



© British Standards Institution. No part of this publication may be photocopied or otherwise reproduced without the prior permission in writing of BSI

Draft for Development

Computer graphics: Metafile for the storage and transfer of picture description information

Part 2. Character encoding (ISO/DIS 8632/2)

Projet à développer

Infographie

Métafichier graphique pour la communication et l'archivage d'information graphique

Partie 2. Codage des caractères

Entwurf zur weiteren Ausarbeitung

Graphische Datenverarbeitung

Metadatei für die Speicherung und Übertragung von Informationen zur Bildbeschreibung

Teil 2. Zeichenkodierung

Contents

	Page		Page
National foreword	2	6.7 Colour specifiers	23
Committees responsible	Back cover	6.8 Colour lists	24
Draft for Development		6.9 String parameters	27
0 Introduction	3	6.10 Enumerated parameters	29
0.1 Purpose of the character encoding	3	6.11 Index parameters	29
0.2 Objectives	3	6.12 Data record parameters	29
0.3 Metafile characteristics	3	7 Character substitution	30
0.4 Relationship to other standards	3	8 Representation of each element	32
0.5 Status of annexes	3	8.1 Delimiter elements	33
1 Scope and field of application	4	8.2 Metafile descriptor elements	34
1.1 Scope	4	8.3 Picture descriptor elements	39
1.2 Field of application	4	8.4 Control elements	41
2 References	5	8.5 Graphical primitive elements	43
3 Notational conventions	6	8.6 Attribute elements	48
3.1 7-bit and 8-bit code tables	6	8.7 Escape elements	56
3.2 Code extension techniques vocabulary	6	8.8 External elements	57
4 Entering and leaving the metafile environment	9	9 Defaults	58
4.1 Implicitly entering the metafile environment	9	10 Conformance	59
4.2 Invoking the CGM coding environment from ISO 2022	9	Annex	
5 Method of encoding opcodes	10	A Formal grammar	62
5.1 Encoding technique of the basic opcode set	10	Tables	
5.2 Extension mechanism	10	1 Opcodes for metafile elements	12
5.3 Opcode assignments	11	2 Huffman code table for incremental mode	21
6 Method of encoding parameters	14	3 CO control set	28
6.1 Basic format	14	4 Character substitution	31
6.2 Bitstream format	15	Figures	
6.3 Coding integers	15	1 The 7-bit code table	7
6.4 Coding real numbers	16	2 The 8-bit code table	8
6.5 Coding VDCs and points	17		
6.6 Coding point list parameters	18		

This publication is not to be regarded as a British Standard.

It is being issued in the Draft for Development series of publications and is of a provisional nature because the subject matter is in the process of ongoing development within the International Standards Committee (ISO/TC 97/SC 21 Information Retrieval, Transfer and Management for Open Systems Interconnection).

It should be applied on this provisional basis, so that information and experience of its practical application may be obtained.

A review of this Draft for Development will be carried out not later than two years after its publication. Notification

of the start of the review period, with a request for the submission of comments from users of this Draft for Development will be made in an announcement in the appropriate issue of *BSI News*. Observations which it is felt should receive attention before the official call for comments will be welcomed.

According to the replies received, the responsible BSI Committee will judge whether the Draft for Development can be converted into a British Standard or what other action should be taken.

National foreword

This Part of DD 107 has been prepared under the direction of the Office and Information Standards Committee.

The primary objective of this Part of DD 107 is to give advance guidance to product and systems designers concerned with computer graphics.

The text of this Part of DD 107 is identical with Draft International Standard ISO/DIS 8632/2-1985, issued by the International Organization for Standardization (ISO).

For ease of reproduction, the ISO text has been reproduced unchanged.

The UK is actively participating through BSI Technical Committee IST/21* in the development of the ISO International Standard. The Technical Committee will review any agreed amendment or addenda to the ISO text and will, where appropriate, update the text of this Draft for Development by means of an amendment or revision. When the ISO draft reaches an appropriate level of agreement and stability, this Part of DD 107 may then be converted to a British Standard.

The four Parts of ISO/DIS 8632 are being published as corresponding Parts of DD 107, as follows.

ISO/DIS 8632/1-1985 is identical with
DD 107 : Part 1 Functional specification

ISO/DIS 8632/2-1985 is identical with
DD 107 : Part 2 Character encoding

ISO/DIS 8632/3-1985 is identical with
DD 107 : Part 3 Binary encoding

ISO/DIS 8632/4-1985 is identical with
DD 107 : Part 4 Clear text encoding

All four Parts are currently at the Draft International Standard stage.

A brochure giving details of existing and projected British Standards publications for Open Systems Interconnection

(OSI) may be obtained from the Sales Department, BSI Linford Wood, Milton Keynes, MK14 6LE, at no charge.

