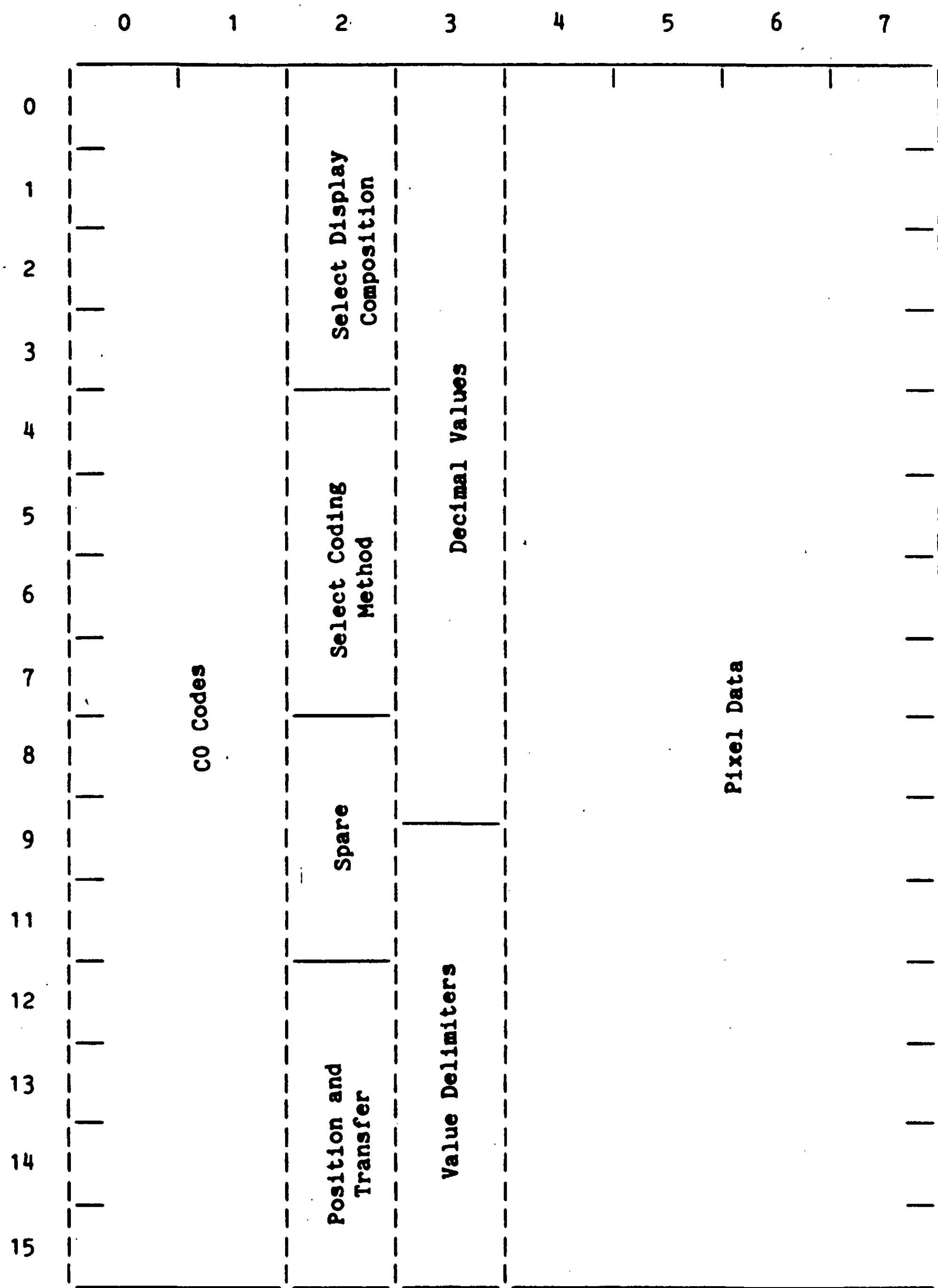


FIGURE 1 PHOTOVIDEOTEX CODE TABLE

	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-Y = 0.701R - 0.587G - 0.114B

B-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-Y)

V* = 0.713(R-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 nominal 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
```

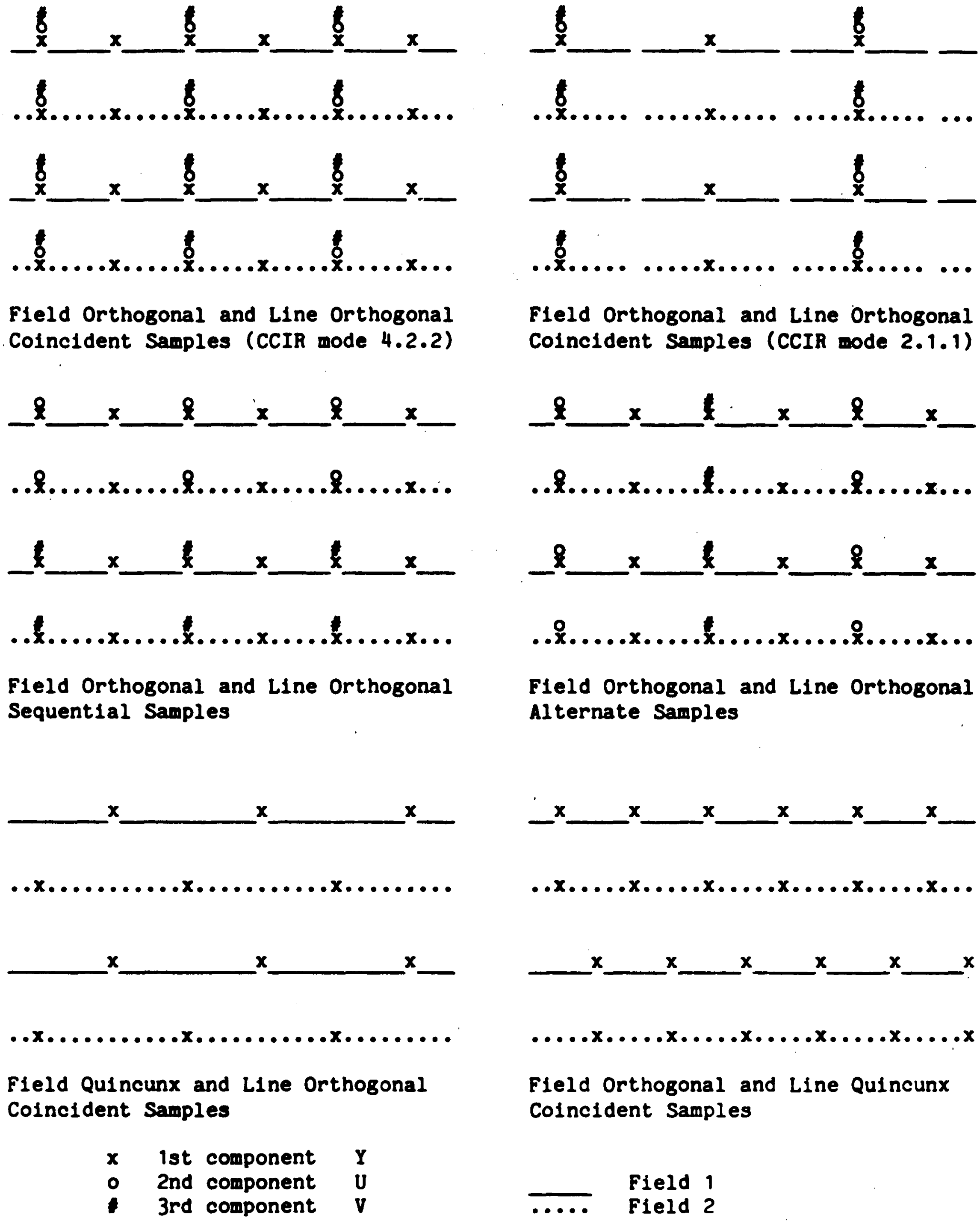


FIGURE 4 SAMPLING STRUCTURES

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:

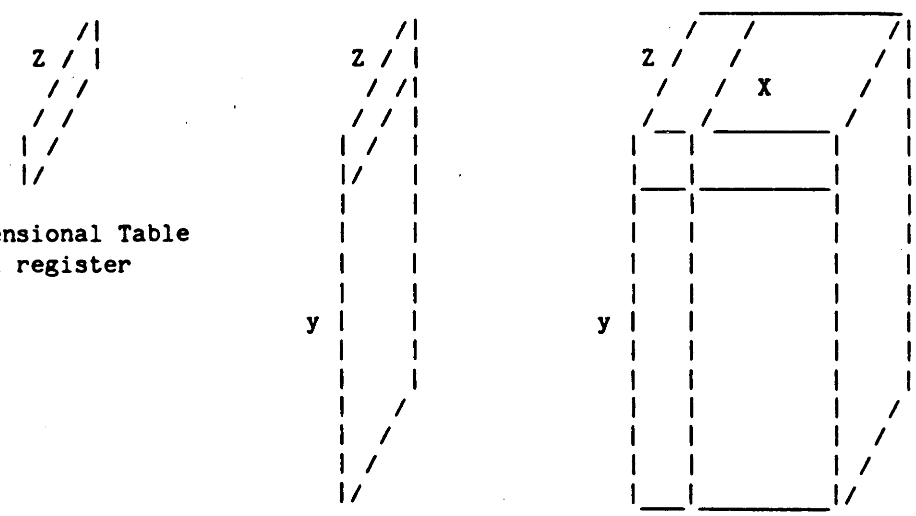
US 3/5 2/0 <SET> <SIZ>

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

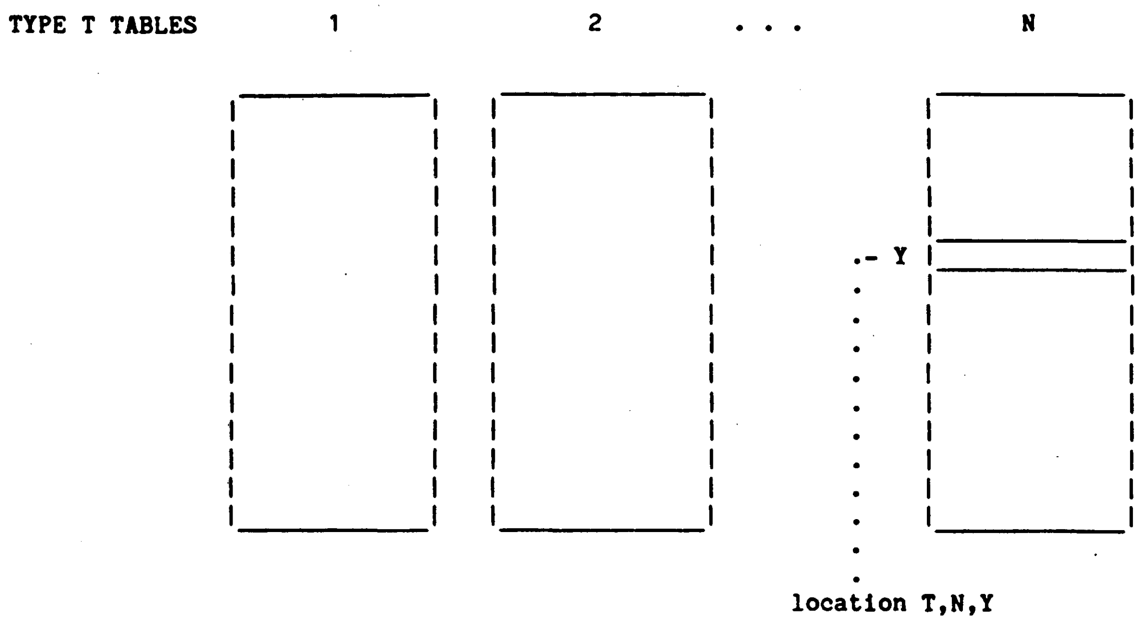
= 2/1 ...3/uT 3/11 ...3/uN 3/11

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.

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

PSDU : 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.

```
PPCI : 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, ..., N-1 is defined as:

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

where u, v = 0, 1, ..., N-1

The inverse transform is:

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

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

C(u) = C(v) = 1 for u, v = 1, 2, ..., 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.