Press releases for new OSI, data communications and computer graphics British Standards publications will be issued to the computing press, and details of all new publications will be recorded in full in *BSI News*.

Terminology and conventions. The text of the Draft International Standard has been approved as suitable for publication as a Draft for Development without deviation. Some terminology and certain conventions are not identical with those used in Drafts for Development; attention is drawn especially to the following.

Wherever the word 'Standard' appears, referring to this Draft for Development, it should be read as 'Draft for Development'.

Cross-references

International Standard	Corresponding British Standard
ISO 646-1983	BS 4730 : 1985 Specification for UK 7 bit data code (Identical, exercising national options)
ISO 7942-1985	BS 6390 : 1985 Specification for a set of functions for computer graphics programming, the Graphical Kernel System (GKS) (Identical)

ISO 2022-1982, to which reference is made in the text, is currently being revised, and this revision may be implemented as a British Standard. BS 4953, which was technically equivalent to ISO 2022-1973, was withdrawn in June 1984.

* Previously OIS/121.

0 Introduction

0.1 Purpose of the Character Encoding

The Character Encoding of the Computer Graphics Metafile (CGM) provides a representation of the Metafile syntax intended for situations in which it is important to minimize the size of the metafile. The encoding uses compact representation of data that is optimized for storage or transfer between computer systems.

If minimizing the processing overhead is more important than data compaction, an encoding such as the Binary Encoding contained in part 3 of this Standard may be more appropriate. If human readability is the most important criterion, an encoding such as the Clear Text Encoding in part 4 of this Standard may be more appropriate.

0.2 Objectives

This encoding was designed with the following objectives:

- a. regular syntax: All elements of the metafile should be encoded in a uniform way so that parsing the metafile is simple;
- b. compactness: The encoding should provide a highly compact metafile, suitable for systems with restricted storage capacity or transfer bandwidth;
- c. extensibility: the encoding should allow for future extensions;
- d. transportability: the encoding should be suitable for use with transport mechanisms designed for character-oriented data based on the standard national character set derived from ISO 646.

0.3 Metafile Characteristics

Each CGM command follows a simple regular syntax. Thus, new commands can be added in a future revision of this Standard such that existing CGM interpreters can recognize (and ignore) the new commands. Also, new operands can be added to existing commands in the future revision of the standard such that existing CGM interpreters can recognize (and ignore) the additional operands.

Each CGM operand follows a simple regular syntax. Operands are variable in length. This permits small values to be represented by the smallest number of bytes.

A certain range of operand values of standard commands have been reserved for private use; the remaining range is either standardized or reserved for future standardization.

0.4 Relationship to Other Standards

The Character Encoding has been developed in collaboration with the ISO subcommittee responsible for character sets and coding (ISO TC97/SC21/WG8), ECMA, and CEPT. The encoding conforms to the rules for code extension specified in ISO 2022 in the category of complete coding system.

For certain elements, the CGM defines value ranges as being reserved for registration. The values and their meanings will be defined using the established procedures (see part 1, 4.11.)

0.5 Status of Annexes

The annexes do not form an integral part of the Standard but are included to provide extra information and explanation.

1 November 1985

1 Scope and Field of Application

1.1 Scope

This part of this Standard specifies a character encoding of the Computer Graphics Metafile. For each of the elements specified in part 1 of this Standard, an encoding is specified.

1.2 Field of Application

This encoding of the Computer Graphics Metafile provides a highly compact representation of the metafile, suitable for applications that require the metafile to be of minimum size and suitable for transmission with character-oriented transmission services.

2 References

The publications referred to in this part of the Standard are as follows.

- ISO 646 Information Processing - 7-bit coded character set for information interchange
- ISO 2022 Information Processing - ISO 7-bit and 8-bit coded character sets - Code Extension techniques
- ISO 2375 Information Processing - Character Set Registration
- ISO 6429 Information Processing - ISO 7-bit and 8-bit coded character sets - Additional control functions for character-imaging devices
- ECMA 96 Graphics Data Syntax for a multiple Workstation Interface
- CEPT Rev. of T/CD 6.1 Videotex Presentation Layer Data Syntax

3 Notational Conventions

3.1 7-Bit and 8-Bit Code Tables

In a 7-bit environment, the code table ("in-use table") is represented as a table of eight columns and sixteen rows. A character, or bit combination, has seven bits, numbered b1 to b7, from least significant to most significant. Bits b7, b6, and b5 address the columns, and bits b4, b3, b2, and b1 address the rows. A "col/row" notation is used in which "col" is the decimal column number, and "row" is the decimal row number. For example, "1/11" refers to the bit combination in column 1, row 11 of the code chart: binary 0011011. Figure 1 shows the 7-bit code table.