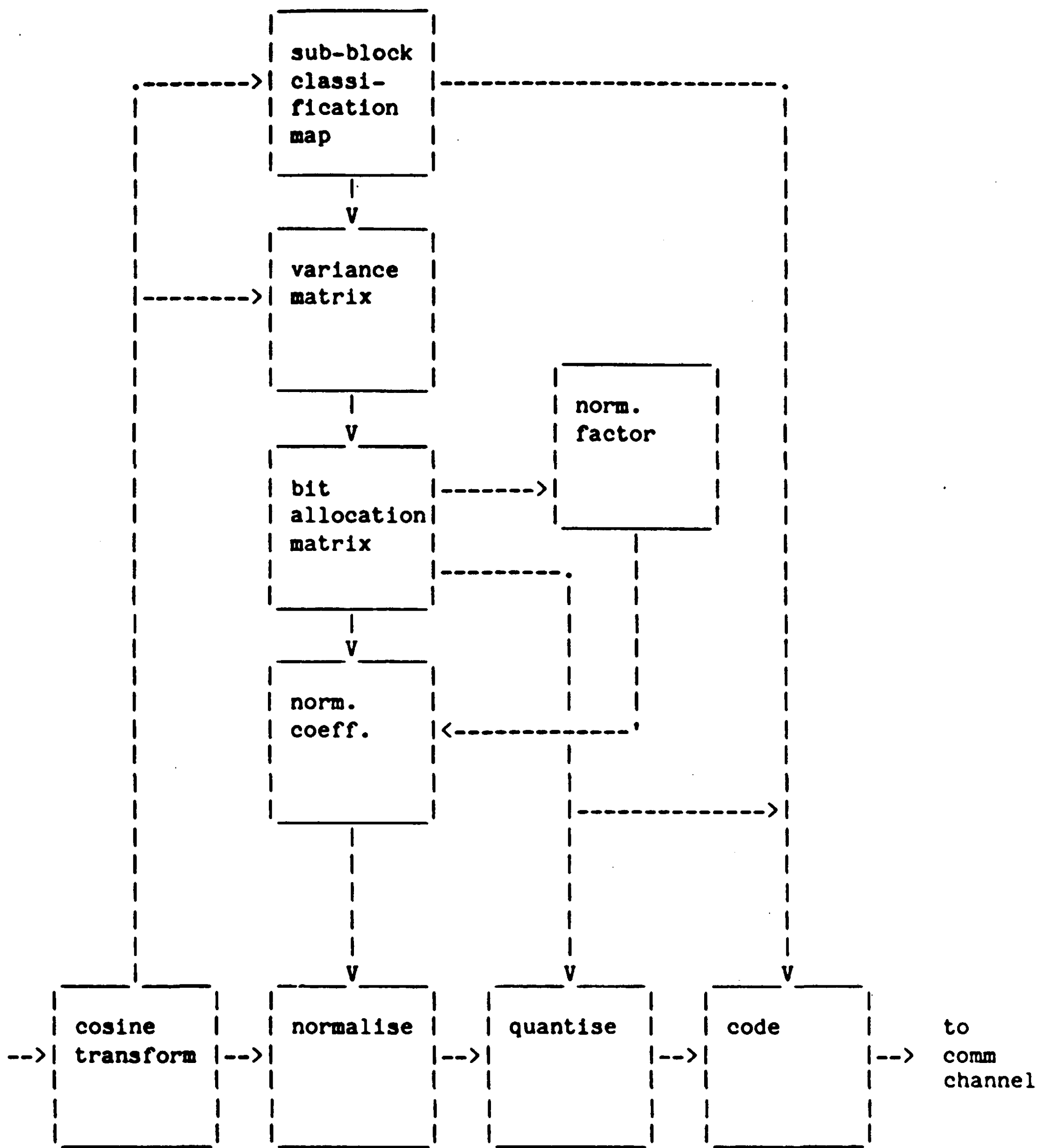


FIGURE B-2 COSINE TRANSFORM ADAPTIVE CODING SYSTEM

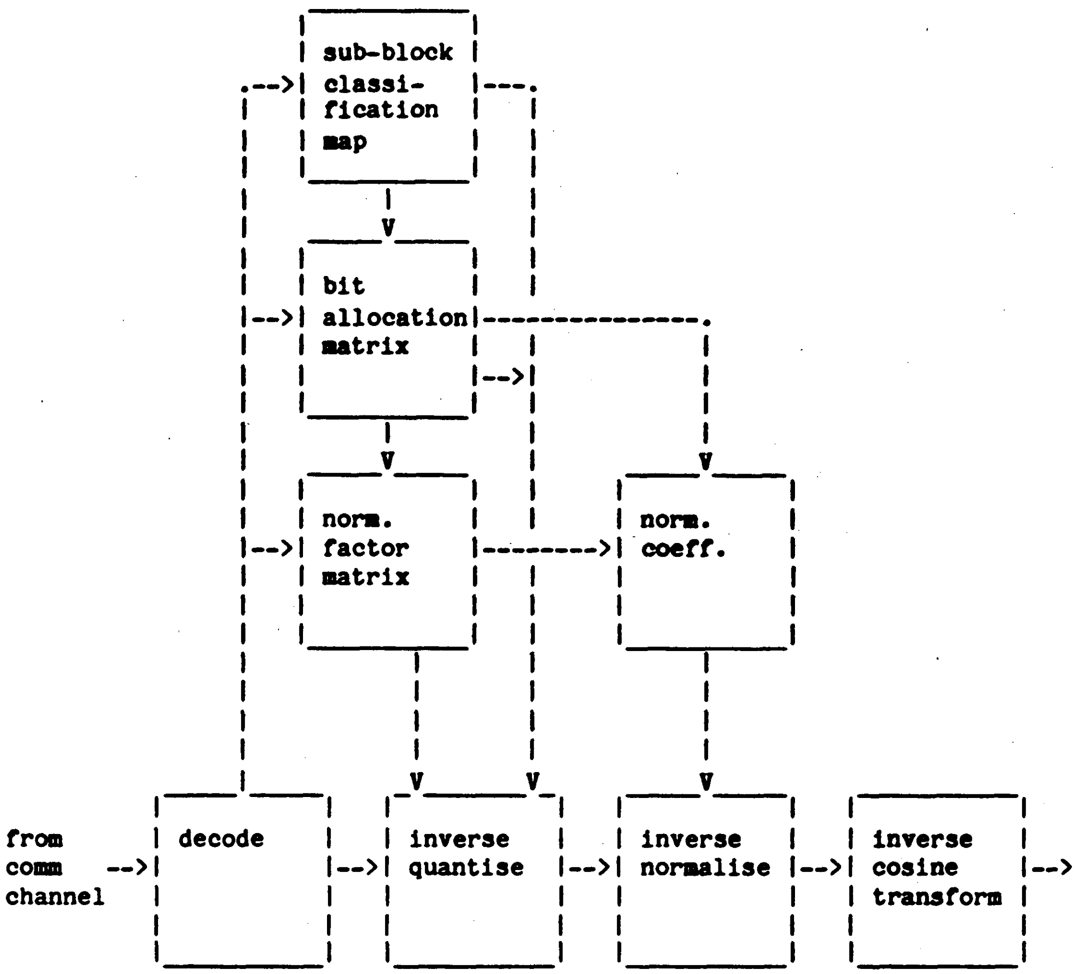


FIGURE B-3 COSINE TRANSFORM ADAPTIVE CODING SYSTEM - DECODE

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

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:

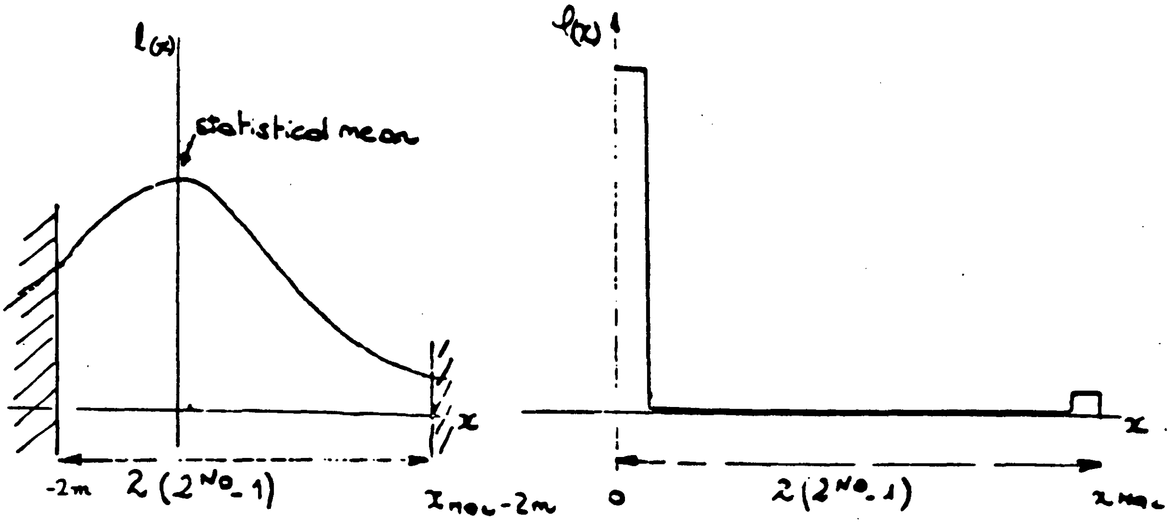
$$x = \frac{F_{m,}{}'}{2(\sim}$$

where $M = E\{f(j,k)\}$

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

X_{128} the highest 8 bit decision level

This guarantees that no dc values are clipped.



Typical Case

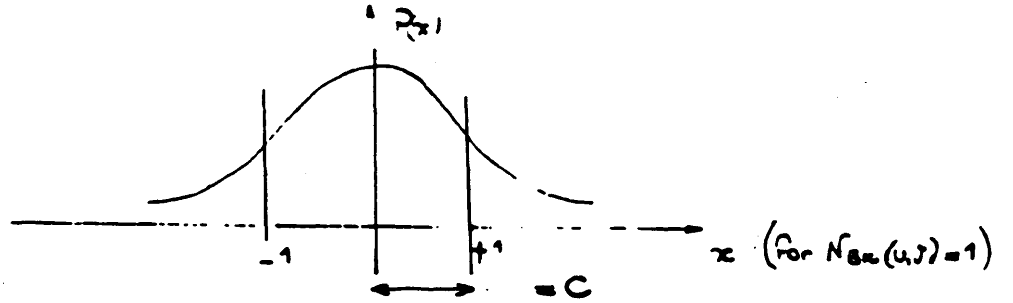
Worst Case
(one sub-block white, the other black)

for u, v ≠ 0

$$x = \frac{F_m\,l\,(u,v)}{C \cdot 2^{N_{bk}(u,v)-1}}$$

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

$$E\left\{Fm,l(u,v)\right\} = 0$$



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:

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

with x in the general form

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

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

$$D = \sum_{i=1}^{2^{N_{Bk}}} \int_{S_{in}(i)}^{S_{in}(i+1)} (S_{in} - S_{out})^2\, p(x).dx$$