In an 8-bit environment, the code table has sixteen columns and sixteen rows. Bits b8, b7, b6, and b5 address the columns, and bits b4, b3, b2, and b1 address the rows.

The same "col/row" notation is used, except that the column numbers are written with two digits. For example, 04/1 represents the 8-bit byte 01000001, whereas 4/1 represents the 7-bit byte 1000001. Figure 2 shows the 8-bit code table.

3.2 Code Extension Techniques Vocabulary

In describing the characters that may occur within string parameters, certain terms imported from other standards (e.g., ISO 2022) are useful. In the context of the CGM, these terms, and the concepts to which they refer, apply only within the string parameters of the TEXT, APPEND TEXT, and RESTRICTED TEXT metafile elements.

3.2.1 C0 Sets

C0 sets are repertoires of 32 control characters intended to occupy columns 0 and 1 of a 7-bit or 8-bit code chart. The meanings of C0 controls within string parameters are described in 6.9.3.

3.2.2 C1 Sets

C1 sets are repertoires of 32 additional controls intended to occupy columns 08 and 09 of an 8-bit code chart, or to be invoked by special 2-byte escape sequences in a 7-bit environment. This CGM encoding reserves the bit combinations 9/8 and 9/12 (ESC 5/8 and ESC 5/12 in 7-bit environment, ESC = 1/11); these shall not be part of the content of string parameters. Other C1 controls from other standards, such as ISO 6429, may be used within string parameters by agreement between the interchanging parties.

3.2.3 G-Sets

The G-sets (G0, G1, G2, G3) are coded character sets of 94 or 96 bit combinations. They may be "invoked into" (caused to occupy) columns 2 through 7 of a 7-bit code chart, or columns 02 through 07 and 10 through 15 of an 8-bit code chart. This encoding of the CGM uses the G0 and G1/G2 sets within string parameters. (CHARACTER SET INDEX designates which character set is to be the G0 set. ALTERNATE CHARACTER SET INDEX designates a character set to be used as both the G1 and G2 sets.) The G3 set may be used within the string parameters of conforming metafiles; this requires selection of the extended 7-bit or extended 8-bit mode in the CHARACTER CODING ANNOUNCER. The CGM does not provide an element to explicitly designate the G3 sets; this may be done within a text string in accordance with ISO 2022, or by other means agreed upon by the interchanging parties.

[ISO page 4]

Notational Conventions

Code Extension Techniques Vocabulary

					0	0	0	0	1	1	1	1	
					0	0	1	1	0	0	1	1	
					0	1	2	3	4	5	6	7	
Bits	b ₄	b ₃	b ₂	b ₁	ROW \ COLUMN	THE C0 SET A G-SET OF 94 OR 96 BIT COMBINATIONS							
	0	0	0	0	0								
	0	0	0	1	1								
	0	0	1	0	2								
	0	0	1	1	3								
	0	1	0	0	4								
	0	1	0	1	5								
	0	1	1	0	6								
	0	1	1	1	7								
	1	0	0	0	8								
	1	0	0	1	9								
	1	0	1	0	10								
	1	0	1	1	11								
	1	1	0	0	12								
	1	1	0	1	13								
	1	1	1	0	14								
	1	1	1	1	15								

Figure 1. The 7-bit code table.

	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
0	C0 SET		02/0								10/0					
1																
2																
3																
4																
5																
6																
7																
8																
9																
10																
11																
12																
13																
14																
15								07/15							15/15	

Figure 2. The 8-bit code table.

4 Entering and Leaving the Metafile Environment

4.1 Implicitly Entering the Metafile Environment

The CGM coding environment may be entered implicitly, by agreement between the interchanging parties. This is suitable only if there is not to be any interchange with services using other coding techniques.

4.2 Invoking the CGM Coding Environment From ISO 2022

For interchange with services using the code extension techniques of ISO 2022, the CGM coding environment shall be invoked from ISO 2022 environment by the following escape sequence:

ESC 2/5 F

where ESC is the bit combination 1/11, and F refers to a bit combination that will be assigned by the ISO Registration Authority for Code Extensions and will be substituted in this Standard prior to its publication.

The first bit combination occurring after this escape sequence will then represent the opcode of a CGM metafile element.

After the end of one or more metafiles (i.e., after the END METAFILE element or between pictures), the following escape sequence may be used to return to the ISO 2022 coding environment:

ESC 2/5 4/0

This not only returns to the ISO 2022 coding environment, but also restores the designation and invocation of coded character sets to the state that existed prior to entering the CGM coding environment with the ESC 2/5 F sequence. (The terms "designation" and "invocation" are defined in ISO 2022.)