which gives

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

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

and

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

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

i	N = 2		N = 4		N = 8	
	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

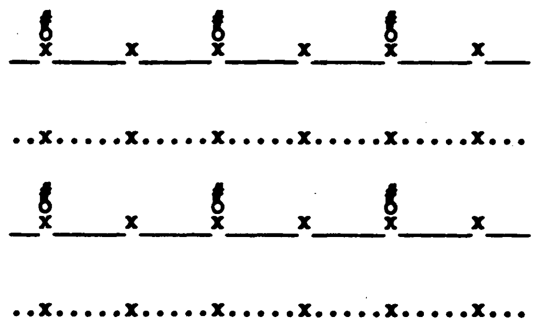
The whole image (two fields) is used.

```
        ____________________________       _
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).



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).

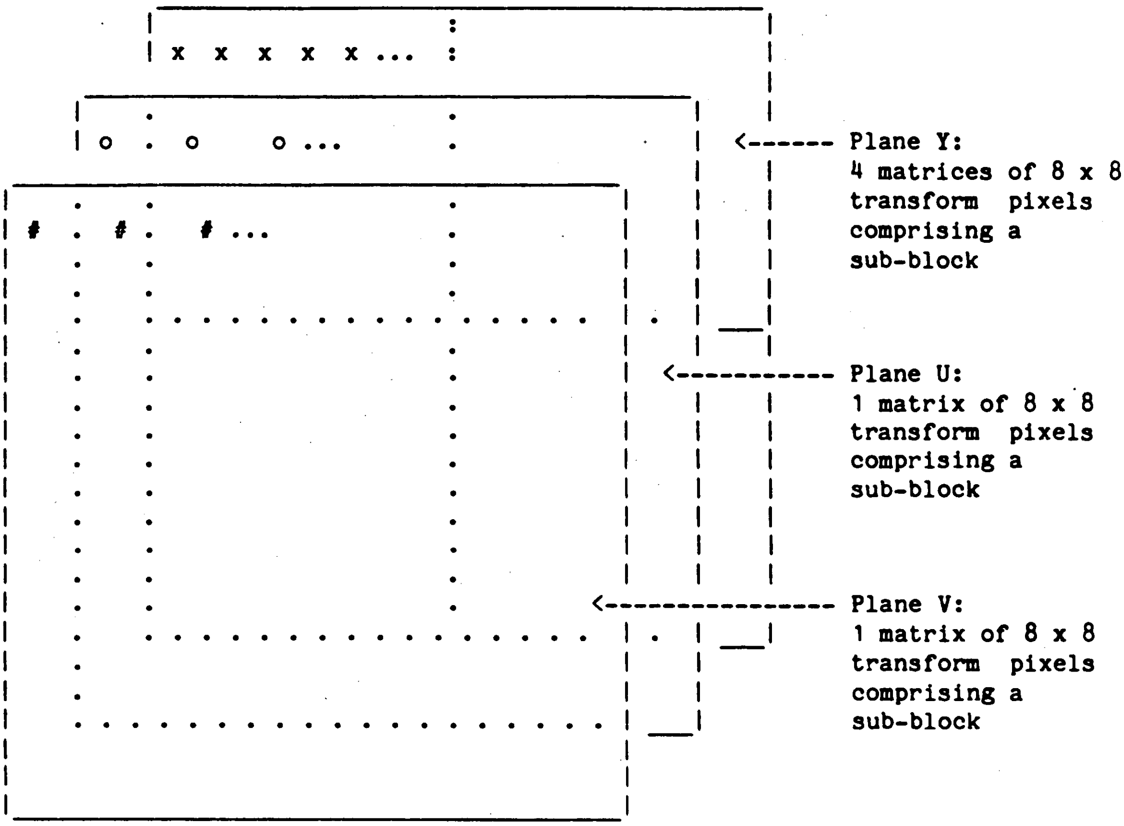


FIGURE B-6 TRANSMITTED TRANSFORM IMAGE STRUCTURE

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.

```
.------------------.   .------------------.   .------------------.
|Y11:Y12        :Y18|   |U11:U12        :U18|   |V11:V12        :V18|
|..:..          .:..|   |..:..          .:..|   |..:..          .:..|
|  :             :  |   |  :             :  |   |  :             :  |
|                   |   |                   |   |                   |
|                   |   |                   |   |                   |
|                   |   |                   |   |                   |
|..:..          .:..|   |..:..          .:..|   |..:..          .:..|
|Y81:Y82        :Y88|   |U81:U82        :U88|   |V81:V82        :V88|
'------------------'   '------------------'   '------------------'
```

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 PPDUs. (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 ---->|AY11              |
(AYii = 1,2 ... 16)     |                  |
                        |                  |
                        |                  |
                        |                  |
                        |                  |
                        |                  |
                        |              AYnn|
                        '------------------'
```

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 PPDUs. (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 PPDU 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 ...	

PART 4 - Define DYNAMICALLY REDIFINABLE CHARACTER SETS

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 ≠ 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.

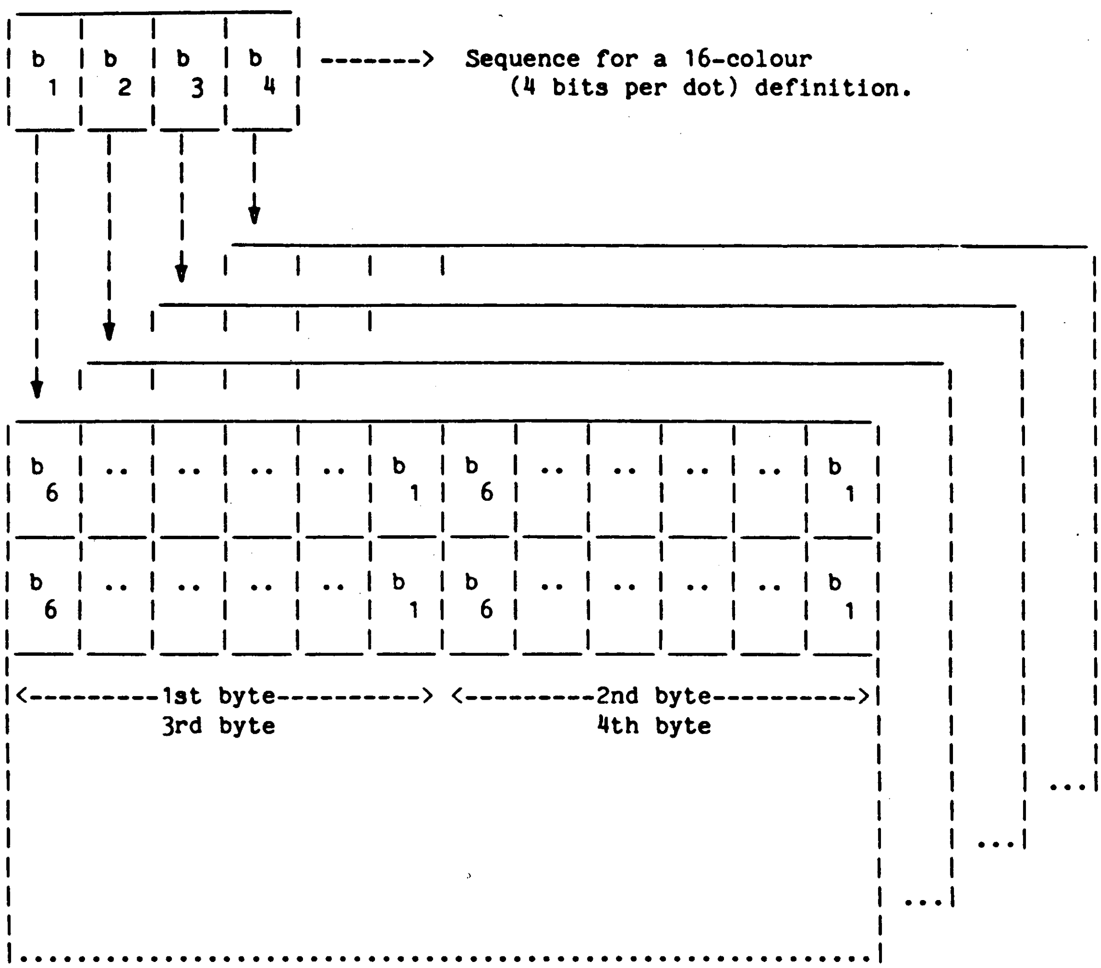


FIGURE 1 DRCS DOWNLOADING FORMAT DEFINITION FOR A 12 x 10 DOT MATRIX

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.

PART 5 - Define COLOUR

CONTENTS

LIST OF FIGURES

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

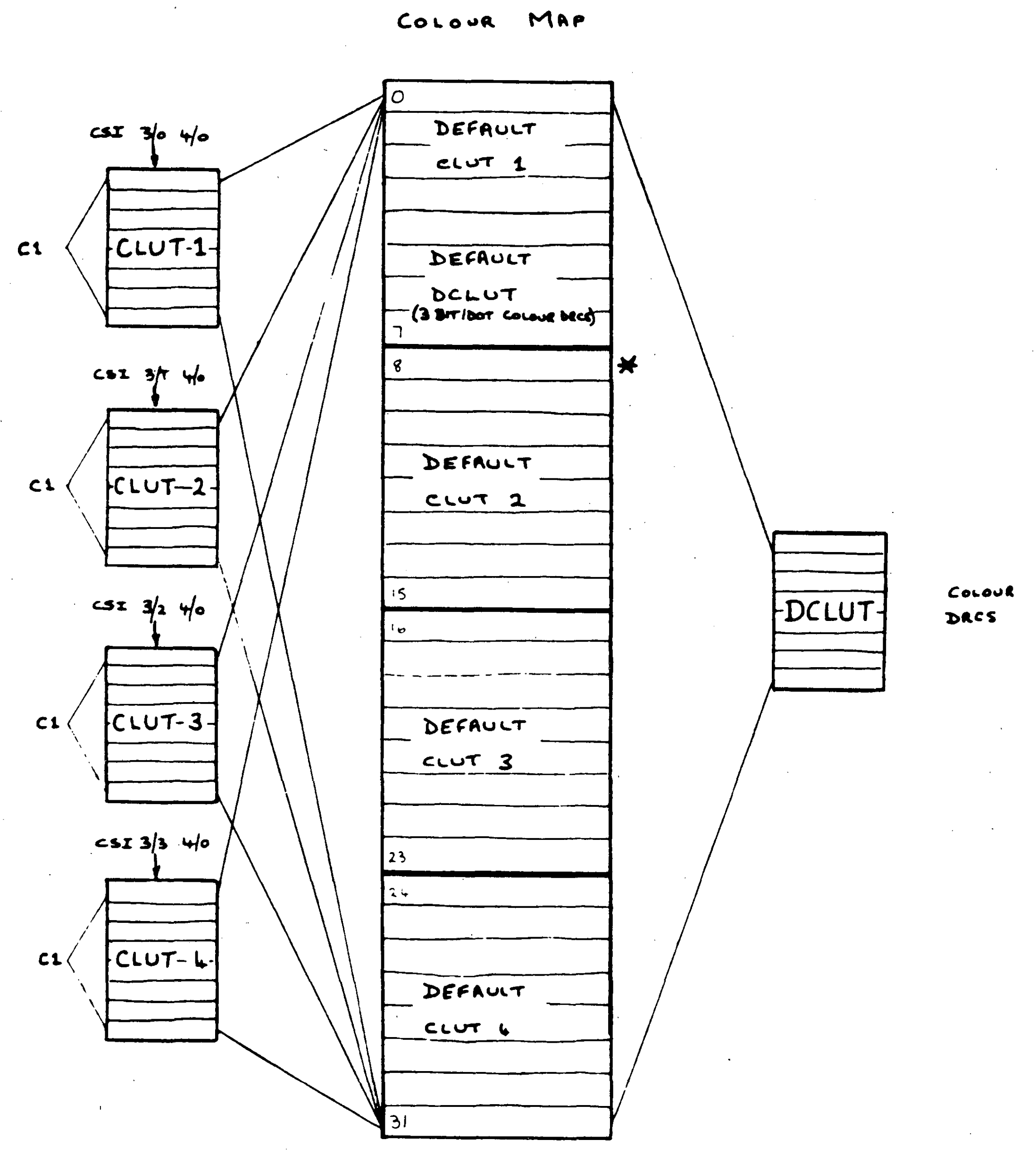
The extension of the colour system is accomplished by providing a number of colour tables of eight colours each. At a given instant only one table can be in use. This table is selected using a CSI sequence (see Part 1 Section 2.3.13). Each table is implemented as a Colour Look Up Table (CLUT) with eight 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.