5 Method of Encoding Opcodes

Each metafile element is composed of one opcode and parameters as required. The opcodes are coded as a sequence of bit combinations from columns 2 and 3 of the code chart. The encoding technique supplies:

- the basic opcode set;
- extension opcode sets.

5.1 Encoding Technique of the Basic Opcode Set

The basic opcode set consists of single-byte and double-byte opcodes. Single-byte opcodes are from column 2 of the code chart. Bits b4 to b1 are used to encode the opcode. The format is as follows:

```

b8          b1
+--+-----+-----+
|X|0 1 0|b b b b|
+--+-----+-----+

```

For double-byte opcodes the first byte is from column 3 and the second byte is from column 2 or 3 of the code chart. Bits b4 to b1 of the first byte and bits b5 to b1 of the second byte are used to encode the opcode:

```

b8          b1  b8          b1
+--+-----+-----+ +--+-----+-----+
|X|0 1 1|b b b b| |X|0 1|b b b b b|
+--+-----+-----+ +--+-----+-----+

```

The bit combination 3/15, the EXTEND OPCODE SPACE (EOS) allows extension of the basic opcode space (see 5.2).

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

- 16 single-byte opcodes (from column 2);
- $15 \times 32 = 480$ double-byte opcodes (first byte from column 3 except bit combination 3/15, second byte from column 2 or 3).

5.2 Extension Mechanism

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

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

Opcode format	Extension codes	Basic opcode set codes
1	$\langle \text{EOS} \rangle \dots \langle \text{EOS} \rangle$ ----- n times	$\langle 2/x \rangle$
2	$\langle \text{EOS} \rangle \dots \langle \text{EOS} \rangle$ ----- n times	$\langle 3/y \rangle \langle 2/z \rangle$

Method of Encoding Opcodes

Extension Mechanism

3

<EOS>...<EOS>

<3/y> <3/z>

|-----|

|
n times

<EOS> = 3/15

x = 0,1,...,15

y = 0,1,...,14

z = 0,1,...,15

n = 0,1,.....

n = 0 selects the basic opcode set.

n = 1 selects the first extension opcode set.

n = N selects the N-th extension opcode set.

The number of opcodes supplied by this encoding technique (basic opcode set plus extension opcode sets) is $496(n+1)$, where n is the number of extension sets. (Each extension set has 496 opcodes -- 16 single-byte opcodes plus 480 double-byte codes.)

5.3 Opcode Assignments

[ISO page 9]

Documents

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1

Opcode Assignments

Method of Encoding Opcodes

TABLE 1. Opcodes for Metafile Elements.

Opcode	7-Bit Coding		8-Bit Coding	
BEGIN METAFILE opcode	3/0	2/0	03/0	02/0
END METAFILE opcode	3/0	2/1	03/0	02/1
BEGIN PICTURE opcode	3/0	2/2	03/0	02/2
BEGIN PICTURE BODY opcode	3/0	2/3	03/0	02/3
END PICTURE opcode	3/0	2/4	03/0	02/4
METAFILE VERSION opcode	3/1	2/0	03/1	02/0
METAFILE DESCRIPTION opcode	3/1	2/1	03/1	02/1
VDC TYPE opcode	3/1	2/2	03/1	02/2
INTEGER PRECISION opcode	3/1	2/3	03/1	02/3
REAL PRECISION opcode	3/1	2/4	03/1	02/4
INDEX PRECISION opcode	3/1	2/5	03/1	02/5
COLOUR PRECISION opcode	3/1	2/6	03/1	02/6
COLOUR INDEX PRECISION opcode	3/1	2/7	03/1	02/7
MAXIMUM COLOUR INDEX opcode	3/1	2/8	03/1	02/8
COLOUR VALUE EXTENT opcode	3/1	2/9	03/1	02/9
METAFILE ELEMENT LIST opcode	3/1	2/10	03/1	02/10
BEGIN METAFILE DEFAULTS REPLACEMENT opcode	3/1	2/11	03/1	02/11
END METAFILE DEFAULTS REPLACEMENT opcode	3/1	2/12	03/1	02/12
FONT LIST opcode	3/1	2/13	03/1	02/13
CHARACTER SET LIST opcode	3/1	2/14	03/1	02/14
CHARACTER CODING ANNOUNCER opcode	3/1	2/15	03/1	02/15
SCALING MODE opcode	3/2	2/0	03/2	02/0
COLOUR SELECTION MODE opcode	3/2	2/1	03/2	02/1
LINE WIDTH SPECIFICATION MODE opcode	3/2	2/2	03/2	02/2
MARKER SIZE SPECIFICATION MODE opcode	3/2	2/3	03/2	02/3
EDGE WIDTH SPECIFICATION MODE opcode	3/2	2/4	03/2	02/4
VDC EXTENT opcode	3/2	2/5	03/2	02/5
BACKGROUND COLOUR opcode	3/2	2/6	03/2	02/6
VDC INTEGER PRECISION opcode	3/3	2/0	03/3	02/0
VDC REAL PRECISION opcode	3/3	2/1	03/3	02/1
AUXILIARY COLOUR opcode	3/3	2/2	03/3	02/2
TRANSPARENCY opcode	3/3	2/3	03/3	02/3
CLIP RECTANGLE opcode	3/3	2/4	03/3	02/4
CLIP INDICATOR opcode	3/3	2/5	03/3	02/5
POLYLINE opcode	2/0		02/0	
DISJOINT POLYLINE opcode	2/1		02/1	
POLYMARKER opcode	2/2		02/2	
TEXT opcode	2/3		02/3	
RESTRICTED TEXT opcode	2/4		02/4	
APPEND TEXT opcode	2/5		02/5	
POLYGON opcode	2/6		02/6	
POLYGON SET opcode	2/7		02/7	
CELL ARRAY opcode	2/8		02/8	
GENERALIZED DRAWING PRIMITIVE opcode	2/9		02/9	
RECTANGLE opcode	2/10		02/10	
CIRCLE opcode	3/4	2/0	03/4	02/0
CIRCULAR ARC 3 POINT opcode	3/4	2/1	03/4	02/1