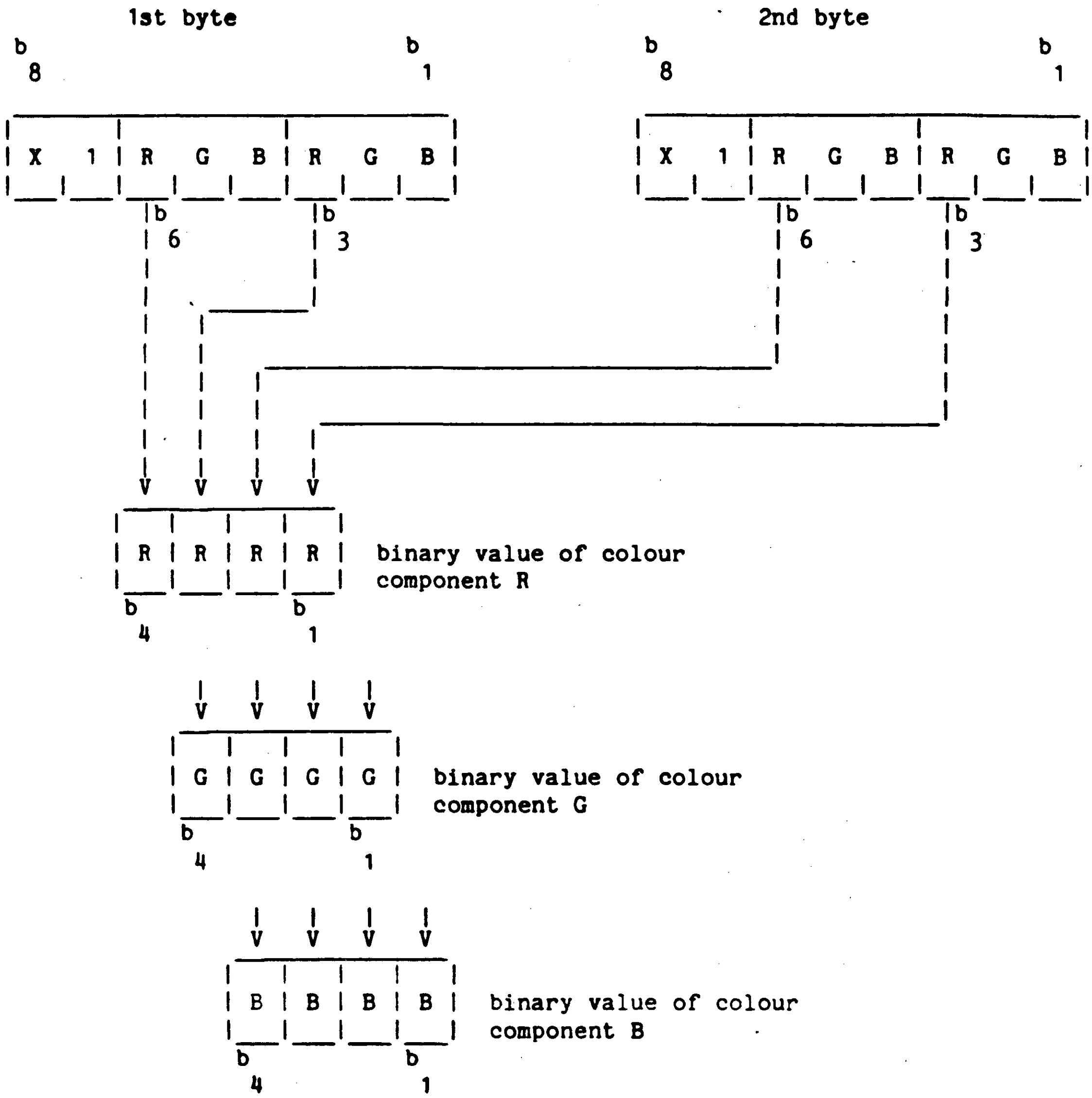
* The coding scheme allows more significant digits to be added if necessary.

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



PART 6 - Define FORMAT

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.

PART 7 - 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. The first byte received in transparent mode specifies the number of bytes that may be received before a return is automatically made to the previous VPDE. The maximum allowable number of characters is 254. This ensures recovery from transparency in a reasonable time in case of any failure. When the US code (start of VPDE code) appears naturally in the data it will be transmitted twice (byte stuffing). The transparent mode byte count is performed on received bytes after stuffing bytes are removed.

Immediate exit from transparent mode is made by starting a new VPDE. This allows for line 24 messages to be displayed. See Figure 1 (Transparent Mode Switching).

Transparent Mode (with US stuffing)

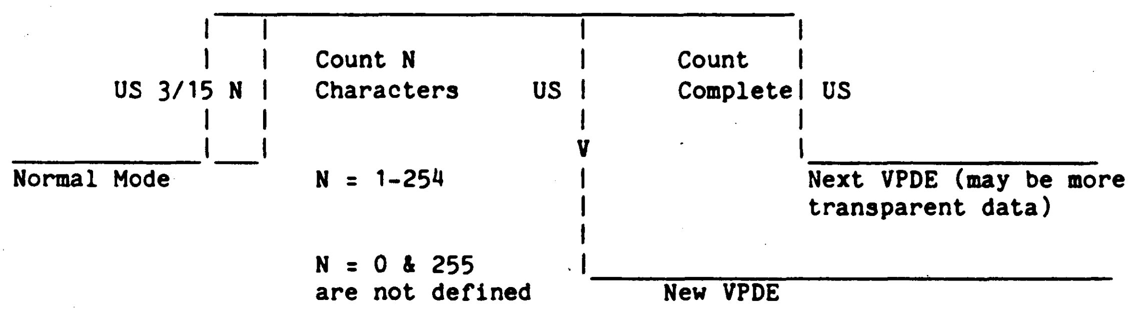


FIGURE 1 TRANSPARENT MODE SWITCHING

PART 8 RESET

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. Alpha-mosaic Displays

All reset operations for alpha-mosaic displays are coded from column 4 of the code table and designate that the data following the reset function is to be interpreted as alpha-mosaic data.

3.1.1. Operation: Reset to defaults

The action of this function is 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 into columns 10 to 15 of the code table.

The format is set to default to 24 rows of 40 characters.

On reset to default the D.D.A. shall be filled with spaces.

The active position is set to the first character position of the first row.

All attributes are set to their default values as described in Part 1 Para 1.5.2.

and a) The serial C1 set is invoked

US 2/15 4/1

or b) The parallel C1 set is invoked

US 2/15 4/2

VIDEOTEX PLDS

3.1.2. Operation: Reset limited defaults

The action of this function is 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 into columns 10 to 15 of the code table.