[ISO page 10]

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1.

Documents

Document Sharing Hub

Method of Encoding Opcodes

Opcode Assignments

CIRCULAR ARC 3 POINT CLOSE opcode	3/4	2/2	03/4	02/2
CIRCULAR ARC CENTRE opcode	3/4	2/3	03/4	02/3
CIRCULAR ARC CENTRE CLOSE opcode	3/4	2/4	03/4	02/4
ELLIPSE opcode	3/4	2/5	03/4	02/5
ELLIPTICAL ARC opcode	3/4	2/6	03/4	02/6
ELLIPTICAL ARC CLOSE opcode	3/4	2/7	03/4	02/7
LINE BUNDLE INDEX opcode	3/5	2/0	03/5	02/0
LINE TYPE opcode	3/5	2/1	03/5	02/1
LINE WIDTH	3/5	2/2	03/5	02/2
LINE COLOUR opcode	3/5	2/3	03/5	02/3
MARKER BUNDLE INDEX opcode	3/5	2/4	03/5	02/4
MARKER TYPE opcode	3/5	2/5	03/5	02/5
MARKER SIZE opcode	3/5	2/6	03/5	02/6
MARKER COLOUR opcode	3/5	2/7	03/5	02/7
TEXT BUNDLE INDEX opcode	3/5	3/0	03/5	03/0
TEXT FONT INDEX opcode	3/5	3/1	03/5	03/1
TEXT PRECISION opcode	3/5	3/2	03/5	03/2
CHARACTER EXPANSION FACTOR opcode	3/5	3/3	03/5	03/3
CHARACTER SPACING opcode	3/5	3/4	03/5	03/4
TEXT COLOUR opcode	3/5	3/5	03/5	03/5
CHARACTER HEIGHT opcode	3/5	3/6	03/5	03/6
CHARACTER ORIENTATION opcode	3/5	3/7	03/5	03/7
TEXT PATH opcode	3/5	3/8	03/5	03/8
TEXT ALIGNMENT opcode	3/5	3/9	03/5	03/9
CHARACTER SET INDEX opcode	3/5	3/10	03/5	03/10
ALTERNATE CHARACTER SET INDEX opcode	3/5	3/11	03/5	03/11
FILL BUNDLE INDEX opcode	3/6	2/0	03/6	02/0
INTERIOR STYLE opcode	3/6	2/1	03/6	02/1
FILL COLOUR opcode	3/6	2/2	03/6	02/2
HATCH INDEX opcode	3/6	2/3	03/6	02/3
PATTERN INDEX opcode	3/6	2/4	03/6	02/4
EDGE BUNDLE INDEX opcode	3/6	2/5	03/6	02/5
EDGE TYPE opcode	3/6	2/6	03/6	02/6
EDGE WIDTH opcode	3/6	2/7	03/6	02/7
EDGE COLOUR opcode	3/6	2/8	03/6	02/8
EDGE VISIBILITY opcode	3/6	2/9	03/6	02/9
FILL REFERENCE POINT opcode	3/6	2/10	03/6	02/10
PATTERN TABLE opcode	3/6	2/11	03/6	02/11
PATTERN SIZE opcode	3/6	2/12	03/6	02/12
COLOUR TABLE opcode	3/6	3/0	03/6	03/0
ASPECT SOURCE FLAGS opcode	3/6	3/1	03/6	03/1
ESCAPE opcode	3/7	2/0	03/7	02/0
DOMAIN RING opcode	3/7	3/0	03/7	03/0
MESSAGE opcode	3/9	2/1	03/7	02/1
APPLICATION DATA opcode	3/7	2/1	03/9	02/1

[ISO page 11]

Documents

This publication is a Draft for Development and is therefore not to be regarded as a British Standard. See page 1.

Document Sharing Hub

6 Method of Encoding Parameters

The parameter part of a CGM element may contain one or more parameters, each parameter consisting of one or more bytes.

All parameters are coded in columns 4 through 7. (However, the coded representation of a 'string' parameter may include bit combinations from other columns of the code table - see the description of string parameters in 6.9). The general format of a parameter byte is:

```

      b8                b1
    +--+---+-----+
    |X|1|b b b b b|
    +--+---+-----+

```

The "X" (bit b8) is the parity bit (or omitted bit) in a 7-bit environment. In an 8-bit environment it is 0. Bit b7 is the parameter flag.

Except for 'string' and 'data record' parameters all parameters are encoded using one or both of two formats, Basic format or Bitstream format.

6.1 Basic Format

Each Basic format parameter is coded as a sequence of one or more bytes, structured as follows:

```

      b8                b1
    +--+---+---+-----+
    |X|1|e|s|b b b b|  first byte
    +--+---+---+-----+

```

⋮

```

      b8                b1
    +--+---+---+-----+
    |X|1|e|b b b b b|  last byte
    +--+---+---+-----+

```

The "X" (bit b8) is the parity bit (or omitted bit) in a 7-bit environment. In an 8-bit environment it is 0. Bit b7 is the parameter flag.

"e" (b6 of each byte) is the extension flag. For single byte parameters, the extension flag is 0. In multi-byte parameters, the extension flag is 1 in all bytes except the last byte, where it is 0.

Bits b5 through b1 are the data bits of the parameter.

"s" is the sign bit; if equal to 0 then the integer is non-negative and if equal to 1 then the integer is negative. The number zero shall always be coded as "plus zero" 4/0. The "minus zero" coding is reserved for special usage (see 6.6.3).

The Basic format is used to encode:

- a. enumerated types (E);
- b. colour indices (CI);
- c. indices other than colour indices (IX);
- d. integers (I);
- e. real numbers (R);

Method of Encoding Parameters

Basic Format

f. non-incremental coordinates.

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

6.2 Bitstream Format

Each Bitstream format parameter is encoded as a sequence of one or more bytes, structured as follows:

```

b8                      b1
+---+-----+
|X|1|b b b b b b| first byte
+---+-----+

```

```

b8                      b1
+---+-----+
|X|1|b b b b b b| last byte
+---+-----+

```

The "X" is the parity bit (or omitted bit) in a 7-bit environment. In an 8-bit environment it is 0. Bit b7 is the parameter flag.

Bits b6 through b1 are the data bits of the parameter.

The Bitstream format is used to encode:

- a. incremental mode coordinates (see 6.6.2);
- b. colour direct (see 6.8);
- c. colour index lists (see 6.8).

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

The end of a Bitstream format parameter cannot be derived from the Bitstream format itself (the format is not self-delimiting). Instead,

- for incremental mode coordinates, the end of the data (which identifies the end of the Bitstream format parameter) is identified by the <End of Block> code;
- for colour index lists, the number of bits needed to encode the colour index list (which identifies the end of the Bitstream format parameter) is set by the COLOUR INDEX PRECISION element, or by the LOCAL COLOUR PRECISION parameter (for those elements which contain such a parameter);
- for colour direct data, the number of bits needed to encode the data (which identifies the end of the Bitstream format operand) is set by the COLOUR PRECISION element, or by the LOCAL COLOUR PRECISION parameter (for those elements which contain such a parameter).

6.3 Coding Integers

Integers are coded as sequences of bytes in the range from 4/0 to 7/15 in the Basic format. If a byte is from columns 4 to 5 of the code table, it is either the last byte in the integer's coded representation or it is a single-byte integer. A multi-byte integer begins with a byte from columns 6 to 7 of the code table.

The structure of integer parameters is as illustrated for Basic format (see 6.1). "bbb..." are bits representing the magnitude of the integer. The most significant part of the parameter is coded in the first byte. The least significant part of the parameter is coded in the last byte.

Coding Integers

Method of Encoding Parameters

Any integer can be coded with leading most significant bits which are all zero. For example, 4/3 and 6/0 6/0 4/3 are both valid codings for the integer "+3"; however, efficient metafile generators should avoid such redundant codings.

The size of integer parameters is limited by the current INTEGER PRECISION value.

Integers in the range of -15 to +15 can be coded as single bytes.