Format, active position and attributes are not affected.

and a) the serial C1 set is invoked

US 2/15 4/3unction affects the Terminal from the time it is

or b) the parallel C1 set is invoked

US 2/15 4/4

3.1.3. Operation: Service break to row X

This function affects the Terminal from the time it is received until any further US command is received at which time the Terminal resets to the previous state before the new 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. the active position will be stored

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.

The designated C1 set is invoked and the following controls are valid:

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.

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 ignored until the next US.

3.1.4. Operation: Reset to the previous state

Previous display states,including character sets,colours, attributes controls and the active position will be restored.

US 2/15 4/15

This is only valid after a sevice break to row X operation

3.2. GEOMETRIC DISPLAYS

3.2.1. Operation: Reset to defaults

The action of this is the same as the action of the Control Clear Primitive (Part 2 Para 5.1.9.1.).

US 2/15 5/0

For further study

3.3. PHOTOGRAPHIC DISPLAYS

3.3.1. Operation: Reset to defaults

The action of this function is as follows:

Clear all photographic tables

Reset photographic display to transparent.

US 2/15 6/0

For further study

PART 9 TELESOFTWARE

(for further study)

PART 10 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.

2) To determine to which parts of the SRM the terminal conforms.

3) To determine whether the terminal can execute a particular VPDU.

2.0 CODING STRUCTURE

Two VPCIs 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 either a single byte from columns 6 or 7 of the code table, (representing the terminal profile) 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 VPDU, the service transmits the code US 2/0 followed by the header of the VPDU and terminated by US 2/1 to the terminal. The terminal replies with either US 2/0 2/0 indicating it can execute the VPDU or with US 2/0 2/1 indicating it cannot execute the VPDU.

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	Defina Colour (part 5)
4/6	Define Format (part 6)
4/7	Transparent data (part 7)
4/8	Reset (part 8)
4/9	Telesoftware
4/10	Terminal Facility Identifier (part 10)

eg. 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 the SRM is revised) 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.

eg. a terminal which conforms to parts 1, 3 (level 2 photographic), 4,5,6,8 and 10 would transmit;

US 2/0, 4/1, 4/3, 3/2, 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 Profile 1 of Annex C.
6/1 Profile B of Annex C.
6/2 Profile A of Annex C.
6/3 Profile C of Annex C.

3.3 Ability to execute a particular VPDU

The service transmits the VPDU header, (without the US code), and terminates the header with a US 2/1.

eg. 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.

Appendix 1

(to Annex 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 recognized profiles are for further study.

The following definitions apply :

RECOGNIZE 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 recognized (see Annex C to Recommendation T.101, 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 (Annex C, 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 recognized and presented.

1.1.2 Extended Alpha-numeric characters

All Alpha-numeric characters of the repertoire (see Annex C, 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 Annex C, 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, APU, CS and APH shall be recognised and executed.

1.3.2 Extended format effectors

All format effectors shall be recognised and executed. (see Annex C, 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
-Normal size and double height
-Flash and steady
-Conceal Display and Stop Conceal

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 flash state and flash rate are 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-protected.

1.4.5 Full Screen Attribute Controls

The following shall be recognized 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 Annex C, 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 Annex C, 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 Annex C, 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 Annex C, Part 1 Section 3.5.6).

Part 2 Geometric Displays

2.1 Primitives

All primitives in classes C1 and C2 (see Annex C, Part 1 Section 5.2) shall be recognised and executed.

For further study

Part 3 Photographic Displays

3.1 General

All header and transfer units (see Annex C, Part 2 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 Annex C, 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 Annex C, 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 Annex C, 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 also be provided within the system for translating from and to systems having the character matrices of the 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 C, 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 Annex C, 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 Annex C, 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.

5.1 Structure of the Colour Map

5.1.1 Size

The colour map shall consist of 32 colours. (see Annex C, 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 Annex C 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 Annex C, 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 Annex C, Part 5 Section 3.1)

5.2.3 Resolution

A Select Unit Resolution (SUR) of 4 for the colour map shall be executed (see Annex C, 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 Annex C, 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 Annex C, Part 6 Section 2.0).

The wraparound ON and OFF commands (see Annex C, Part 6 Section 2.0) shall be executed.

Part 7 Transparent Data

The whole of the transparent data mode shall be executed (see Annex C, Part 7).

Part 8 Reset

All reset sequences shall be recognised (see Annex C, 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 Annex C, Part 10).

Appendix 2

(to Annex C)

Service Reference Model - Profiles of terminals applying to Data Syntax II

1. Profile 1

This is the profile including the first horizon (see Figure 1). It uses 8 bit coding and satisfies the following parts of the conformance requirements described in Appendix 1 :

Part 0 General

Part 1 Alpha-mosaic 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

Options

These options satisfy the following parts of the conformance requirements described in Appendix 1 :

1.2.1 Geometric displays - Part 2

1.2.2 Photographic displays - Part 3

1.2.3 Transparent data - Part 7

1.2.4 Telesoftware - Part 9

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 Appendix 1.

Part 1 - Alpha mosaic 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 - Only the following accented and special letters are guaranteed :

13 lower case letters : à. è. ù. é. â. ê. î. ô. û. ë. ï. œ. ç.

8 upper case letters : Â. È. É. Î. Ô. Ù. Œ. Ç

1.2.1. Simple mosaic and other character
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 lines.

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 wraparound.

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 Appendix 1 :

Part 1 Alpha-mosaic 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

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 Appendix 1 :

Part 1 Alpha-mosaic 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.

Appendix 3

(to Annex C)

Future horizon

Fig 1c

STANDARD FOR NEW ENHANCED SERVICES AS YET UNFORSEEN 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
	1.1.2	6.2.2
	1.2.2	8
	1.3.2	10
	1.4 to 1.9 (1.4.1 and 1.4.2 simultaneously)	
	4	
	5	

Fig 1a

FLEXIBLE INITIAL BUILDING BLOCKS
(alternatives)

BASIC SERVICE LEVEL

Profile 1	Profile 2	Profile 3	Profile 4

see 2.4(b)

FIG 1 - CEPT RECOMMENDATION ON ENHANCED VIDEOTEX SERVICE AND ITS RELATIONSHIP WITH EXISTING BASIC VIDEOTEX SERVICE AND NEW AS YET UNDEFINED FACILITIES.