(integer: +1) = 4/1	(integer: -1) = 5/1
(integer: +15) = 4/15	(integer: -15) = 5/15

Larger integers require more bytes.

(integer: +16) = 6/0 5/0	(integer: -16) = 7/0 5/0
(integer: +1034) = 6/1 6/0 4/10	(integer: -1034) = 7/1 6/0 4/10

6.4 Coding Real Numbers

Each real number is coded as an integer mantissa followed by an optional exponent, both coded in the Basic format. The exponent is the power of two by which the integer mantissa is to be multiplied.

The exponent may be implicitly defined as a default exponent, which is then omitted in the Real format, or the exponent may be coded explicitly as the second part of the Real format:

<real format> = <mantissa part> [<exponent part>]

Depending on the "exponent allowed" parameter of the REAL PRECISION element, one of the bits in the first byte of the mantissa tells whether the exponent follows. If the "exponent follows" bit in the mantissa is zero, or if REAL PRECISION has specified that there is to be no "exponent follows" bit, then the exponent is omitted and a default value is assumed which is set by another parameter of the REAL PRECISION element.

Note that the coding of real VDC coordinates is controlled by VDC REAL PRECISION, not by REAL PRECISION, and that the rules for default exponents in real VDC coordinates are slightly different (see 6.5 and 6.6).

The mantissa is an integer which is coded in the Basic format. The first byte takes one of two forms, depending on whether or not REAL PRECISION has specified that an "exponent follows" bit is to be included. The format is as follows (first byte):

b8	b1	
+--+--+--+--+-----+		{if explicit exponent allowed = 'allowed',
X 1 e s p b b b		i.e. the exponent-follows bit is present}
+--+--+--+--+-----+		

or

b8	b1	
+--+--+--+--+-----+		{if explicit exponent allowed = 'forbidden',
X 1 e s b b b b		i.e. there is no exponent-follows bit}
+--+--+--+--+-----+		

b8	b1	
+--+--+--+--+-----+		
X 1 e b b b b b		{last byte}
+--+--+--+--+-----+		

[ISO page 14]

Method of Encoding Parameters

Coding Real Numbers

"e" is the extension flag, see 6.1 for a description.

"s" is the sign bit (bit b5 of the first byte): 0 for a positive mantissa, 1 for a negative mantissa.

Here "p" is the exponent-follows bit. If the current "explicit exponent allowed" value of the REAL PRECISION element is 'allowed', bit b4 of the first byte of a mantissa is used as the "exponent follows" bit: 1 if an explicit exponent follows, 0 if no exponent follows the mantissa (then the default exponent set by REAL PRECISION is assumed). If the current "explicit exponent allowed" value of the REAL PRECISION element is 'forbidden', bit b4 of the first byte is used as a databit and the default exponent set by REAL PRECISION is assumed.

The exponent is coded as an integer in Basic format, see 6.3.

Mantissas or exponents of "minus zero" are not allowed and reserved for future use.

For example, suppose that REAL PRECISION has specified that each real parameter is to be coded with an "exponent follows" bit in its mantissa. In that case, the binary numeral +1.110010110011 is coded as follows:

```
(real: binary +1.11 00101 10011)
= (mantissa: +111 00101 10011, "exponent-follows")
  (exponent: -12)
=
  +-+---+---+---+---+ +-+---+---+---+---+ +-+---+---+---+---+
  |X|1|1|0|1|1 1 1| |X|1|1|0 0 1 0 1| |X|1|0|1 0 0 1 1| {mantissa}
  +-+---+---+---+---+ +-+---+---+---+---+ +-+---+---+---+---+
  +-+---+---+---+---+
  |X|1|0|1|1 1 0 0| {exponent}
  +-+---+---+---+---+
= 6/15 6/5 5/3 5/12
```

Again, suppose that REAL PRECISION has specified that real parameters are to be coded without exponents and without exponent-follows bits in their mantissas, and that the default exponent is to be -13. In that case, binary +1.110010110011 is coded as follows:

```
(real: binary +1.110010110011)
= (mantissa: +1110 01011 00110)
=
  +-+---+---+---+---+ +-+---+---+---+---+ +-+---+---+---+---+
  |X|1|1|0|1 1 1 0| |X|1|1|0 1 0 1 1| |X|1|0|0 0 1 1 0| {mantissa}
  +-+---+---+---+---+ +-+---+---+---+---+ +-+---+---+---+---+
= 6/14 6/11 4/6
```

6.5 Coding VDCs and Points

A point is a pair of VDC scalars. A VDC scalar is either an integer or real number according to whether VDC TYPE is integer or real.

When VDC TYPE is integer, the encodings of the VDC and point data types are as described in 6.3, Coding Integers. The size of the VDC and point parameters is limited by the current VDC INTEGER PRECISION value.

When VDC TYPE is real, the encodings of the VDC and point data types are as described in 6.4, Coding Real Numbers. The size of the VDC and point parameters is limited by the current VDC REAL PRECISION value. Whether or not "exponent follows" bits are included in the mantissas of VDCs and points is

Coding VDCs and Points

Method of Encoding Parameters

determined by a parameter of VDC REAL PRECISION. The default value for an omitted exponent in a VDC is determined by the "default exponent" parameter in VDC REAL PRECISION. If the exponent is omitted from a point data type, a default exponent is assumed as follows.

- If the point is not in a point list, or is the first point of a point list, the omitted exponent assumes the default value exponents of VDC parameters, as specified by VDC REAL PRECISION.
- If the point is in a point list, but is not the first point of that point list, an omitted exponent may assume a different value as follows: if the exponent is omitted from the x-component of a real point, it defaults to the value of the exponent in the preceding x-component; similarly, an exponent omitted from a y-component assumes the value of the exponent in the preceding y-coordinate.

6.6 Coding Point List Parameters

Point lists may be coded with one or both of the following coding structures. These structures are called 'Displacement Mode' and 'Incremental Mode'. Both formats may be used within a single point list.

Individual points and the first point of each point list are always be coded in Displacement Mode. The displacement values delta-x and delta-y are displacements measured from the origin, i.e. absolute positions. For points after the first point in a point list, each displacement is measured from the preceding point of the point list (this is true regardless of whether the preceding point was coded in Displacement Mode or in Incremental Mode).

6.6.1 Displacement Mode

In Displacement Mode, each point is coded as a sequence of two VDCs. The first VDC value gives the x-component of the point's displacement from the preceding point, while the second VDC value specifies the y-component of that displacement.

$$\langle P \rangle = \langle \text{VDC: delta } x \rangle \langle \text{VDC: delta } y \rangle$$

6.6.2 Incremental Mode

The Incremental Mode is defined as a Differential Chain Code (DCC). The data in this mode does not reflect actual coordinates, but defines steps (increments) from one coordinate position to another. These increments are identified by points on a Ring. A Ring is a set of points on a square whose centre is the previously identified point. The first centre point is encoded in Displacement Mode.

A Ring is characterized by its Radius (R) in Basic Grid Units (BGU), 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:

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

N is required to be even.

The points on the Ring are numbered, starting at the Direction point D, counter clockwise from 0 to M-1 and clockwise from -1 to -M, with $M = N/2$.

Following are examples of two Rings, both having Radius $R=3$. The Ring on the left has Angular resolution factor $p=0$ and the one on the right has Angular resolution factor $p=1$.

Method of Encoding Parameters

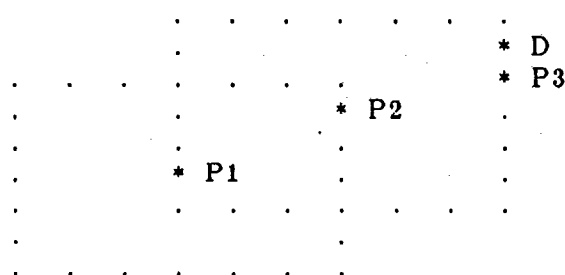
Coding Point List Parameters

	8	7	6	5	4		3		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

The Direction of the Ring is given by the position of the point with number ZERO. The initial position of this point is on the positive x-axis of VDC space, 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:

Let P2 be the current centre point and P1 the previous one, i.e., P2 is a point on the Ring with centre in P1. The Direction of the Ring is obtained by drawing a line from P1 through P2; where this direction ray intersects P2's Ring is the ZERO point of that Ring. So the Direction of the Ring is dependent on the direction of the line to be displayed.

In the DCC only the differences between points on the consecutive Rings are coded. The position of point P3, i.e., the increment on the new Ring (centre P2) is described by the difference between the position of point P2 on the previous Ring and the position of the new point P3 on the current Ring, the positions numbered with respect to the previous Ring. As shown in the figure below, 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 the previous figure. The Direction (position of the point with number ZERO) is identified by D.



Change of direction with $R=3$

The basic Radius of the Ring, as used in Incremental mode, is dependent on the BGU. The BGU is the smallest nonzero value that can be expressed within the precision set by the VDC INTEGER or REAL PRECISION element. For VDC TYPE integer, basic Radius ($=BGU$) = 1. For VDC TYPE real the size of the BGU is set by a parameter of VDC REAL PRECISION, called smallest-real code: letting "src" stand for smallest-real code, $BGU=2^{src}$. The basic Radius is a multiple of the BGU, i.e. it is a positive integer. For VDC TYPE real the default value for the basic Radius follows from: