



INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

T.101

(11/94)

TERMINALS FOR TELEMATIC SERVICES

INTERNATIONAL INTERWORKING FOR VIDEOTEX SERVICES

ITU-T Recommendation T.101

(Previously "CCITT Recommendation")

FOREWORD

The ITU-T (Telecommunication Standardization Sector) is a permanent organ of the International Telecommunication Union (ITU). The ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

The approval of Recommendations by the Members of the ITU-T is covered by the procedure laid down in WTSC Resolution No. 1 (Helsinki, March 1-12, 1993).

ITU-T Recommendation T.101 was revised by ITU-T Study Group 8 (1993-1996) and was approved under the WTSC Resolution No. 1 procedure on the 11th of November 1994.

NOTE

In this Recommendation, the expression “Administration” is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

© ITU 1995

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the ITU.

CONTENTS

	<i>Page</i>
Summary.....	x
1 Scope.....	1
2 Normative references	2
3 Abbreviations	4
4 Interworking between Videotex services: General	4
5 International interworking of Videotex services.....	5
5.1 International interworking configurations	5
5.2 Gateway to gateway interworking	5
5.3 Terminal to host interworking	5
6 International interworking between gateways.....	5
6.1 International interworking at network level.....	6
6.2 Transport layer.....	6
6.3 Session layer	6
6.4 Presentation layer.....	6
6.4.1 Presentation protocol	6
6.4.2 Coding of Videotex information	6
6.5 Application layer	7
6.6 Relationship with DTAM/ODA.....	7
7 International interworking between a terminal and a host	8
7.1 Access via PSTN or ISDN bearer service	8
7.2 Access via PSPDN or ISDN bearer service.....	9
7.3 Access via PSPDN/PAD.....	9
7.4 Access via PSPDN through a VIU	10
7.5 Access via PSPDN through a VSU	11
7.6 Application rules for Recommendation X.29 to support administrative functions.....	11
7.6.1 Charging.....	12
7.6.2 Limit handling.....	14
7.6.3 Identification	14
7.6.4 Language-Management.....	15
7.6.5 Data-Syntax-Management	15
7.6.6 Error-Message.....	15
7.7 Coding	15
7.8. ASN.1 encoding of the Administrative commands of Terminal to Host interworking	16
7.9 State table for charging events received at VSU side	18
7.9.1 Used abbreviations.....	18
7.9.2 State table.....	20
7.9.3 Collision.....	20
8 Content architecture class attributes.....	20
8.1 Content architecture class	20
8.2 Content type.....	20

9	Content portion attributes	21
9.1	Type of coding	21
9.2	Specific coding attributes	21
9.2.1	Subset	22
9.2.2	Rank	22
9.2.3	Profile	22
10	Formal definition of Videotex dependent data type	22
10.1	Introduction	22
10.2	Summary of ASN.1 object identifiers	23
11	Common enhancements to data syntaxes I, II and III	24
11.1	Terminal identifier	24
11.1.1	Terminal identification command response messages	24
11.1.2	Modem capability	25
11.1.3	Audio capabilities	26
11.1.4	Photographic capabilities	27
11.2	ISO 9281 support	28
11.3	Abnormal truncation of data	30
12	Definition of VideotexString within ASN.1	30
12.1	General	30
12.2	Definition of VideotexString	32
12.3	ASN.1 syntax element definition	32
Annex A	Interworking Data Syntax (IDS) described in ASN.1 (Recommendation X.208)	32
A.1	Videotex page	32
A.2	State vector	33
A.2.1	Vector definition	33
A.2.2	Reset state vector	36
A.2.3	NULL	36
A.3	Functions and parameters	36
A.3.0	Alpha char string	37
A.3.1	Special char string	37
A.3.2	Kana char string	37
A.3.3	Kanji char string	37
A.3.4	Block mosaic string	37
A.3.5	Smooth mosaic string	38
A.3.6	Special mosaic string	38
A.3.7	Format effector C0-char	39
A.3.8	Special format-C0-char	39
A.3.9	General control characters	39
A.3.10	Geometric string	51
A.3.11	Animation control string	59
A.3.12	Segment control string	60
A.3.13	Colour control string	62
A.3.14	Text colour string	63
A.3.15	Photographic string synthetic image	65
A.3.16	Photographic string natural image	67
A.3.17	Macro	68
A.3.18	DRCS string	69
A.3.19	Fill pattern control string	71
A.3.20	Music string	72
A.3.21	Telesoftware string	73
A.3.22	Audio data string	73

Appendix I – Text and mosaic character repertoires	73
I.1 Repertoire I – Common alphanumeric text characters.....	73
I.1.1 Latin alphabetic characters.....	73
I.1.2 Non-alphabetic characters.....	75
I.2 Repertoire 2 – Special alphanumeric text characters.....	77
I.3 Repertoire 3 – Kana characters.....	77
I.4 Repertoire 4 – Kanji characters	78
I.5 Repertoire 5 – Greek characters.....	79
I.6 Repertoire 6 – Cyrillic characters	79
I.7 Repertoire 7 – Block mosaic characters.....	80
I.8 Repertoire 8 – Sub cell aligned smooth mosaics 1	81
I.9 Repertoire 9 – General smooth mosaics	83
I.10 Repertoire 10 – Drawing characters	83
I.11 Repertoire 11 – Special mosaics.....	84
Appendix II – Default states of the interworking data syntax	85
Annex B – Data Syntax I for international interactive Videotex service	90
Preface	90
B.1 General.....	90
B.1.1 Scope	90
B.1.2 References.....	90
B.1.3 Definitions.....	90
B.1.4 Visual information display.....	92
B.1.5 Sound information	95
B.2 Coding architecture.....	95
B.2.1 General	95
B.2.2 Mode switching technique	95
B.2.3 Coding structure in the character code mode.....	96
B.2.4 Coding structure in the transparent mode	99
B.2.5 Coding structure in the sound mode	100
B.2.6 Coding structure in the message mode.....	101
B.3 Coding in the character code mode.....	101
B.3.1 C0 Control Set.....	101
B.3.2 C1 Control Set.....	103
B.3.3 Display control command set.....	108
B.3.4 Character Sets	125
B.3.5 Mosaic Sets.....	125
B.3.6 Dynamically Redefinable Character Sets (DRCs).....	125
B.3.7 Macro Set.....	137
B.3.8 Picture Description Instruction (PDI) Set	138
B.3.9 Moving Instruction (MVI) Set	139
B.4 Coding in the transparent mode	142
B.4.1 Photographic Data Unit (PDU).....	142
B.4.2 LINE DOT PATTERN	143
B.4.3 LINE DOT PATTERN COMPRESSED	143
B.4.4 FIELD DOT PATTERN	148
B.4.5 FIELD DOT PATTERN COMPRESSED.....	149
B.4.6 COLOURING BLOCK	150
B.4.7 COLOURING BLOCK COMPRESSED	150
B.4.8 FIELD COLOURING BLOCK	152
B.4.9 FIELD COLOURING BLOCK COMPRESSED	154
B.4.10 FREE FORMAT COLOURING BLOCK	157
B.4.11 PHOTO DRCS 1 AND PHOTO DRCS 2	157

B.4.12	Telesoftware.....	161
B.4.13	PCM photography.....	162
B.4.14	ABTC photography.....	163
B.4.15	ADCT photography.....	164
B.5	Coding in the sound mode.....	165
B.5.1	General.....	165
B.5.2	Sound tone set.....	165
B.5.3	Sound control set.....	171
B.6	Terminal identification.....	175
B.6.1	Introduction.....	175
B.6.2	General structure.....	175
B.6.3	Details for Data Syntax I terminals.....	176
B.7	Default conditions.....	177
Appendix I – Service Reference Model (SRM)		178
Appendix II – Operation of CS and NSR.....		179
Appendix III – Adaptive block truncation coding.....		180
III.1	Introduction.....	180
III.2	Adaptive block truncation coding (ABTC) algorithm.....	180
III.3	ABTC decoding algorithm.....	185
Annex C – Data Syntax II for international interactive Videotex service.....		187
General.....		187
C.1	Introduction.....	187
C.1.1	Coding principles.....	187
C.1.2	Display principles.....	188
C.2	References.....	189
C.3	Definitions.....	189
Part 1 – Alphamosaic display.....		190
C.1	Description.....	190
C.1.1	Introduction.....	190
C.1.2	Theoretical Terminal Model.....	191
C.1.3	Defined Attributes and Qualified Areas.....	192
C.1.4	Rules for the Action of the SIZE Attribute.....	195
C.1.5	Defaults.....	195
C.2	Repertory.....	198
C.2.1	Character Repertoire.....	198
C.2.2	Format Effector Repertoire.....	224
C.2.3	Attribute Control Repertoire.....	225
C.2.4	Device control function repertoire.....	234
C.3	Coding Structure.....	235
C.3.1	Code extension and invocation.....	235
C.3.2	The Primary control function set.....	237
C.3.3	The supplementary control function sets.....	239
C.3.4	The coding of graphic characters.....	245
C.3.5	Supplementary attribute and qualified area controls.....	279
C.3.6	Device controls.....	281
C.3.7	Designation and invocation in the 7-bit environment.....	282
C.3.8	Designation and invocation in the 8-bit environment.....	282
Appendix I – Identification system.....		285
Appendix II – Example of time dependency in the unified alphamosaic model.....		287

Part 2 – Geometric displays.....	289
Part 3 – Photographic display.....	409
Part 4 – Define Dynamically Redefinable Character Sets (DRCS).....	434
Part 5 – Define COLOUR	443
Part 6 – Define FORMAT	447
Part 7 – TRANSPARENT Data	448
Part 8 – RESET	449
Part 9 – Telesoftware.....	452
Part 10 – Terminal Facility Identifier	562
Part 11 – Timing Control.....	575
Annex D – International Interworking for Videotex Services Data Syntax III	585
Preface	585
D.1 Scope	586
D.2 Definitions	586
D.3 Reference Publications	589
D.3.2 ANSI Standards	589
D.3.3 CSA Standards	589
D.3.4 CCIR Report	589
D.3.5 CCITT Recommendations	589
D.3.6 Department of Communication, Canada.....	589
D.3.7 EIA/CVCC Recommendation 1983,	589
D.3.8 ISO Standards	590
D.4 Coding Architecture.....	590
D.4.1 Reference Model (OSI).....	590
D.4.2 Presentation Level Overview	590
D.4.3 Code Extension	592
D.4.4 SPACE and DELETE	598
D.5 Coding of G-sets	598
D.5.1 Primary character set.....	598
D.5.2 Supplementary character set	600
D.5.3 Picture Description Instruction (PDI) set.....	602
D.5.4 Mosaic set	646
D.5.5 Macro set.....	647
D.5.6 Dynamically Redefinable Character Set (DRCS)	648
D.6 Coding of C-sets	648
D.6.1 C0 control set	648
D.6.2 C1 control set	651
D.7 Graphic character repertoire	657
D.8 Conformance requirements.....	670
D.8.1 General	670
D.8.2 Conforming interchange	671
D.8.3 Conforming Presentation Process	671
D.9 Enhanced Capabilities.....	672
Appendix I – Layered Architecture Model.....	674
Appendix II – Coordinate System Concepts	675
Appendix III – Code extension and 7-bit/8-bit transform	677
Appendix IV – A general Service Reference Model (SRM) for Videotex and a general Service Reference Model (SRM) for Teletext.....	678

Addendum to Data Syntax III for international interactive videotex service (1993)	687
Preface	687
1 Underline.....	687
2 Absence of operands on PDIs	687
3 RESET PDI, NON-SELECTIVE RESET (NSR) AND ACTIVE POSITION SET (APS).....	687
4 Cursors	687
5 Macro linkages	688
6 Logical pel dimensions.....	688
7 Word wrap.....	688
8 INCREMENTAL commands	688
9 Unprotected fields	688
10 Cursor and macro	688
11 Proportional spacing of text	689
11.1 Width tables	689
11.2 The displacement function.....	689
11.3 Features of the function	690
11.4 The function.....	690
11.5 Text sizes $0 \leq DX < 12/256$	690
11.6 Text sizes $DX \geq 12/256$	691
12 Character field sizes	692
13 Code extension.....	692
13.1 Subclause D.4.3.1.1	692
13.2 Subclause D.4.3.1.2	692
13.3 Subclause D.4.3.2	693
13.4 Figures D.4 and D.6.....	694
14 ISO/IEC 9281 support.....	696
15 Alignment with Recommendation T.51	698
15.1 Subclause D.5.2	698
15.2 Figure D.8.....	699
15.3 Repertoire	701
15.4 NO BREAK SPACE and SOFT HYPHEN	703
16 Terminal identification.....	703
Annex E – Audio data syntax	704
E.1 Introduction	704
E.2 Scope	705
E.3 Normative references.....	705
E.4 Definitions	705
E.5 Symbols and abbreviations	706
E.6 Overview	706
E.7 Introducer.....	706
E.8 ISO/IEC 9281 [8] syntax and switching structure	706
E.8.1 Overall switching of coding environment.....	706
E.8.2 Switching into the Audio Mode	708
E.8.3 ISO/IEC 9281 [8] syntax structure.....	708

E.9	Sound header	709
E.9.1	Introduction.....	709
E.9.2	Header structure	709
E.9.3	Default values	711
E.10	Sound block	711
E.11	Application rules for ISDN syntax-based Videotex	712
E.11.1	Encoding/bitrate combinations	712
E.11.2	Translation mechanisms.....	712
E.12	Translation modes.....	712
E.12.1	Mode 0 (no translation is performed)	712
E.12.2	Mode 1 (No translation except US)	712
E.12.3	Mode 2 (3-in-4 coding)	712
E.12.4	Mode 3 (Shift scheme – 8 bits)	713
E.12.5	Mode 4 (Shift scheme – 7 bits)	714
E.12.6	Mode 5 (No translation except specific characters)	714
Annex F – Photographic data syntax		715
Introduction		715
F.1	Scope	715
F.2	Normative references.....	716
F.3	Definitions and abbreviations	717
F.3.1	Definitions.....	717
F.3.2	Abbreviations	719
F.4	Overview	720
F.5	ISO/IEC 9281, Part 1, syntax and switching structure	720
F.5.1	Overall switching of coding environment.....	720
F.5.2	Switching into the photographic mode	722
F.5.3	ISO/IEC 9281, Part 1, syntax structure	722
F.6	Coding of the Picture Data Entity (PDE).....	723
F.6.1	Introduction.....	723
F.6.2	PDE data content identification mechanism	724
F.7	Photographic header	725
F.7.1	Introduction.....	725
F.7.2	Header structure	725
F.7.3	Header functionalities	727
F.8	Coding rules.....	742
F.8.1	Purpose.....	742
F.8.2	General rules for coding the header	742
F.8.3	Photographic header code assignment	749
F.8.4	Encoding of photographic header parameters	750
F.9	Photographic data	754
F.9.1	Introduction.....	754
F.9.2	Translation modes.....	755
F.10	Defaults.....	755
F.10.1	Default values for photographic header attributes	755
F.10.2	Default tables	758
F.11	Photographic profiles.....	768
F.11.1	Compatible photographic profiles (P1 to P5).....	770
F.11.2	Private choice of photographic profile (P _{priv}).....	772
F.12	Combination rule	772

Appendix I – Photovideotex tutorial	773
I.1 Introduction	773
I.2 The present state of photovideotex	773
I.3 Image representation.....	774
I.4 The JPEG compression technique	775
I.4.1 Lossy and lossless compression	775
I.4.2 Modes of encoding.....	776
I.4.3 The DCT-based coding	777
I.4.4 Lossless coding	782
I.4.5 Source images and data interleaving.....	782
I.4.6 Data organization and signalling parameters	783
I.5 The baseline system.....	783
I.6 The extended system.....	783
I.6.1 Coding model for successive approximation	783
I.6.2 Coding model for spectral selection.....	783
I.6.3 Hierarchical encoding	784
I.7 Summary.....	784
I.8 Bibliography	784
Appendix II – Implementation guidelines on display rendering	785
II.1 Introduction	785
II.2 Rendering of resolution	785
II.2.1 Resolution independence	785
II.2.2 Display rendering guidelines for Data Syntax II profiles	787
II.2.3 Display rendering for Data Syntax III profiles	789
II.3 The concept of normalised colour space.....	790
Appendix III – Solutions for common compatible photovideotex databases serving different resolution terminals	790
III.1 Introduction	790
III.2 Hierarchical mode.....	791
III.2.1 Coding	791
III.2.2 Decoding.....	792
III.2.3 Example for a “resolution pyramid” for hierarchical build-up	792
III.3 Special spectral selection.....	794
III.3.1 Coding	795
III.3.2 Decoding.....	796
Appendix IV – Coding examples	797
IV.1 Introduction	797
IV.1.1 Example 1	797
IV.1.2 Example 2	798
IV.1.3 Example 3	801
IV.2 Image positioning examples	802
IV.2.1 Example 1: 640 × 480 picture inside DDA.....	802
IV.2.2 Example 2: 720 × 576 picture full screen	802
IV.2.3 Example 3: 720 × 346 picture covering upper 60% of full screen.....	803
IV.3 Example for Source Picture Comments (PCT).....	804
IV.3.1 An application scenario.....	804
IV.3.2 Sample logical record of a source picture	804

Appendix V – Encoding parameters values for the 2:1:1 derived from CCIR Recommendation 601, Part 1 [9].....	805
V.1 Introduction	805
V.2 Encoding parameters for 2:1:1.....	806
V.2.1 Main body of CCIR Recommendation 601, Part 1 [9]	806
V.2.2 Appendix I of the CCIR Recommendation 601, Part 1 [9]	806
V.2.3 Appendix II of the CCIR Recommendation 601, Part 1 [9].....	806
V.2.4 Appendix III of the CCIR Recommendation 601 [9].....	806
Appendix VI – Translation modes	812
VI.0 Mode 0 (No translation, full transparency).....	812
VI.1 Mode 1 (No translation except US)	812
VI.2 Mode 2 (3-in-4 coding).....	812
VI.3 Mode 3 (Shift scheme – 8 bits).....	812
VI.4 Mode 4 (Shift scheme – 7 bits).....	813
VI.5 Mode 5 (no translation except specific characters).....	813
Appendix VII – Huffman tables for the “special spectral selection”	814
VII.1 Introduction	814
VII.2 Spectral bands.....	814
VII.3 Luminance DC differences	814
VII.3.1 List of codelengths	814
VII.3.2 List of values.....	814
VII.4 Chrominance DC differences.....	815
VII.4.1 List of codelengths	815
VII.4.2 List of values.....	815
VII.5 Luminance AC coefficients	815
VII.5.1 List of codelengths	815
VII.5.2 List of values.....	816
VII.6 Chrominance AC coefficients.....	820
VII.6.1 List of codelengths	820
VII.6.2 List of values.....	820
Appendix VIII – Examples of local presentation facilities with T4 and T6 encoded data	824
VIII.1 Local presentation facilities	824
VIII.2 Local presentation modes	824
VIII.2.1 Full view mode	825
VIII.2.2 Full width mode	825
VIII.2.3 Full height mode	825
VIII.2.4 Original size mode	826
VIII.2.5 Terminal dependant mode.....	826

SUMMARY

This Recommendation defines the rules applicable to the international interworking between videotex services. Two classes of interworking are specified:

- interworking between gateways (see Recommendations T.504, T.523, T.541 and T.564);
- interworking between a terminal and a host (X.29 based protocols).

Moreover the different Data Syntaxes are defined in the following annexes:

- Interworking Data Syntax (Annex A);
- Data Syntax I (CAPTAIN) (Annex B);
- Data Syntax II (former CEPT data syntax) (Annex C);
- Data Syntax III (NAPLPS) (Annex D).

In addition, common extensions to the various data syntaxes are defined:

- Audio Data Syntax (Annex E);
- Photographic Data Syntax (Annex F),

as well as the ISO 9281 based switching mechanism between all syntaxes.

- Annex A – Interworking Data Syntax (IDS);
- Annex B – Data Syntax I (CAPTAIN);
- Annex C – Data Syntax II (former CEPT data syntax);
- Annex D – Data Syntax III (NAPLPS);
- Annex E – Audio Data Syntax;
- Annex F – Photographic Data Syntax.

INTERNATIONAL INTERWORKING FOR VIDEOTEX SERVICES

(revised, 1994)

1 Scope

1.1 The purpose of this Recommendation taking into account:

- that Videotex services have been implemented in different countries/regions using different data syntaxes referred to as Data Syntax I, Data Syntax II and Data Syntax III which have an equal status;
- that different countries are entitled to use their existing systems;
- that interworking between Videotex services in different countries may require transcoding and/or conversion;
- that the interworking between Videotex services may be provided by using different types of networks such as the public switched telephone network (PSTN), packet switched public data network (PSPDN), circuit switched public data network (CSPDN), integrated services digital network (ISDN), etc.;
- that Videotex interworking protocols should offer a large degree of compatibility with protocols used in other telematic services,

is:

- to facilitate the interworking of different Videotex services;
- to identify parameters needed to communicate with Videotex terminals;
- to provide technical recommendations desirable for potential interworking of other telematic services with Videotex services.

1.2 This Recommendation describes the characteristics of coded information that is exchanged between countries participating in the international interactive Videotex service.

Videotex systems are text communication systems having in addition 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.

Different data syntaxes are offered as a choice for the Administrations to implement their national services. Substantial degrees of compatibility exist between these options, but some transcoding and/or conversion may be necessary to facilitate interworking.

For the purpose of the international service, different data syntax have been identified:

- a) interworking data syntax;
- b) data syntax I;
- c) data syntax II;
- d) data syntax III;
- e) audio data syntax;
- f) photographic data syntax;
- g) other data syntaxes are for further study.

The common enhancements to data syntaxes I, II and III are identified in clause 9.

2 Normative references

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

- CCITT Recommendation F.300 (1988), *Videotex service*.
- CCITT Recommendation G.711 (1988), *Pulse code modulation of voice frequencies*.
- CCITT Recommendation G.721 (1988), *32 kbit/s adaptive differential pulse code modulation (ADPCM)*.
- CCITT Recommendation G.722 (1988), *7 kHz audio-coding within 64 kbit/s*.
- CCITT Recommendation G.723 (1988), *Extensions of Recommendation G.721 adaptive differential pulse code modulation to 20 and 40 kbit/s for digital circuit multiplication equipment application*.
- CCITT Recommendation H.221 (1988), *Frame structure of a 64 kbit/s channel in audio-visual teleservices*.
- CCITT Recommendation J.41 (1988), *Characteristics of equipment for the coding of analogue high quality sound programme signals for transmission on 384 kbit/s channels*.
- CCITT Recommendation J.42 (1988), *Characteristics of equipment for the coding of analogue medium quality sound programme signals for transmission on 384 kbit/s channels*.
- CCITT Recommendation T.50, *International Alphabet No. 5*.
- CCITT Recommendation T.51, *Latin based coded character sets for telematic services*.
- CCITT Recommendation T.52, *Non-Latin based coded character sets for telematic services*.
- CCITT Recommendation T.70, *Applicability of telematic protocols and terminal characteristics to computerized communication terminals (CCTs)*.
- CCITT Recommendation T.90, *Characteristics and protocols for terminal for telematics services in ISDN*.
- ITU-T Recommendation T.107, *Videotex enhanced man machine interface (VEMMI)*.
- CCITT Recommendation T.400, *Introduction to document architecture transfer and manipulation*.
- CCITT Recommendation T.411, *Open document architecture (ODA) and interchange format – Introduction and general principles*.
- CCITT Recommendation T.412, *Open document architecture (ODA) and interchange format – Document structures*.
- CCITT Recommendation T.414, *Open document architecture (ODA) and interchange format – Document profile*.
- CCITT Recommendation T.415, *Open document architecture (ODA) and interchange format – Document interchange format (ODIF)*.
- CCITT Recommendation T.431, *Document transfer and manipulation (DTAM) – Introduction and general principles*.
- CCITT Recommendation T.432, *Document transfer and manipulation (DTAM) – Service definitions*.
- CCITT Recommendation T.433, *Document transfer and manipulation (DTAM) – Protocol specifications*.
- CCITT Recommendation T.441, *Document transfer and manipulation (DTAM) – Operational structure*.
- CCITT Recommendation T.504, *Document application profile for Videotex interworking*.

- CCITT Recommendation T.523, *Communication application profile DM1 for Videotex interworking.*
- CCITT Recommendation T.541, *Operational application profile for Videotex interworking.*
- CCITT Recommendation T.564, *Gateway characteristics for Videotex interworking.*
- CCITT Recommendation V.17 (1990), *Recommendation for a 2-wire modem for facsimile applications with rates up to 14 400 bit/s.*
- CCITT Recommendation V.21 (1988), *300 bits per second duplex modem standardized for use in the general switched telephone network.*
- CCITT Recommendation V.22 (1988), *1200 bits per second duplex modem standardized for use in the general switched telephone network and on point-to-point 2-wire leased telephone-type circuits.*
- CCITT Recommendation V.22 bis (1988), *2400 bits per second duplex modem using the frequency division technique standardized for use on the general switched telephone network and on point-to-point 2-wire leased telephone-type circuits.*
- CCITT Recommendation V.23 (1988), *600/1200 bits per second modem standardized for use in the general switched telephone network.*
- CCITT Recommendation V.26 bis (1988), *2400/1200 bits per second modem standardized for use in the general switched telephone network.*
- CCITT Recommendation V.26 ter (1988), *2400 bits per second duplex modem using the echo cancellation technique standardised for use on the general switched telephone network and on point-to-point 2-wire leased telephone-type circuits.*
- CCITT Recommendation V.27 ter (1988), *4800/2400 bits per second modem standardised for use in the general switched telephone network.*
- CCITT Recommendation V.29 (1988), *9600 bits per second modem standardised for use on point-to-point 4-wire leased telephone-type circuits.*
- CCITT Recommendation V.32 (1988), *A family of 2-wire, duplex modems operating at data signalling rates of up to 9600 bit/s for use on the general switched telephone network and on leased telephone-type circuits.*
- CCITT Recommendation V.32 bis (1991), *A duplex modem operating at data signalling rates of up to 14 400 bit/s for use on the general switched telephone network and on leased point to point 2-wire telephone type circuits.*
- ITU-T Recommendation V.34 (1994), *A modem operating at data signalling rates of up to 28 800 bit/s for use on the general switched telephone network and on leased point-to-point 2-wire telephone-type circuits.*
- CCITT Recommendation V.42 (1988), *Error correcting procedures for DCEs using asynchronous to synchronous conversion.*
- CCITT Recommendation V.42 bis (1990), *Data compression procedures for data circuit-terminating equipment (DCE) using error correcting procedures.*
- CCITT Recommendation X.3, *Packet assembly/disassembly facility (PAD) in a public data network.*
- CCITT Recommendation X.25, *Interface between data terminal equipment (DTE) and data circuit-terminating equipment (DCE) for terminal operating in the packet mode and connected to public data networks by dedicated circuit.*
- CCITT Recommendation X.28, *DTE/DCE interface for a start-stop mode data terminal equipment accessing the packet assembly/disassembly facility (PAD) in a public data network situated in the same country.*
- CCITT Recommendation X.29, *Procedures for the exchange of control information and user data between a PAD facility and a packet mode DTE or another PAD.*
- CCITT Recommendation X.75, *Packet switched signalling system between public networks providing data transmission services.*
- CCITT Recommendation X.200, *Reference Model of Open System Interconnection for CCITT applications.*

- CCITT Recommendation X.213, *Network service definition for Open System Interconnection for CCITT applications.*
- CCITT Recommendation X.214, *Transport service definition for Open System Interconnection for CCITT applications.*
- CCITT Recommendation X.224, *Transport protocol specification for Open System Interconnection for CCITT applications.*
- CCITT Recommendation X.215, *Session service definition for Open System Interconnection for CCITT applications.*
- CCITT Recommendation X.225, *Session protocol specification for Open System Interconnection for CCITT applications.*
- CCITT Recommendation X.216, *Presentation service definition for Open System Interconnection for CCITT applications.*
- CCITT Recommendation X.226, *Presentation protocol specification for Open System Interconnection for CCITT applications.*
- CCITT Recommendation X.217, *Association control service element service definition for Open System Interconnection for CCITT applications.*
- CCITT Recommendation X.227, *Association control service element protocol specification for Open System Interconnection for CCITT applications.*
- ISO 639, *Language, country and authority indicatifs.*
- ISO 11172-3, *Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s – Audio part.*

3 Abbreviations

For the purpose of this Recommendation, the following abbreviations are used:

CCITT	International Telegraph and Telephone Consultative Committee
CCIR	International Radiocommunication Consultative Committee
CSPDN	Circuit Switched Public Data Network
ISDN	Integrated Service Data Network
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union – Telecommunication Standardization Sector (formerly the CCITT)
PAD	Packet Assembly Disassembly
PDN	Public Data Network
PSPDN	Packet Switched Public Data Network
PSTN	Public Switched Telephone Network
VEMMI	Videotex Enhanced Man Machine Interface
VIU	Videotex Interface Unit
VSU	Videotex Service Unit

4 Interworking between Videotex services: General

It is the responsibility of Administrations to decide on which network(s) the Videotex service(s) are to be provided.

Several possibilities are considered below.

Videotex service operated on the PSTN: the communication between a Videotex terminal and a Videotex host computer is established over the PSTN.

Videotex service operated on the PSTN and a public data network (PDN) (generally a PSPDN): the communication between a Videotex terminal connected to the PSTN and a Videotex host computer connected to a PDN is established via a Videotex access point or a Videotex service centre interfacing between both networks.

Other possibilities (CSPDN, ISDN, etc.) could also be considered.

International interworking between Videotex services via gateways and connected to any network (PSTN, PSPDN, CSPDN, ISDN, etc.) may be possible. Such interworking allows a Videotex terminal pertaining to a Videotex service to access a Videotex host computer pertaining to another Videotex service. International interworking between a Videotex terminal in one country and a Videotex host in another country may also be possible. All international data exchange should comply with the specifications contained in this Recommendation. (See Recommendation F.300 for the service description).

5 International interworking of Videotex services

Videotex interworking allows a Videotex terminal in a given country to interact in real time with a Videotex application located in a different country.

International interworking between Videotex services should use those functions that are defined in the data syntaxes implemented by the Administrations concerned: data syntaxes I, II and III defined in Annexes B, C and D respectively.

5.1 International interworking configurations

The various configurations for international interworking are defined in Recommendation F.300. The two major classes of interworking are defined below.

5.2 Gateway to gateway interworking

This class of interworking involves communication between gateways located in each country and where all the data handling processes involved in the interworking are performed. The protocols and data syntaxes for this class of interworking are specified in clause 6.

5.3 Terminal to host interworking

This class of interworking involves communication between a terminal and a host located in different countries, either directly or through a conversion unit situated in the country where the terminal is located. Several cases have been identified. The protocols and data syntaxes for the various cases of this class of interworking are specified in clause 7.

6 International interworking between gateways

The international interworking between gateways allows a Videotex terminal located in country A to access the Videotex services located in country B via a Videotex service of country A. The configuration for the international interworking between gateways is described by Figure 1.



Figure 1/T.101 – International interworking between gateways

6.1 International interworking at network level

International interworking between Videotex services should preferably take place between networks of the same type when these networks are provided by both Administrations involved (PSPDN, CSPDN, ISDN and leased lines).

The network service definition of Open Systems Interconnection for ITU-T application is defined in Recommendation X.213.

When the interworking takes place between Videotex services operating on different types of network, Recommendation X.75 should apply. Interworking with ISDN should be in accordance with CCITT Recommendation T.90.

6.2 Transport layer

The Transport layer service of Open Systems Interconnection for ITU-T applications is defined in Recommendation X.214.

The Transport protocol of Open Systems Interconnection for ITU-T applications is specified in Recommendation X.224.

Both classes 0 (corresponding to Recommendation T.70) and 2 may be used.

When class 0 is selected, then the protocol used is fully compatible with CCITT Recommendation T.70. When class 2 is selected, explicit flow control is to be used.

6.3 Session layer

This Session layer service of Open Systems Interconnection for ITU-T applications is defined in Recommendation X.215. The session protocol of Open Systems Interconnection for ITU-T applications is specified in Recommendation X.225.

The use of the session protocols by Videotex Interworking is defined in Recommendation T.523.

6.4 Presentation layer

6.4.1 Presentation protocol

The presentation layer service of Open Systems Interconnection for ITU-T applications is defined in Recommendation X.216. Presentation protocol of Open Systems Interconnection for ITU-T applications is specified in Recommendation X.226.

The use of the presentation protocols by Videotex Interworking is defined in Recommendation T.523.

6.4.2 Coding of Videotex information

Coding of the contents of the display-data element:

The Videotex content conforms to one of the several different data syntaxes. A data syntax, referred to as interworking data syntax is described in Annex A. There are three existing data syntaxes, based on Recommendation T.50, and referred to as data syntax I, data syntax II and data syntax III. They are described in Annex B, Annex C and Annex D, respectively. All the four annexes form an integral part of this Recommendation.

Different Administrations implementing a Videotex service may use one of the three above data syntaxes.

If two countries implement the same data syntax, then Videotex interworking between the two countries can use that same data syntax.

If one country implements one data syntax and another country implements a different data syntax profile, then Videotex interworking between the two countries can either:

- i) use the interworking data syntax as the intermediary syntax: transcoding/conversion into and from the interworking data syntax by the two countries will be required; or
- ii) use one of the two data syntaxes with transcoding/conversion performed either at the originating or at the destination country.

For designation and invocation of the particular in-use data syntax (I or II or III), the designation and invocation of the “complete code” escape sequence may be used:

- ESC 2/5 4/3 for data syntax I;
- ESC 2/5 4/4 for data syntax II;
- ESC 2/5 4/1 for data syntax III.

The “complete code” environment will be terminated either:

- by the sequence ESC 2/5 4/0; or
- by the designation and invocation of any other “complete code”.

6.5 Application layer

The Association Control Service Element (ACSE) of Open Systems Interconnection for ITU-T applications is defined in Recommendation X.217. The Association Control Service Element (ACSE) protocol of Open Systems Interconnection for ITU-T applications is specified in Recommendation X.227.

The application layer for Videotex interworking makes use of the following Recommendations :

- Recommendation T.400, *Introduction to Document Architecture Transfer and Manipulation*.
- Recommendation T.411, *Open document architecture (ODA) and interchange format – Introduction and general principles*.
- Recommendation T.412, *Open document architecture (ODA) and interchange format – Document structures*.
- Recommendation T.414, *Open document architecture (ODA) and interchange format – Document profile*.
- Recommendation T.415, *Open document architecture (ODA) and interchange format – Document interchange format (ODIF)*.

The application layer for Videotex interworking makes use of DTAM (Document Transfer And Manipulation) service and protocol described in Recommendation T.431, Recommendation T.432, Recommendation T.433.

The application layer for Videotex interworking makes use of operational structures described in Recommendation T.441.

Recommendation T.564 describes the Videotex interworking application profile and the gateway characteristics.

Recommendation T.504 describes the document application profile for Videotex interworking.

Recommendation T.523 describes the communication application profile for Videotex interworking.

Recommendation T.541 describes the operational application profile for Videotex interworking.

6.6 Relationship with DTAM/ODA

The relationships with the Open Document Architecture (see Recommendation T.412) and the document interchange format (see Recommendation T.415) are expressed through the content architecture class attributes and the content portion attributes as described in clauses 8 and 9.

7 International interworking between a terminal and a host

In this configuration, for designation and invocation of the particular host data syntax (I or II or III), the designation and invocation of the “complete code” escape sequence may be used :

- ESC 2/5 4/3 for data syntax I;
- ESC 2/5 4/4 for data syntax II;
- ESC 2/5 4/1 for data syntax III.

The “complete code” environment will be terminated either:

- by the sequence ESC 2/5 4/0; or
- by the designation and invocation of any other “complete code”.

7.1 Access via PSTN or ISDN bearer service

See Figure 2.

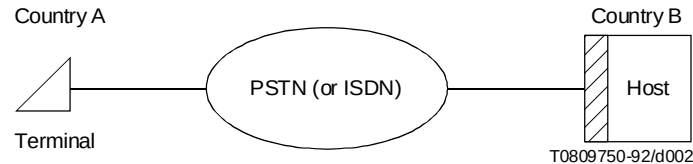


Figure 2/T.101 – Access via PSTN or ISDN bearer service

In this configuration, the terminal uses the international PSTN (respectively the ISDN bearer services) to reach the host. On the international link, the following protocols should be used:

- Layers 1-3 via PSTN: The protocols defined by the host.
- Layers 1-3 via ISDN bearer service: Recommendation T.90.
- Layers 4-7: The protocols (if any) defined by the host located in country B.
- Data syntax: Data syntax defined by the host.
- Dialogue/service functions: Functions defined by the host.

7.2 Access via PSPDN or ISDN bearer service

See Figure 3.

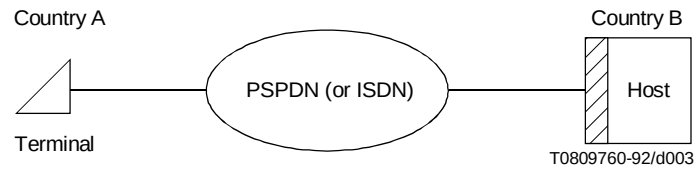


Figure 3/T.101 – Access via PSPDN or ISDN bearer service

In this configuration, the terminal uses the international PSPDN (respectively the ISDN bearer services) to reach the host. On the international link, the following protocols should be used:

- Layers 1-3 via PSPDN: Recommendation X.75.
- Layers 1-3 via ISDN bearer service: Recommendation T.90.
- Layers 4-7: The protocols (if any) defined by the host located in country B.
- Data syntax: Data syntax defined by the host.
- Dialogue/service functions: Functions defined by the host.

7.3 Access via PSPDN/PAD

See Figure 4.

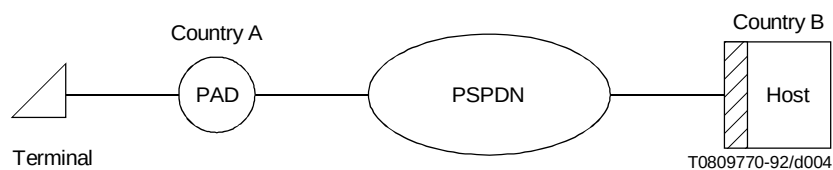


Figure 4/T.101 – Access via PSPDN/PAD

In this configuration, the terminal is connected to a PAD which gives access to the international PSPDN; both the terminal and the PAD are located in country A. The type of connection between the Terminal and the PAD is a national matter (generally the PSTN or a leased line).

The host of country B may be accessed through the international PSPDN. The type of connection between the host and the national PSPDN is a national matter (generally a leased line).

On the international link, the following protocols should be used:

- Layers (1-3): Recommendation X.75.
- Above layer 3: Recommendation X.29 and Recommendation X.3.
- Data syntax: Data syntax defined by the host located in country B.
- Dialogue/service functions: Functions defined by the host.

7.4 Access via PSPDN through a VIU

See Figure 5.

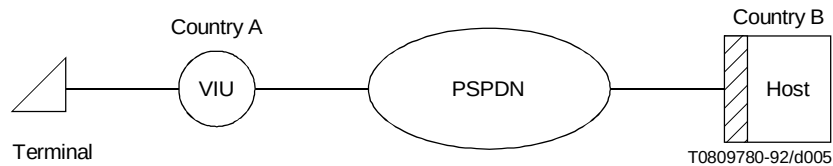


Figure 5/T.101 – Access via PSPDN through a VIU

In this configuration, the terminal is connected to a VIU (Videotex Interface Unit) which gives access to the international PSPDN; both the terminal and the VIU are located in country A. The type of connection between the Terminal and the VIU is a national matter (generally the PSTN or a leased line). The VIU performs two functions: it supports terminals and converts data-syntaxes. It is up to the Administration of country A to decide how a VIU is implemented: it may be realized as a separate system or integrated with an existing equipment (PAD or Videotex access point for example).

The Host of country B may be accessed through the international PSPDN. The type of connection between the host and the national PSPDN is a national matter (generally a leased line).

On the international link, the following protocols should be used:

- Layers (1-3): Recommendation X.75.
- Above layer 3: Recommendation X.29 and Recommendation X.3.

Alternatively Recommendation X.200 based protocols can be used. For this case, application profiles will need to be defined in T.500-Series Recommendations. This is for further study.

- Data syntax: The data syntax defined by the host located in country B.
- Dialogue/service functions: Those defined by the host.

7.5 Access via PSPDN through a VSU

See Figure 6.

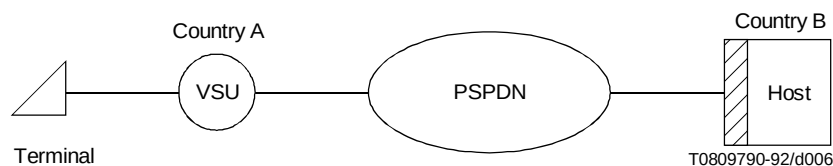


Figure 6/T.101 – Access via PSPDN through a VSU

In this configuration, the terminal is connected to a VSU (Videotex Service Unit) which gives access to the international PSPDN; both the terminal and the VSU are located in country A. A VSU is a VIU which is also in charge of handling application charge and accounting. It is up to the Administration of country A to decide to set up or not a VSU and to decide how a VSU, if any, is to be implemented: it may be realized as a separate system or integrated with existing equipment (PAD, Videotex access point or Videotex service centre).

The host in country B may be reached through the international PSPDN. The type of connection between the host and the national PSPDN is a national matter (generally a leased line).

On the international link, the following protocols should be used :

- Layers (1-3): Recommendation X.75.
- Above layer 3 : Recommendation X.200 based protocols.

For this case, application profiles need to be defined in the T.500-Series Recommendations. This is for further study.

Alternatively Recommendation X.29 plus Recommendation X.3 may be used. Extension (Application rules) to Recommendation X.29 are necessary (see 7.6).

- Data syntax: The data syntax defined by the host located in country B.
- Dialogue/service functions: Those defined by the host.

7.6 Application rules for Recommendation X.29 to support administrative functions

When an international communication is established via a VSU using Recommendation X.29 , X.29 Videotex commands may be used to allow application charges (if any) to be passed from the host to the VSU.

The Videotex commands should be sent in complete packet sequences with the Q-bit set to one.

Videotex commands use a T(ype)-L(ength)-V(alue) encoding. Fixed length commands do not require any length indicator. When used, length indicator is coded on two bytes and defines the total length, in bytes, of the V-field.

In order to distinguish Videotex commands from PAD commands as currently defined by Recommendation X.29, the type value of a Videotex command is defined with the most significant bit set to one.

The commands defined in this Recommendation have been identified in order to allow the exchange of information for charging and accounting and eventually for identification:

(The list may not be exhaustive)

Table 1 describes the command, the direction of the command and whether the command shall be implemented or not in the VSU and/or the host.

Table 1/T.101 – Terminal to host administrative commands

Command	Direction		Implementation at	
	VSU to Host	Host to VSU	VSU	Host
Charging-Modify-Request		X	M	O
Charging-Modify-Response	X		M	C
Application-Connection-Report		X	M	O
Application-Disconnection-Report		X	M	O
Cost-Limit-Information-Request	X		O	O
Cost-Limit-Information-Response		X	C	C
Item-Over-Limit		X	C1	C2
Item-Over-Limit-Response	X		C1	C2
Identification-Request		X	O	O
Identification-Response	X		O	C
Language-to-Use-Request	X		O	O
Language-to-Use-Response		X	C	O
Data-Syntax-Switching-Request		X	O	O
Data-Syntax-Switching-Response	X		O	C
Error-Message	X	X	M	M
M Mandatory O Optional C Mandatory when the related request is implemented C1 Mandatory when Cost-Limit-Information-Request is implemented C2 Mandatory when handling of limits is supported.				

7.6.1 Charging

The user in country A who accesses, through a VSU, a host machine in country B is charged for the session according to a bilateral agreed charging level (initial charging level). The host in country B may adjust the charging level by changing TBC-rate (Time-Based-Charging-rate) and/or Volume-rate. In addition, the host may charge the user for items accessed (frame price) and transactions performed (transaction price).

TBC-rate is specified in terms of period and price per period. Volume-rate is specified in terms of volume size and price per volume. All data sent and received by the VSU on the host side shall be taken into account.

All costs are expressed in the currency of the country of the host.

As state tables can be seen as a way of describing the interaction of protocol events, the state table for the charging commands received by the VSU is given in 7.9.

7.6.1.1 Charging level

For negotiation of charging level the commands Charging-Modify-Request and Charging-Modify-Response shall be used.

Application-Connection-Report is used to indicate the moment in time that the negotiated charging level becomes active.

The host may announce an Application-Connection-Report by a parameter in the Charging-Modify-Request. If such a parameter is set (start at connection report) the VSU shall delay the use of the new charging level until it has received the Application-Connection-Report. When the host has not announced the Application-Connection-Report, the VSU shall activate the new charging level upon receipt of an Application-Connection-Report command or any data packet with the Q bit set to zero.

Application-Disconnection-Report is used to reset the charging level either to the still running charging level or to the bilateral agreed charging level (initial charging level).

7.6.1.2 Description of charging commands

7.6.1.2.1 Charging-Modify-Request

This command is sent by the host to the VSU. It contains information about modifications in communication costs (TBC-rate and volume-rate) and/or application costs (TBC-rate, volume-rate, frame-price and transaction-price).

At least one modification shall be present when the command is used.

This command shall be sent whenever the charging changes and is always sent before a frame or a transaction with a cost not equal to zero.

The host shall not initiate any other charging or limit handling command until the Charging-Modify-Response has been received. The VSU shall respond either accepting or refusing the indicated charging modifications before any other protocol elements are exchanged.

7.6.1.2.2 Charging-Modify-Response

This command is sent by the VSU to the host to accept or to refuse the charging modification requested by the host.

When the VSU refuses the charging modifications, charging shall be according to the previous charging level. It is up to the host to decide whether to disconnect the session or to continue.

When the VSU accepts the charging modifications frame the new charging level shall become active upon receipt by the VSU of an Application-Connection-Report (default situation) or any data packet with the Q-bit set to zero

7.6.1.2.3 Application-Connection-Report

This command is sent by the host to the VSU when a successful connection to the selected application is established. It implicitly means that the previous accepted Charging-Modify-Request command becomes active, and it also implies an implicit Application-Disconnection-Report from the previous application if any.

This command carries one parameter which indicates the connected application.

7.6.1.2.4 Application-Disconnection-Report

This command is sent by the host to the VSU when the association with the selected application is finished or not possible. The Application-Disconnection-Report command implicitly carries a Charging-Modify-Request command setting the communication costs either to the initial value (basic tariff) or to the still running charging level.

This command carries two parameters, the first one indicating the disconnected application and the second one indicating which tariff to apply (basic or still running tariff).

7.6.2 Limit handling

The VSU in country A can inform the host in country B about the charging limits of the user of the session.

The host may accept or refuse the limit-handling or issue an error. When accepted, the host is obliged to issue an Item-Over-Limit command whenever the charges exceed the indicated limits.

All costs are expressed in the currency of the country of the host.

7.6.2.1 Description of limit handling commands

7.6.2.1.1 Cost-Limit-Information-Request

This command may be sent only at connection time by the VSU to the host. This command will be used to request the host to handle the cost limit facility. The host must accept or refuse this command or send an Error-Message command.

This command may contain information about item-cost-limit and/or session-cost-limit and/or time-based-charging-limit. The usage of the session-cost-limit needs further clarification.

7.6.2.1.2 Cost-Limit-Information-Response

This command is sent by the host to the VSU in response to a Cost-Limit-Information-Request command to indicate whether the host can handle the Cost-Limit facility or not.

This command carries one parameter which can take the value "Accept" or "Refuse".

If the value of the parameter is "Accept", it is up to the host not to deliver more information than the limit allows. If the host continues to send information which exceeds the Cost-Limit, it is the responsibility of the VSU not to charge the user for the access of the information.

7.6.2.1.3 Item-Over-Limit

This command is sent by the host to the VSU when the cost limit facility is handled by the host and when the price of frame, transaction or time-based-charging exceeds the limit fixed at connection time. This command is used only for transaction, frame or time-based-charging.

The host shall not initiate any other limit handling or charging command until the Item-Over-Limit-Response or Error-Message has been received. The VSU shall respond immediately either accepting or refusing the indicated costs.

7.6.2.1.4 Item-Over-Limit-Response

This command is sent by the VSU to the host in response to an Item-Over-Limit command to indicate whether the VSU accepts the Over-Limit price or not.

This command carries one parameter which can take the value "Accept" or "Refuse".

When the VSU accepts the costs indicated in the Item-Over-Limit command, the host shall issue a Charging-Modify-Request command in order to get the charges modified according to the accepted Item-Over-Limit command.

7.6.3 Identification

The host in country B may ask the VSU in country A to give an identification of the user of the session.

7.6.3.1 Description of Identification commands

7.6.3.1.1 Identification-Request

This command may be sent by the host to the VSU if the host needs an identification.

This request may apply to user identification, terminal identification or line identification as described in CCITT Recommendation F.300.

7.6.3.1.2 Identification-Response

This command may be sent by the VSU to the host in response to an Identification-Request command.

Identification parameters asked for by the host but not supported by the VSU shall either be absent or empty in the Identification-Response.

7.6.4 Language-Management

7.6.4.1 Language-to-Use-Request

This command may be sent by the VSU to the host at any time to indicate a list of languages in preferred order. It is used to allow a host that supports different languages for an information item to choose the instance that corresponds to the user's preference.

7.6.4.2 Language-to-Use-Response

This command may be sent by the host to the VSU in response to a Language-to-Use-Request. It indicates the language used to deliver the information item or items to follow.

7.6.5 Data-Syntax-Management

If possible the host should perform by itself the switching of the terminal. The host should use the Data-Syntax-Switching-Request command.

7.6.5.1 Data-Syntax-Switching-Request

This command may be sent by the host to the VSU at any time during the session. It contains information about the data-syntax the application will use.

7.6.5.2 Data-Syntax-Switching-Response

This command may be sent by the VSU to the host in response to the Data-Syntax-Switching-Request. This command carries one parameter which may take one of the following values:

- the requested data-syntax is not supported;
- the requested data-syntax is supported with transcoding in the VSU;
- the terminal is in the requested data-syntax.

7.6.6 Error-Message

This command shall be sent by the VSU or the host in reply to either an unrecognised command or an unrecognized parameter of the command.

Receipt of an Error-Message by the host when it is waiting for a response (Charging-Modify-Response, Item-Over-Limit-Response, Profile-Switching-Response or Identification-Response) shall finish the waiting state. It is up to the host to decide whether the session will be continued or disconnected.

Receipt of an Error-Message by the VSU when it is waiting for a response (Cost-Limit Information-Response or Language-to-Use-Response) shall finish the waiting state.

7.7 Coding

The coding of the Administrative commands is in line with the Telematic service message as defined in CCITT Recommendation X.29, subclauses 4.4.1, 4.4.2 and 4.4.11 and is illustrated in Table 2.

All protocol elements are carried via a qualified packet. Each command will appear as one single data packet sequence with Q-bit set to 1. The coding of the commands is defined in 7.8.

Table 2/T.101 – Terminal to Host administrative command format

Octet 1	40
	Command

7.8. ASN.1 encoding of the Administrative commands of Terminal to Host interworking

Terminal-to-Host-interworking-commands DEFINITIONS ::= BEGIN

terminalToHostCommand ::= CHOICE

```
{
chargingModifyRequest      [0]          ChargingModifyRequest,
chargingModifyResponse     [1]  IMPLICIT ChargingModifyResponse,
appliConnectReport         [2]  IMPLICIT AppliConnectReport,
appliDisconnectReport      [3]  IMPLICIT AppliDisconnectReport,
costLimitInformationRequest [4]  IMPLICIT CostLimitInformationRequest,
costLimitInformationResponse [5] IMPLICIT CostLimitInformationResponse,
itemOverLimit              [6]  IMPLICIT ItemOverLimit,
itemOverLimitResponse      [7]  IMPLICIT ItemOverLimitResponse,
identificationRequest       [8]  IMPLICIT IdentificationRequest,
identificationResponse      [9]  IMPLICIT IdentificationResponse,
errorMessage               [10] IMPLICIT ErrorMessage
languageToUseRequest        [11] IMPLICIT SEQUENCE OF Language,
languageToUseResponse       [12] IMPLICIT Language,
dataSyntaxSwitchingRequest  [13] IMPLICIT DataSyntaxRequest,
dataSyntaxSwitchingResponse [14] IMPLICIT INTEGER {
    requestedProfileNotSupported(0),
    requestedProfileSupportedTranscodingDoneByVSU(1),
    terminalInTheRequestedProfile(2) }
vTX                         [30] -- reserved for Syntax-Based-Videotex
}
```

ChargingModifyRequest ::= CHOICE

```
{
    predefinedTariff      [0]  IMPLICIT  INTEGER, -- to be used for bilateral agreed predefined tariffs
    nonpredefinedTariff   [1]  IMPLICIT  NonpredefinedTariff,
    cCITTpredefinedTariff [2]  IMPLICIT  CCITTpredefinedTariff,
    -- CCITTpredefinedTariff is for further study.
    startAtConnectReport [3]  IMPLICIT BOOLEAN OPTIONAL
    -- default is FALSE
    -- TRUE means wait for receipt of Application-Connection-Report
    -- FALSE means start on receipt of Application-Connection-Report
    -- or any data packet with Q = 0
}
```

NonpredefinedTariff ::= SEQUENCE

```
{
    tBCPrice      [0]  IMPLICIT  TBCPrice  OPTIONAL,
    framePrice    [1]  IMPLICIT  RealNumber OPTIONAL,
    transactionPrice [2] IMPLICIT  RealNumber OPTIONAL,
    volumePrice    [3]  IMPLICIT  VolumePrice OPTIONAL
    -- at least one of the types should appear
}
```

TBCPrice ::= SEQUENCE

```
{
    period      [0]  IMPLICIT INTEGER, -- period in seconds
    price       [1]  IMPLICIT RealNumber
}
```

```

VolumePrice ::= SEQUENCE
{
    volume [0] IMPLICIT INTEGER {
        1byte(0),
        16bytes(1),
        32bytes(2),
        64bytes(3),
        128bytes(4),
        256bytes(5),
        512bytes(6),
        1024bytes(7),
        2048bytes(8),
        4096bytes(9) },
    price [1] IMPLICIT RealNumber,
}

ChargingModifyResponse ::= BOOLEAN -- "Accept"=TRUE
-- "Refuse"=FALSE

AppliConnectReport ::= SEQUENCE
{
    applicationConnectionId [0] IMPLICIT OCTETSTRING
}

AppliDisconnectReport ::= SEQUENCE
{
    applicationDisconnectionId [0] IMPLICIT OCTETSTRING
    applicableTariff [1] IMPLICIT BOOLEAN DEFAULT TRUE
    -- TRUE means basic tariff
    -- FALSE means still running tariff
}

CostLimitInformationRequest ::= SEQUENCE
{
    itemCostLimit [0] IMPLICIT RealNumber OPTIONAL,
    sessionCostLimit [1] IMPLICIT RealNumber OPTIONAL,
    tBCPriceLimit [2] IMPLICIT TBCPrice OPTIONAL,
    -- at least one of the types should appear
}

CostLimitInformationResponse ::= BOOLEAN -- "supported"=TRUE
-- "not supported"=FALSE

ItemOverLimit ::= SEQUENCE
{
    framePrice [0] IMPLICIT RealNumber OPTIONAL,
    transactionPrice [1] IMPLICIT RealNumber OPTIONAL,
    proposedTBCPrice [2] IMPLICIT TBCPrice OPTIONAL,
    -- at least one of the types should appear
}

ItemOverLimitResponse ::= BOOLEAN -- "Accept"=TRUE
-- "Refuse"=FALSE

IdentificationRequest ::= SEQUENCE
{
    identificationCode [0] IMPLICIT IdentificationCode
    -- at least one identificationCode should appear
}

IdentificationResponse ::= SEQUENCE OF Identification

```

```

Identification ::= SEQUENCE OF
{
    identificationCode [0] IMPLICIT IdentificationCode,
    identificationContents [1] IMPLICIT OCTETSTRING
}
IdentificationCode ::= INTEGER
{
    userIdentification(15),
    lineIdentification(16),
    terminalIdentification(17)
    -- other identification codes may be used on
    -- a bilateral base; it is suggested that,
    -- when applicable, values corresponding
    -- to field-type of Annex B/T.541 shall be used.
}
ErrorMessage ::= INTEGER
{
    unrecognisedCommand(0),
    unrecognisedParameter(1)
}

Language ::= IMPLICIT OCTETSTRING
    -- value as defined in ISO 639 Annex B, alphabetical list of
    -- letter symbols for languages. For example, Dutch language is
    -- represented by "NL".

DataSyntaxRequest ::= CHOICE
{
    dataSyntaxISwitching [1] IMPLICIT OCTETSTRING,
    -- value as defined in Annex B
    dataSyntaxIISwitching [2] IMPLICIT OCTETSTRING,
    -- value as defined in Annex C
    dataSyntaxIIISwitching [3] IMPLICIT OCTETSTRING
    -- value as defined in Annex D
}

RealNumber ::= SEQUENCE
{
    integerPart [0] IMPLICIT INTEGER DEFAULT 0,
    decimalExponent [1] IMPLICIT INTEGER DEFAULT 2
    -- the encoded real number is obtained by dividing the
    -- integerPart by 10**decimalExponent
}
END

```

7.9 State table for charging events received at VSU side

7.9.1 Used abbreviations

7.9.1.1 States:

ST_RAA	Basic charging level running, other charging levels absent.
ST_RPA	Basic charging level running, first charging level proposed, second charging level absent.
ST_SRA	Basic charging level sleeping, first charging level running, second charging level absent.
ST_SRP	Basic charging level sleeping, first charging level running, second charging level proposed.
ST_SSR	Basic charging level sleeping, first charging level sleeping, second charging level running.

7.9.1.2 Actions

- [1] Add frame-price and/or transaction-price to user account.
- [2] Finish charging time period; install proposed charging level and start new time period.
- [3] Finish charging time period; reinstall first charging level and start new time period.
- [4] Finish charging time period; reinstall basic charging level and start new time period.
- [5] Store proposed charging level according to the received values, set V_activate_on_ACR according to the received value.

NOTE – When transaction or frame price is absent, then value 0 is assumed. When volume or TBC-rate is absent, then the running values are assumed.

- [6] Copy second charging level to first charging level.

7.9.1.3 Incoming events

CMreq Charging-Modify-Request

ACR Application-Connection-Report

ADR Application-Disconnection-Report

Data0 Any data with Q-bit = 0;

Data1 Any other data with Q-bit = 1.

7.9.1.4 Outgoing events

CMrsp+ Charging-Modify-Response “Accepted”

CMrsp– Charging-Modify-Response “Refused”

7.9.1.5 Predicates

P0 Charging modifications can be accepted

P1 V_activate_on_ACR

P2 Basic tariff required

7.9.1.6 Variables

V_activate_on_ACR TRUE when charging level is only to become active on receipt of an Application-connection-Report.

7.9.2 State table

State Stack: second first basic Event	ST_RAA absent absent running	ST_RPA absent proposed running	ST_SRA absent running sleeping	ST_SRP proposed running sleeping	ST_SSR running sleeping sleeping
CMreq	P0: CMrsp+ [5] ST_RPA ^P0: CMrsp- ST_RAA	P0: CMrsp+ [5] ST_RPA ^P0: CMrsp- ST_RPA	P0: CMrsp+ [5] ST_SRP ^P0: CMrsp- ST_SRA	P0: CMrsp+ [5] ST_SRP ^P0: CMrsp- ST_SRP	P0: CMrsp+ [6][5] ST_SRP ^P0: CMrsp- ST_SSR
ACR	ST_RAA	[1] [2] ST_SRA	ST_SRA	[1] [2] ST_SSR	ST_SSR
ADR	ST_RAA	ST_RAA	P2: [4] ST_RAA ^P2: ST_SRA	P2: [4] ST_RAA ^P2: ST_SRA	P2: [4] ST_RAA ^P2: [3] ST_SRA
Data0	ST_RAA	P1: ST_RPA ^P1: [1] [2] ST_SRA	ST_SRA	P1: ST_SRP ^P1: [1] [2] ST_SSR	ST_SSR
Data1	ST_RAA	ST_RPA	ST_SRA	ST_SRP	ST_SSR

7.9.3 Collision

When sending of Limit Cost Information clashes with receipt of a Charging-Modify-Request and the costs being modified exceed the user limits the following sequence is exchanged:

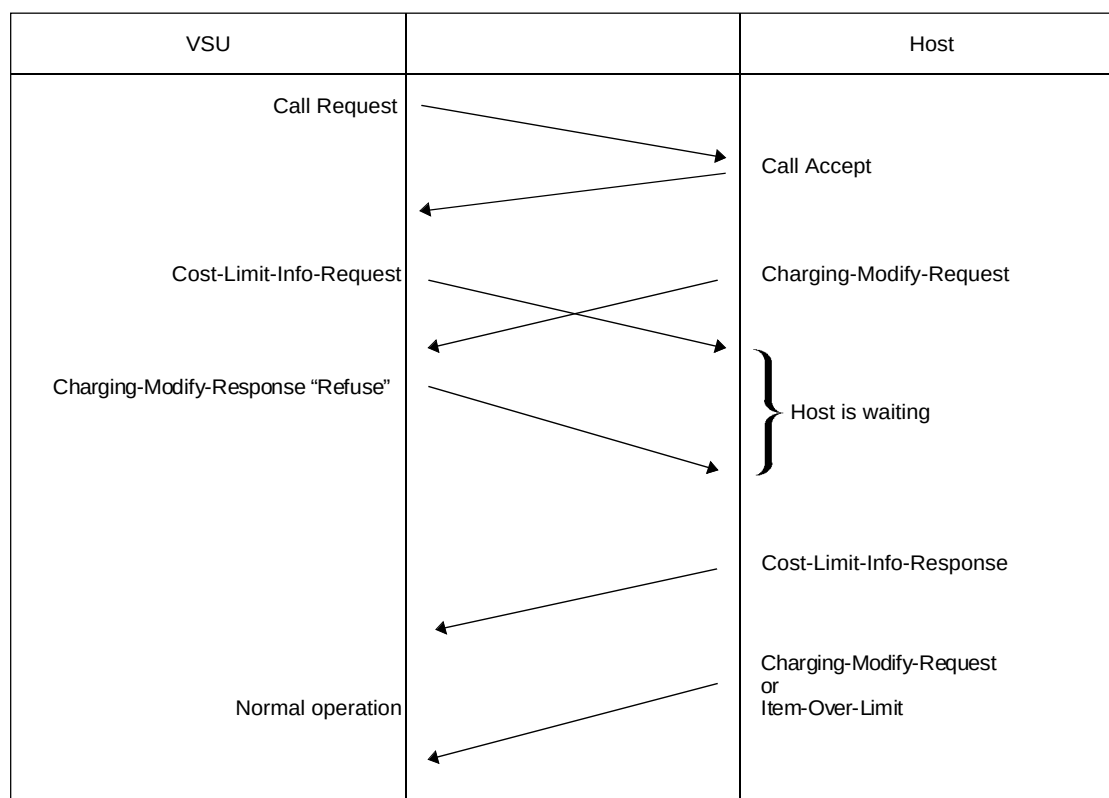
8 Content architecture class attributes

8.1 Content architecture class

The value of the attribute “content architecture class” of a basic component description that conforms to this Recommendation is an ANS.1 object identifier with the value: { 0 1 8 16 3 }.

8.2 Content type

The content architecture class attribute “content type” cannot be used to specify the content architecture class defined in this Recommendation.



T0817260-94/d007

9 Content portion attributes

9.1 Type of coding

Classification	Defaultable
Applicability	Videotex content architecture class
Structure	ASN.1 object identifier
Permissible values	ASN.1 object identifier
{ 0 1 8 16 4 }	for "IDS encoding"
{ 0 1 8 16 5 }	for "Data syntax I encoding"
{ 0 1 8 16 6 }	for "Data syntax II encoding"
{ 0 1 8 16 7 }	for "Data syntax III encoding"
Default value	"IDS encoding"
Definition	For Videotex interworking, the possible values correspond to the data syntaxes described in Annexes A, B, C and D.

9.2 Specific coding attributes

These attributes provide additional information required for encoding/decoding the content information, as well as other information intrinsic to the content portion and type of coding.

9.2.1 Subset

Classification	Defaultable
Applicability	Videotex content architecture class Type of coding “IDS encoding”
Values	Integer [0, 1 to 5, 81 to 92]
Default value	0
Definition	This attribute identifies the subset (rank or profile) used within the IDS. Value 0 is used when no subset is specified.

9.2.2 Rank

Classification	Defaultable
Applicability	Videotex content architecture class Type of coding “Data syntax I encoding”
Values	Integer [0, 1 to 5]
Default value	0
Definition	This attribute identifies the rank used within the data syntax I. Value 0 is used when no rank is specified.

9.2.3 Profile

Classification	Defaultable
Applicability	Videotex content architecture class Type of coding “Data syntax II encoding”
Values	Integer [0, 81 to 92]
Default value	0
Definition	This attribute identifies the profile used within the Data syntax II. Value 0 is used when no profile is specified.

10 Formal definition of Videotex dependent data type

10.1 Introduction

This subclause contains formal definition in ANS.1 notation (defined in Recommendation X.208) of the data type corresponding to attributes applicable to Videotex.

The data type is:

- the data type to represent specific coding attributes.

```
Videotex-coding-attributes ::= CHOICE
{
    subset      [0]  IMPLICIT Subset OPTIONAL
    rank       [1]  IMPLICIT Rank  OPTIONAL
    profile    [2]  IMPLICIT Profile OPTIONAL
}
Subseta) ::= INTEGER
{
    undefined   (0)
    rank 1     (1)
    rank 2     (2)
    rank 3     (3)
    rank 4     (4)
    rank 5     (5)
```

^{a)} Use of subset IDS is for further study.

```

        profile 1      (81)
        profile 2      (82)
        profile 3      (83)
        profile 4      (84)
        profile X1-1    (85)b)
        profile X1-2    (86)
        profile X1-3    (87)
        profile X1-4    (88)
        profile X2-1    (89)
        profile X2-2    (90)
        profile X2-3    (91)
        profile X2-4    (92)
    }
    Rank ::= INTEGER
    {
        undefined      (0)
        rank 1          (1)
        rank 2          (2)
        rank 3          (3)
        rank 4          (4)
        rank 5          (5)
    }
    Profile ::= INTEGER
    {
        undefined      (0)
        profile 1      (81)
        profile 2      (82)
        profile 3      (83)
        profile 4      (84)
        profile X1-1    (85)
        profile X1-2    (86)
        profile X1-3    (87)
        profile X1-4    (88)
        profile X2-1    (89)
        profile X2-2    (90)
        profile X2-3    (91)
        profile X2-4    (92)
    }
}

```

10.2 Summary of ASN.1 object identifiers

See Table 3.

Table 3/T.101 – ASN.1 object identifiers

Videotex document application profile	0 1 8 16 0
DM1 communication application profile	0 1 8 16 1
Videotex operational application profile	0 1 8 16 2
T.101 Content architecture class	0 1 8 16 3
Type of coding	
IDS	0 1 8 16 4
Data syntax I	0 1 8 16 5
Data syntax II	0 1 8 16 6
Data syntax III	0 1 8 16 7
Application context	0 1 8 16 8

^{b)} Profile Xi-j: geometric profile Xi together with alphamosaic profile j.

11 Common enhancements to data syntaxes I, II and III

The coding of the Videotex information conforms to one of the data syntaxes contained in Annexes B, C and D. In addition, Annex E, Audio data syntax, Annex F, Photographic data syntax and Recommendation T.107 (VEMMI) contain extensions which are independent from the three data syntaxes. The following subclauses define two common extensions, namely, a general terminal identifier function and a switching mechanism based on ISO 9281.

11.1 Terminal identifier

A terminal identifier may be used to ascertain the capabilities of the terminal by a Videotex service. Videotex terminals have not all been designed to support all coding techniques; for example, photographic and audio, or support of different types of local facilities such as telesoftware or various types of modems. In addition, a specific terminal may support one or more of the defined base data syntaxes DS I, DS II or DS III.

The purpose of the terminal identifier is to allow a host to determine whether a terminal can support a given facility.

The operation of the terminal identifier is in two steps. First a unique code sequence is sent to the terminal, to which the terminal responds with a defined message identifying the particular terminal facilities available.

Each of the national or regional Videotex services may already define a terminal facility identifier facility. This facility can always be used in accordance with the application rules defined in T.101, Annexes B, C and D.

A common terminal facility identifier is defined below. This identifier is proposed to be of the form:

US 2/0 4/0

where US is the C0 code (1/15).

NOTE – US x/y x/y is equivalent to NSR x/y x/y in DS I and DS III. On a DS I or DS III terminal the issuing of this terminal identifier command has the effect of resetting the presentation display parameters as well as the terminal to issue a terminal identifier response.

The form of the response message is described below and additional details are contained in the various data syntaxes.

11.1.1 Terminal identification command response messages

The terminal identifier command is sent to the terminal during the communication phase causing the terminal to respond with an identification message.

NOTE 1 – Some systems, notably DS I systems, divide interchange into a connection and a communication phase.

Host

Terminal

Identification Command: US 2/0 4/0 ----->

<----- US 2/0 x/y...x/y 4/0

Terminal identifier
Response message

NOTE 2 – An optional non-interfering character may follow a terminal identification command, notably DS I systems.

If no response is received from a terminal within a specific time (depending upon the transmission network being used), then the basic terminal used within that network is assumed.

The terminal identifier response message takes the form similar to the terminal identifier command. Intermediate codes identifying the capabilities of the terminal and may be selected from the following list :

- 4/0 = terminator code of response message
- 4/1-4/15 and 6/0-7/15 = reserved for internal use of each data syntaxes DS I, DS II and DS III
- 5/0, 5/1 = audio mode (as defined in 12.1.3)
- 5/2 = modem capability (as defined in 12.1.2)
- 5/3 = reserved
- 5/4 = data syntax and profile (rank/facility) identification:
 - data syntax
 - 3/1 = DS I
 - 3/2 = DS II
 - 3/3 = DS III
 - :
 - :
 - 3/14 = private
 - 3/15 = terminator code
- 5/5 = photographic mode (as defined in 12.1.4)
- 5/6 = support of ISO 9281 switching
- 5/7 = support of VEMMI data
- 5/8-5/15 = reserved

The values in the range 4/1-4/15 and 6/0-7/15 terminated by the code 3/15 are used for the profile identification. The codes following a particular data syntax identification pertain to that particular data syntax.

NOTE 3 – Identification of common text facilities based on Recommendations T.50, T.51 and T.52 is for further study.

If a terminal were to receive a terminal facility identifier response message, due to echo by a PAD, the terminal may ignore the response message beginning with the US 2/0 command up to the 4/0 terminator (or up to any other presentation C0 character). An optional data forwarding character (e.g. CR) may follow the terminal response message in case of operation via an X.3 PAD.

11.1.2 Modem capability

When receiving a profile request US 2/0 4/0, the terminal supporting the modem capability shall issue together with the other codes used for identifying profiles or parts of the SRM, or an audio capability, the modem conformance code followed by one or more byte(s) in order to indicate the type of modems supported by the terminal.

The modem conformance code has been assigned as follows:

- 5/2 modem capability

Each byte following the modem conformance code shall take one of the following values:

- 3/0 Not applicable or unknown capability (default)
- 3/1 No modem used
- 3/2 Modem running in asynchronous mode, according to the following specification:
 - 4/0 Reserved code (terminator code)
 - 4/1 Unknown speed (default)
 - 4/2 Modem according to CCITT Recommendation V.21
 - 4/3 Modem according to CCITT Recommendation V.22
 - 4/4 Modem according to CCITT Recommendation V.22 *bis*
 - 4/5 Modem according to CCITT Recommendation V.23
 - 4/6 Modem according to CCITT Recommendation V.32

- 4/7 Modem according to CCITT Recommendation V.32 *bis*
- 4/8 Modem according to CCITT Recommendation V.26 *ter*
- 4/9 Modem according to ITU-T Recommendation V.34
- 4/10-4/13 Reserved codes
- 4/14 Reserved code for non-ITU-T modem
- 4/15 Reserved code
- 3/3 Modem running in synchronous mode, according to the following specification:
 - 4/0 Reserved (terminator code)
 - 4/1 Unknown speed (default)
 - 4/2 Modem according to CCITT Recommendation V.26 *bis*
 - 4/3 Modem according to CCITT Recommendation V.26 *ter*
 - 4/4 Modem according to CCITT Recommendation V.27 *ter*
 - 4/5 Modem using the modulation technique described in CCITT Recommendation V.29
 - 4/6 Modem according to CCITT Recommendation V.32
 - 4/7 Modem according to CCITT Recommendation V.32 *bis*
 - 4/8 Modem according to CCITT Recommendation V.17
 - 4/9 Modem according to CCITT Recommendation V.22
 - 4/10 Modem according to CCITT Recommendation V.22 *bis*
 - 4/11 Modem according to ITU-T Recommendation V.34
 - 4/12-4/13 Reserved codes
 - 4/14 Reserved code for non-ITU-T modem
 - 4/15 Reserved code
- 3/4 Modem using an embedded error correction or compression mechanism, whose type(s) is (are) identified by the following byte(s):
 - 4/0 Reserved (terminator code)
 - 4/1 Unknown mechanism
 - 4/2 Correction mechanism as described by CCITT Recommendation V.42
 - 4/3 Compression mechanism as described by CCITT Recommendation V.42 *bis*
 - 4/4-4/15 Reserved codes.

The compression or correction mechanism only applies to the immediate preceding defined modem type. In this case, the asynchronous or synchronous type of any additional modem shall be repeated.

Example

When receiving a profile request, a terminal which conforms to DS II alphasaid profile 3 and may use either a V.23 modem, or a V.32 and V.42 modem, or a V.27 *ter* modem shall transmit:

```
US 2/0 5/4 3/2 6/2 3/15 5/2 3/2 4/5 3/3 4/6 3/4 4/2 3/3 4/4 4/0
3/5-3/14 reserved
3/15 forbidden (ITU-T terminator code for alphasaid capabilities).
```

11.1.3 Audio capabilities

When receiving a profile request US 2/0 4/0, the terminal supporting an audio capability shall issue, together with the other codes audio conformance code(s) followed by zero, one or more couple(s) of octets indicating respectively the algorithm supported and the audio bitrate.

The audio conformance codes have been assigned as follows:

- 5/0 Audio capability, block mode (as described in Annex E)
- 5/1 Audio capability with framing (as described in CCITT Recommendation H.221)

Algorithm identification:

3/0	PCM A-law described in CCITT Recommendation G.711
3/1	PCM μ -law described in CCITT Recommendation G.711
3/2	ADPCM described in CCITT Recommendations G.721 [11] and G.723
3/3	Sub-band ADPCM described in CCITT Recommendation G.722
3/4	RPE-LTP coding method
3/5	Near instantaneous (CCITT Recommendation J.41)
3/6	Sub-band ADPCM (CCITT Recommendation J.42)
3/7	MPEG Audio (ISO CD 11172-3)
3/8-3/14	Reserved for future use
3/15	Forbidden (ITU-T terminator code for alphamosaic capabilities)

Bitrate audio coding:

3/0	8 kbit/s
3/1	16 kbit/s
3/2	24 kbit/s
3/3	32 kbit/s
3/4	40 kbit/s
3/5	48 kbit/s
3/6	56 kbit/s
3/7	64 kbit/s
3/8	13 kbit/s
3/9	2.4 kbit/s (provisional)
3/10	4.8 kbit/s (provisional)
3/11	128 kbit/s (provisional)
3/12	192 kbit/s (provisional)
3/13	384 kbit/s (provisional)
3/14	256 kbit/s (provisional)
3/15	Forbidden (ITU-T terminator code for alphamosaic capabilities)

Example

When receiving a profile request, a terminal which conforms to the alphamosaic, geometric and photographic parts of DS II and which supports the audio capability in block mode using PCM A-law 64 kbit/s or ADPCM 32 kbit/s shall transmit:

US 2/0, 5/4, 3/2, 4/1, 4/2, 4/3, 3/15, 5/0, 3/0, 3/7, 3/2, 3/3, 4/0

11.1.4 Photographic capabilities

When receiving a profile request US 2/0 4/0, the terminal supporting a photographic capability shall issue, together with the other codes, one or more of the conformance codes.

Photographic conformance codes:

5/5	photographic conformance code.
-----	--------------------------------

This conformance code may be followed by one or more codes identifying the profile(s) in use:

3/0	Reserved
3/1	Compatible photographic profile P1 (CIF, sequential)
3/2	Compatible photographic profile P2 (P1 + 2-1-1 sequential)
3/3	Compatible photographic profile P3 (P2 + 4-2-2 sequential)

3/4	Compatible photographic profile P4 (P3 + spectral selection c mode)
3/5	Compatible photographic profile P5 (P4 + successive approximation progressive mode)
3/6-3/13	Reserved
3/14	Private choice of photographic profile P _{PRIV}
3/15	Forbidden (ITU-T terminator code for alphamosaic capabilities)

For a given profile, it is possible to indicate if the terminal has a specific display capability whose type is identified by one or more bytes from the following list:

4/0	reserved (terminator code)
4/1	monochrome capability
4/2	T.4 decoding capability
4/3	T.6 decoding capability
4/4-4/15	Reserved.

It is assumed that, in absence of any additional information, the terminal has a colour display capability. In addition, a monochrome terminal shall always be able to process received colour information and to have it displayed in monochrome.

Example

A terminal supporting all of DS I profile (rank) l, m, n and DS II profiles 1, 2, 3, P1, P2, VT52 and DS III profiles (facilities) i, j, k together with audio ADPCM 32 kbit/s/s and V.21 and V.26 *bis* modems, the ISO 9281 switching, photo facility (P1 monochrome with T4 and T6 option, P2) and VEMMI data, would generate the response message:

```

US 2/0 5/4 3/1 6/1 6/m 6/n 3/15
3/2 6/0 6/1 6/2 7/2 7/3 7/14 4/1 3/15
3/3 6/i 6/j 6/k 3/15
5/0 3/2 3/3
5/2 3/2 4/2 3/3 4/2
5/5 3/1 4/1 4/2 4/3 3/2
5/6
5/7
4/0

```

11.2 ISO 9281 support

The ISO 9281 switching is used to switch a terminal in the audio mode, the photographic mode or VEMMI mode. Upon a terminal identifier request, a terminal supporting the audio and/or the photographic mode and/or the VEMMI mode should include the code 5/6 in its response message.

The base Videotex data syntaxes defined in Annexes B, C and D are based primarily on ISO 2022 for code extension. This is appropriate for the text and graphic components of the Videotex data syntaxes. The common extensions for audio and photographic defined in Annexes E and F and VEMMI defined in Recommendation T.107 require the use of a more general picture coding extension mechanism. This mechanism is common to all Videotex systems. If a terminal indicates by a TFI code that it supports ISO 9281, then it must ignore the data corresponding to any picture coding environment that it does not implement.

The detailed choices of the use of ISO 9281 for Videotex are given in Annexes E, F and in ITU-T Recommendation T.107. The general structure of the ISO 9281 coding mechanism is given below. Reference should be made directly to the ISO Standard for the complete description of the coding mechanism.

ISO 9281 allows the definition of picture coding environment, where the term picture is broadly defined to include such data as audio. A picture coding environment may be established in several ways; however in CCITT Telematic services, a picture coding environment is established from a base ISO 2022 coding environment. A picture coding environment

may only be established from within an ISO 2022 “complete coding environment” that is after the use of an ISO 2022 ESC 2/5 F command. Each of the base Videotex data syntaxes are established by the use of such an ESC 2/5 F code as listed below:

- ESC 2/5 4/3 establish Data Syntax I (per Annex B);
- ESC 2/5 4/4 establish Data Syntax II (per Annex C);
- ESC 2/5 4/1 establish Data Syntax III (per Annex D).

Once within the complete coding environment defined by each of the base Videotex data syntaxes, a picture coding environment may be established by a command with the following structure :

PCD	CMI	LI
Picture Coding Delimiter	Coding Method Identifier	Length Indicator

The Picture Coding Delimiter is coded as ESC 7/0.

The Coding Method Identifier consists of two codes, the Picture Mode (PM) and the Picture Identifier (PI) which are both registered and identify a particular picture coding method. The Length Indicator (LI) indicates the length of the following Picture Data Entity.

The Length Indicator (LI) shall comprise one or more bytes and has the structure shown below,

where

- b_8 = X ; indifferently ZERO or ONE
- b_7 = ONE
- b_6 = Extension flag
- b_5 to b_1 = specify the number of data bytes (see Figure 7).

b_8	b_7	b_6	b_5	b_4	b_3	b_2	b_1
X	ONE	ZERO or ONE					

Figure 7/T.101 – Structure of the LI field

Bit b_8 of each byte shall be ignored. Bit b_7 of each byte shall be set to ONE. Bit b_6 of each byte shall be the Extension Flag. The LI value is specified in binary notation as an unsigned number using bits b_5 to b_1 with the weights 2^4 , 2^3 , 2^2 , 2^1 and 2^0 respectively. If the value of LI is less than or equal to 31 it shall be represented by one byte and the Extension Flag shall be set to ZERO. If the value of LI is greater than 31 it shall be represented by more than one byte. The most significant part of this value shall be recorded in the first byte. The Extension Flag shall be set to ONE in all bytes except the last where it shall be set to ZERO.

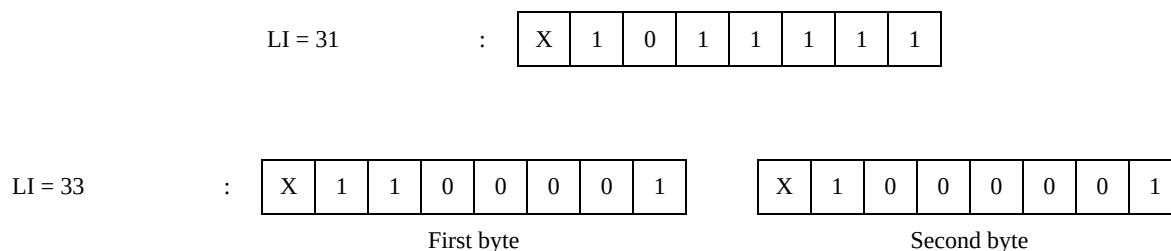
Values of LI

1 byte: $0 \leq LI \leq 2^5 - 1$

2 byte: $2^5 \leq LI \leq 2^{10} - 1$ **Recommendation T.101 (11/94)**

: : : :

Examples of coded representation



If a coding method identifier is not supported by a particular Videotex system, then the data corresponding to the picture data entity, up to the end point specified by the Length Indicator should be ignored.

The global ISO 9281 switching mechanism is described in Figure 8:

11.3 Abnormal truncation of data

Videotex systems can operate over a number of different communication facilities. In addition to basic operation over the PSTN, Recommendations exist for operation over the ISDN network in both circuit and packet switched mode as well as the PSTN in packet mode. In these additional configurations, lower layers communication protocols handle data integrity and provide the capability to support interactive dialogue functions such as the support of a user-initiated cancellation of the transmission. In the operation of Videotex over the existing PSTN network, using ISO 9281 transparent data, the handling of an abnormal termination interactive dialogue function is more complex since no lower layers facilities are available. It is recommended that only relatively short blocks of data are transmitted under a particular ISO 9281 length indicator, thereby minimising the required padding following an abnormal termination interrupt sequence to re-synchronise the server and the terminal. A recommended block length is given in Annex E Audio Data Syntax and in Annex F, Photographic Data syntax.

Optionally, an additional mechanism is provided to handle truncation of a stream of data over an asynchronous communication channel, including X.29 through an X.3 PAD. This mechanism provides a quicker response time than waiting until a block of transparent data is padded to completion.

For the case of using translation mechanism modes 2 (3-in-4 coding) and 4(shift scheme 7-bit) of Annexes E and F and for the case of using Videotex system dependent link layer procedures, the following mechanism should be used:

The Videotex system will send the character DC4 (1/4) to the terminal as an answer to the user's interruption. After receiving the DC4, the terminal is ready to process any other valid data string starting with a data syntax dependent introduce. No specific default case can be assumed. The rules follow the global switching mechanism, see Figure F.1.

The receipt of DC4 in the photographic mode is equivalent to the receipt of a PE with PDE1 = 5/3 and containing one byte (length=1). In the case of a photographic header, the whole header is discarded. In the case of photographic data, it signals the end of the picture.

12 Definition of VideotexString within ASN.1

12.1 General

ASN.1 contains several built-in data types relating to a string of textual characters. This clause defines the meaning of VideotexString.

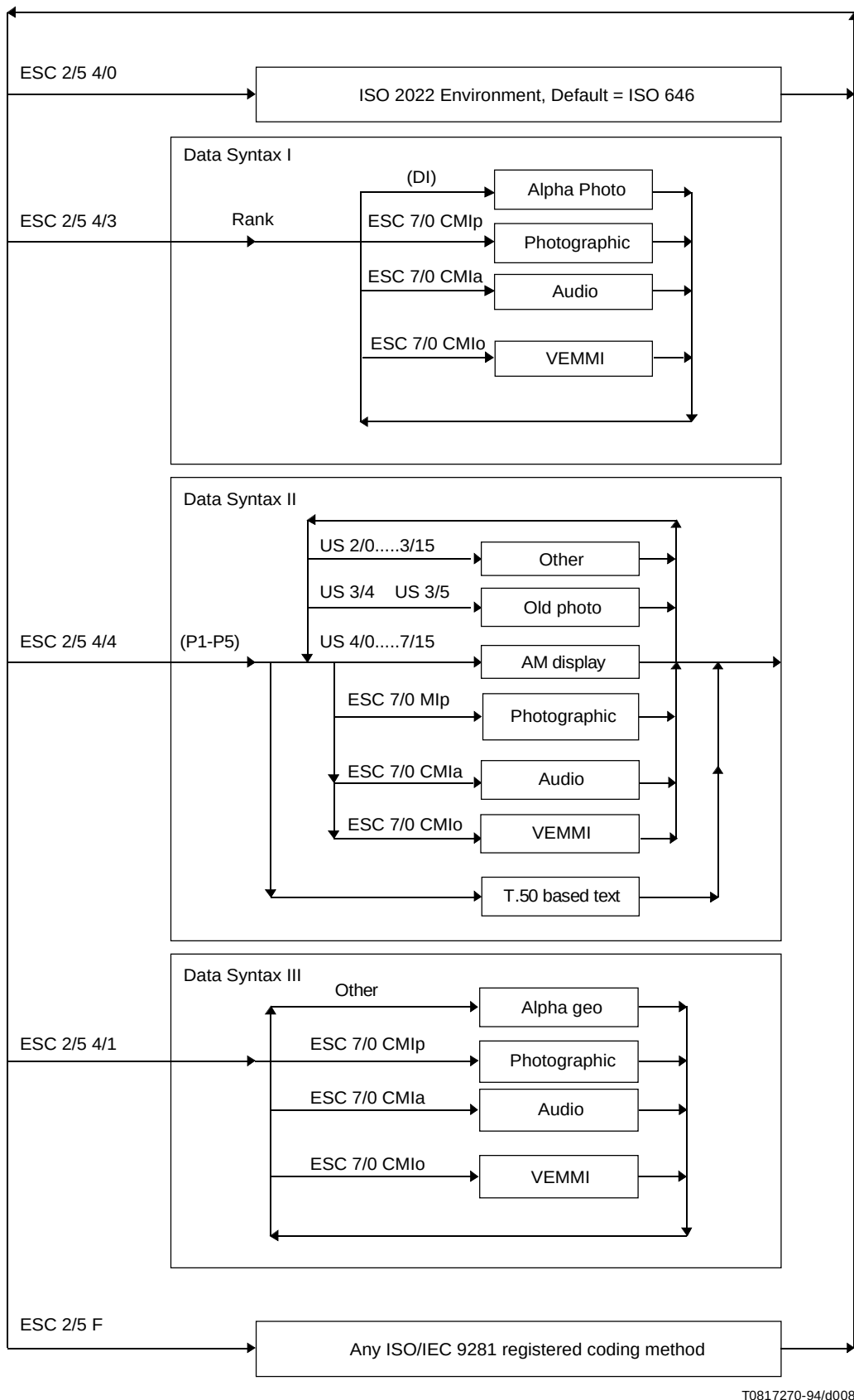


Figure 8/T.101 – Global switching mechanism

12.2 Definition of VideotexString

A VideotexString within ASN.1 consists of a string of characters selected from the data syntaxes I, II and III of this Recommendation or from the annexes defining common Videotex components (audio, photographic). These are defined in registration 131, 145 and 108 under ISO 2375. The common component of audio and photographic are registered under ISO 9281, part II (TBD). Within the context of a single T.101 data syntax, it is not mandatory to make use of the ESC 2/5 F control sequence.

12.3 ASN.1 syntax element definition

name	VideotexString
universal tag	21
registered sets	131, 145 and 108

Annex A

Interworking Data Syntax (IDS) described in ASN.1 (Recommendation X.208)

(This annex forms an integral part of this Recommendation)

Preamble

For Videotex interchange:

- a) If two countries implement the same data syntax, then interworking can use the same data syntax (DS I, or DS II, or DS III).
- b) If two countries implement two different data syntaxes, then interworking can use either:
 - i) the interworking data syntax (IDS) as defined herein; or
 - ii) any one of the three data syntaxes and convert directly between DS I/DS II/DS III. The data syntaxes may be identified by the ESC 2/5 F mechanism described in 4.4.2 of the main body of this Recommendation.

If the IDS is used, then the Administration in which country the data base is located will be responsible to convert into the IDS and the Administration in which country the user terminal is located will be responsible to convert from the IDS.

If the direct conversion method is used instead of using the IDS, the IDS would serve as a technical guide in designing the conversion process.

The IDS is not intended to be used in terminal to host communications.

A.1 Videotex page

A Videotex page in the interworking data syntax (IDS) is a sequence of presentation commands expressed in a manner independent of any of the terminal data syntaxes. This formulation of the presentation information which composes a Videotex page is intended to aid interworking between basically different terminal data syntaxes. It does this by isolating the unique and common elements between each of the data syntaxes. The interworking data syntax is not meant to be used as a terminal data syntax in its own right. One encoding of the interworking data syntax is that defined in Recommendation X.209. Other types of encoding are for further study.

Videotex-Page ::= SEQUENCE OF Presentation-Commands

Presentation-Commands ::= SEQUENCE { State-Vector, Function-&-Parameters }

A.2 State vector

A state vector is defined along with each presentation command to establish the relationship of that presentation command to each other presentation command. Although the information explicitly contained in the state vector is also implicitly contained within each presentation command, it would require the conversion apparatus to fully understand each of the three terminal oriented data syntaxes to uncover this information. Therefore a state vector is included with each presentation command in which a global state is affected or in which a boundary value is encountered, so that the conversion process might operate on a general level.

State-Vector ::= CHOICE {[1] **Vector-Definition**
[2] **Reset-State-Vector**,
[3] **NULL** }

A.2.1 Vector definition

Vector-Definition ::= SEQUENCE { **Global-State-Affected-Indicator**,
Terminal-Model-Precedence,
Boundary-Condition-Definition }

-- Only that information which changes between state vectors need be communicated. If there is no change in a particular component of the state vector, then that component need not be communicated. This means that the state vector is not communicated often and does not introduce significant overhead.

A.2.1.1 Global state affected indicator

The global state affected indicator carries information relating to the global states of the presentation data syntax. Global state variables are variables representing those states of the presentation data syntax which are established by presentation commands and which carry on to affect the results of subsequent presentation commands. By declaring the global state variables explicitly, it is not necessary for the conversion process to understand the interrelationship between presentation commands. This means that the conversion process does not have to simulate a terminal of the source data syntax in order to handle the conversion of its elements.

The global state affected indicator does not carry information about the value to which a global state has been set. That information is carried within the 'Function and Parameter' section of the IDS. The indicator merely identifies which states have changed. This is of great importance in situations where it is necessary for the conversion process to sort the presentation commands to account for differences in the terminal model used in the source and destination of the interchange. If the order of presentation commands is altered, the conversion process must establish the appropriate global variables before each command in the altered sequence. By referring to the global state affected indicator, the conversion process can determine which global states must be re-established. For example, if a sorting of presentation commands is necessary to convert from a multi-plane to a single plane terminal mode, and colour control commands have been used, then the global state affected indicator will indicate that the appropriate colour state must be established ahead of each portion of the sorted data.

In some of the terminal data syntaxes attributes have global effects while in other data syntaxes the effects are localized according to the particular display primitive type. For example, in Data Syntax III a colour command remains in effect for all primitives, until the next colour command, whereas in Data Syntax II there are various colour states which apply independently to different primitives, such as LINE colour, FILLED AREA colour, etc. The global state indicator carries a reference to a number of independent 'attribute state vectors' which define the attribute context. For those data syntaxes which make use of only global parameters, only one 'attribute state vector' need be referenced. For other data syntaxes which make use of multiple localized attributes, several 'attribute state vectors' may be referenced.

Global-State-Affected-Indicator ::= SEQUENCE {
attribute-state-vector-reference INTEGER,
attribute-affected-indicators SEQUENCE OF {
INTEGER {
current-text-position (1),
current-foreground-colour (2),
current-auxiliary-colour (3),
lining-state (4),
flash-blink-state (5),
basic-char-size-state (6),
conceal-state (7),
char-invert-box-state (8),

char-marking-state	(9),
screen-protection-state	(10),
display-control-state	(11),
device-control-state	(12),
cursor-control-state	(13),
geometric-control-1-state	(14),
geometric-control-2-state	(15),
wait-state	(16),
general-text-state	(17),
p-text-state	(18),
geometric-text-state	(19),
DRCS-definition-state	(20),
macro-definition-state	(21),
texture-pattern-state	(22),
music-part-memory-state	(23),
animation-configuration-state	(24),
workstation-configuration-state	(25)

}}}

- *The Global State Affected Indicator consists of a set of indicator flags which identify particular global states or in some cases categories of global states which may be altered by presentation and control commands. Some states, such as all the forms of Flashing and Blink processes, are grouped together for simplicity.*

A.2.1.2 Terminal model

The terminal model differs significantly between the various terminal data syntaxes. For the presentation of static images this manifests itself in the manner by which presentation commands overlay each other. A picture developed for a multi-plane terminal model can be represented on a terminal using a single-plane terminal model or a multi-plane terminal model with a different order of precedence for the planes, by sorting the presentation commands so that they build up an equivalent picture. The sorting operation is necessary since otherwise the buildup order might conflict with the precedence order in the new environment. The terminal model precedence indicator is simply a numeric indicator of the overlay precedence for presentation commands intended by the source terminal data syntax. The conversion process is independent of the terminal model or a particular data syntax and simply sorts presentation commands based on this indicator. Note that certain commands such as resets which have an effect in more than one plane of a terminal model might have to be repeated in different parts of the presentation sequence after the sort. The terminal model precedence indicator consists of a sequence of numbers to indicate the effect of a command across the terminal model.

Terminal-Model-Precedence ::= INTEGER

- *The terminal model precedence indicator is a number which indicates the order of precedence of the identified presentation information. The number '1' indicates that the identified information is of highest precedence and should be placed in front of any other information currently displayed. The number '2' indicates that the identified information is of the second level of precedence and should be placed behind any level '1' information but ahead of any other level information. For example, Data Syntax II Text and Mosaic data is level '1' information, whereas geometric information may be level '1' or level '2'. For Data Syntax III, all of the information is at level '1' since the precedence order by which it is displayed is determined only by the order in which the information is communicated. For Data Syntax I, the order is not fixed since certain 'planes' of memory may be changed in precedence by the ASSIGN FRAME command.*
- *The value '0' has special meaning for the Terminal Model Precedence Indicator. It indicates that the identified information requires special interpretation. Such special information includes partial reset commands which affect more than one layer of the terminal model, as well as commands having a time dependent effect, specifically WAIT, the BEL character, and REVEAL.*

A.2.1.3 Boundary conditions

Boundary condition variables represent the limits within which the particular presentation command has been defined. Each presentation command takes on its normal interpretation only within a certain range of values. For example, the number of characters which may be displayed on the screen varies between each of the source terminal data syntaxes, and therefore the operation of presenting a single character cannot be considered to be the same in each terminal data

syntax. To factor out the commonality, the boundary condition of encountering the edge of the display area is identified separately from the presentation of a character. This aids conversion since it means that the boundary conditions applying to each presentation command are given explicitly. The conversion process is therefore independent of the internal boundary conditions within each of the source terminal data syntaxes.

Boundary-Condition-Definition ::= SET {

- [1] Screen-Dimensions,**
- [2] Colour-Map-Limit,**
- [3] Presentation-Sub-Area,**
- [4] Char-Mode-Constraints,**
- [5] Coordinate-Limit-Polygon,**
- [6] Coordinate-Limit-Spline,**
- [7] Presentation-Resolution,**
- [8] Macro-Seg-Memory-Limit,**
- [9] DRCS-Memory-Limit,**
- [10] Direct-Colours-Limit }**

A.2.1.3.1 Screen dimensions

```
Screen-Dimensions ::= SEQUENCE { INTEGER, INTEGER }
```

- *Screen-Dimensions* indicates the aspect ratio of the display screen expressed in terms of a fraction of the Y and a fraction of the X unit dimensions, where the INTEGER number represents a binary fraction with an implied binary point before the most significant bit. Note that a dimension of (1,1) implies no geometric constraint. A character mode service could use (1,1) to imply no constraint.

A.2.1.3.2 Colour map limit

Colour-Map-Limit ::= INTEGER

- The colour map limit indicates the maximum number of colours which may be stored in a single colour map, or the combined total for multiple colour maps, and represents the maximum number of colour states which may be encountered in a particular presentation page. In the case where no colour map is used, the integer specifies the number of fixed colours.

A.2.1.3.3 Presentation sub-area

Presentation-Sub-Area	::= SEQUENCE { Abs-Coord, Rel-Coord, INTEGER, INTEGER }
------------------------------	--

-- The two coordinates give the boundary dimensions of a sub-area of the display screen both in terms of the dimensions of the sub-area and the number of characters per row and the number of columns. The absolute coordinate specifies the origin of the sub-area, the relative coordinate the size of the sub-area and the INTEGER coordinates the limit on characters per row and rows respectively.

A.2.1.3.4 Char mode constraints

Char-Mode-Constraints ::= SEQUENCE { INTEGER, INTEGER }

-- The two parameters give the limit to the number of characters per row and the number of rows of text which may be presented on the display screen; that is, the boundaries at which character (or word) wrap and scroll will occur.

A.2.1.3.5 Coordinate limit polygon

Coordinate-Limit-Polygon ::= INTEGER

-- *The polygon coordinate limit specifies the maximum number of coordinates which may be specified for a filled polygon.*

A.2.1.3.6 Coordinate limit spline

Coordinate-Limit-Spline ::= INTEGER

- *The spline coordinate limit specifies the maximum number of coordinates which may be specified.*

A.2.1.3.7 Presentation resolution

Presentation-Resolution ::= SEQUENCE { INTEGER, INTEGER }

-- The presentation resolution specifies the nominal resolution of the display screen which was used by the information source.

A.2.1.3.8 Macro seg memory limit

Macro-Seg-Memory-Limit ::= INTEGER

-- The macro memory limit specifies the upper bound on the amount of memory which is available for the storage of Macros or Segments. The *INTEGER* parameter represents available memory expressed in bytes.

A.2.1.3.9 DRCS memory limit

DRCS-Memory-Limit ::= INTEGER

-- The DRCS memory limit specifies the upper bound on the amount of memory which is available for the storage of DRCS. The *INTEGER* parameter represents available memory expressed in bytes.

A.2.1.4 Data syntax identifier (SID)

**SID ::= IMPLICIT INTEGER { data-syntax-I (1),
data-syntax-II (2),
data-syntax-III (3) }**

-- *SID* is an identifier which is referenced in a number of primitive commands and which identifies the source data syntax of the command.

A.2.2 Reset state vector

Reset-State-Vector ::= SEQUENCE { SID, Vector-Definition }

-- The Reset State Vector command is used to establish the initial state for the Interworking Data Syntax. The default state may be selected from the table corresponding to the source terminal data syntax (or profile) given in Appendix II. Alternate parameters may be specified by use of explicit state vector and function and parameter definitions.

A.2.3 NULL

NULL implies that the state vector is unchanged from the previous presentation command.

A.3 Functions and parameters

The functions and parameters which make up the presentation commands are grouped into categories which depend upon their commonality between the various terminal data syntaxes. Those functions which are compatible, such as the basic repertoire of alphanumeric characters defined in Recommendation T.51, define separate groups. Those functions which are unique, such as certain specific special characters, also establish separate groups so that they may be converted or otherwise handled in a special manner. Functions such as DRCS and graphics drawing commands, which differ in fundamental ways between the various terminal data syntaxes, are organized so that those underlying capabilities which are common may be exploited in the necessary conversion process.

**Functions-&-Parameters ::= CHOICE { [0] Alpha-Char-String,
[1] Special-Char-String,
[2] Kana-Char-String,
[3] Kanji-Char-String,
[4] Block-Mosaic-String,
[5] Smooth-Mosaic-String,
[6] Special-Mosaic-String,
[7] Format-Effector-C0-Chars,
[8] Special-Format-C0-Characters,
[9] General-Control-Characters,
[10] Geometric-String,
[11] Animation-Control-String,
[12] Segment-Control-String,
[13] Colour-Control-String,
[14] Text-Control-String,
[15] Photo-Graphic-String-Syntetic-Image,
[16] Photo-Graphic-String-Natural-Image,
[17] MACRO-String,
[18] DRCS-String,
[19] Fill Pattern-Control-String,
[20] Music-String,
[21] Tele-Software-String,
[22] Audio-Data-String,
[23] Greek-Char-String }**

The first six categories of functions and the last one are various text or mosaic characters. None of the terminal data syntaxes defined in this Recommendation encompasses all of these characters. There are different unique characters in each of the terminal data syntaxes. However, a large portion of the repertoire is common between the different terminal data syntaxes, although the characters may be coded differently. Since coding is irrelevant here, and the use of particular tables could in fact cause serious confusion, characters extracted from the different character repertoires will be distinguished by the identifier name codes for each character as defined in Recommendation T.51. Since all of the terminal-oriented data syntaxes in this Recommendation do not explicitly make use of these name codes in the body of the Recommendation, the entire character repertoire, together with the name codes for each character are included here as an appendix.

A.3.0 Alpha char string

Alpha-Char-String ::= GRAPHICSTRING

- Characters (LA01 to LZ30, ND01 to ND09 and ND10, SC01 to SC05, SP01 to SP22, SA01 to SA07, NS01 to NS03, NF01 to NF21, SM01 to SM44 and SM47 to SM49, and SD11 to SD43) taken from Repertoire 1 which are the characters from the primary and supplementary character sets of Recommendation T.51 together with the SPACE character (SP01) and DELETE character (SM34).
- The coding of characters within an Alpha Character String will be taken from the IRV primary character code table (ISO Registration Number 2 under ISO 2375) and the secondary code table for use with IRV from ISO 6937/2 (ISO Registration Number 90).
- NOTE – The coding for the character \$ “Dollar Sign” (SC02) will be taken from the supplementary character set.
- NOTE – The coding for the character # “number sign” (SM01) will be taken from the primary character set.
- NOTE – The coding for the character “general currency sign” (SC01) will be taken from the primary character set.

A.3.1 Special char string

**Special-Char-String ::= INTEGER { non-spacing-vector-overbar (1),
non-spacing-slant (2),
left-vertical-bar-jointive (3),
right-vertical-bar-jointive (4) }**

- Non-Spacing-Vector-Overbar is a character (SM50) from Repertoire 2.
- Non-Spacing-Slant is a character (SM51) from Repertoire 2.
- Left-Vertical-Bar-Jointive is a character (SM45) from Repertoire 2.
- Right-Vertical-Bar-Jointive is a character (SM46) from Repertoire 2.

A.3.2 Kana char string

Kana-Char-String ::= GRAPHICSTRING

- Characters (JA01 to JA63) taken from Repertoire 3.
- The coding of characters within a Kana Character String will be taken from the Kana character code table (ISO Registration Number 56 under ISO 2375).

A.3.3 Kanji char string

Kanji-Char-String ::= GRAPHICSTRING

- Characters (JK01 to JK2980, HK01 to HK83, and JS01 to JS366) from Repertoire 4.
- The coding of characters within a Kanji Character String will be taken from the two byte Kanji character code table (ISO Registration Number 87 under ISO 2375).
- NOTE – The characters in this two byte code table which overlap other defined videotex character set are not considered to be part of Repertoire 4, and therefore are communicated as characters from Repertoire 1, Repertoire 3 or Repertoire 8 where appropriate. Specifically this involves the Latin alphanumeric characters (LA01 to LZ30), and non-alphabetic characters (ND01 to ND09 and ND10, SC01 to SC05, SP01, SP02, SP04 to SP15, SP17 to SP22, SA01 to SA07, NS02 to NS03, NF01 to NF05, SM01 to SM14, SM19, SM24 to SM34, SM38, SM43, SM44, SM47, SM48, and SD11 to SD43) from Repertoire 1, the Kana characters (JA01 to JA63) from Repertoire 3, the drawing characters (DG01 to DG04, DG13 to DG24, and DG32) from Repertoire 8, which have an alternate coding within the two byte code, but which are included in other Repertoires.

A.3.4 Block mosaic string

Block-Mosaic-String ::= GRAPHICSTRING

- Block Mosaic characters (MG01 to MG63) taken from Repertoire 7.
- The coding of characters for the Block Mosaic sub-Repertoire is identical between the three terminal data syntaxes defined in this Recommendation. The set is registered with ISO Number 129 under ISO 2375.

A.3.5 Smooth mosaic string

**Smooth-Mosaic-String ::= CHOICE { [1] Sub-Cell-Aligned-Smooth-Mosaics,
[2] General-Smooth-Mosaics }**

A.3.5.1 Sub-call aligned smooth mosaics

Sub-Cell-Aligned-Smooth-Mosaics ::= GRAPHICSTRING

- Smooth Mosaic characters (SG01 to SG56) taken from Repertoire 8.
- The coding of characters for the Sub Cell Aligned Smooth Mosaic sub-Repertoire is identical between the two terminal data syntaxes defined in this Recommendation which makes available these characters. These are registered as Numbers 71 and 72 under ISO 2375.

A.3.5.2 General smooth mosaics

General-Smooth-Mosaics ::= GRAPHICSTRING

- Smooth Mosaic characters (MS01 to MS28) taken from Repertoire 9.
- The coding of characters for the General Smooth Mosaic sub-Repertoire is taken from the terminal data syntax in this Recommendation which makes available these characters. This code table is registered under ISO 2375 as Registration Number 137.

A.3.6 Special mosaic string

**Special-Mosaic-String ::= CHOICE { [1] Drawing-Characters,
[2] Other-Special-Mosaics }**

A.3.6.1 Drawing characters

Drawing-Characters ::= GRAPHICSTRING

- Drawing characters (DG01 to DG50) taken from Repertoire 10.

A.3.6.2 Other special mosaics

**Other-Special-Mosaics ::= INTEGER { open-left-half-oval (1),
open-right-half-oval (2),
filled-left-half-oval (3),
filled-right-half-oval (4),
reverse-left-half-oval (5),
reverse-right-half-oval (6) }**

- Open-Left-Half-Oval is a Special Mosaic characters (MS13) from Repertoire 11.
- Open-Right-Half-Oval is a Special Mosaic characters (MS14) from Repertoire 11.
- Filled-Left-Half-Oval is a Special Mosaic characters (MS30) from Repertoire 11.
- Filled-Right-Half-Oval is a Special Mosaic characters (MS29) from Repertoire 11.
- Reverse-Left-Half-Oval is a Special Mosaic characters (MS15) from Repertoire 11.
- Reverse-Right-Half-Oval is a Special Mosaic characters (MS31) from Repertoire 11.

The function and parameter categories 7 and 8 contain basic control characters which are used to control the state of presentation of alphanumeric text and mosaic characters (including DRCS). These control characters can be broken down into two categories: format effector control characters and special format control characters. The format effector control characters have basically the same meaning in each of the three terminal data syntaxes. The only difference is how the functions invoked by these control characters interact with the terminal model and display environment of the various terminal data syntaxes; for example, they may apply to only one plane of display in a multi-plane terminal model or to all planes of display. The coding of the format effector characters is also compatible between the terminal data syntaxes.

The special format control characters in category 8, in general have a special meaning which is not shared by all of the data syntaxes. These functions must be specially converted during interworking, even between data syntaxes which appear to assign the same meaning to a particular control function. This is because the terminal model and display environment of the various terminal data syntaxes are quite different. The Bell character is included in this category because it requires special handling due to the timing of presentation. If a sorting of presentation commands is required in the interworking conversion process to accommodate differences in a terminal model, such as the handling of data intended for a multi-plane terminal on a single plane terminal, then the time of presentation of the Bell character must be altered.

A.3.7 Format effector C0-char

Format-Effector-C0-Char ::= GRAPHICSTRING

- *Format Effector C0 characters (APB, APF, APD, APU, CS, APR, APH) taken from CCITT Recommendation T.101 DS I, II, III; (C0 code table positions 0/8 to 0/13 and 1/14 respectively)*
- *APB – Active Position Back – Analogous to ISO 646 (FE₀ BS)*
- *APF – Active Position Forward – (FE₁ HT)*
- *APD – Active Position Down – (FE₂ LF)*
- *APU – Active Position Up – (FE₃ VT)*
- *CS – Clear Screen – (FE₄ FF)*
- *APR Active Position Return – (FE₅ CR)*
- *APH Active Position Home*

A.3.8 Special format-C0-char

Special-Format-C0-Char ::= CHOICE { [1] **Bell-Character,**
[2] **Position-Set,**
[3] **Cancel-Macro,**
[4] **Non-Selective-Reset,**
[5] **Cancel-Row }**

A.3.8.1 Bell character

Bell-Character ::= GRAPHICSTRING

- *Special C0 character (BEL) from Recommendation T.101 DS I, III (C0 set position 0/7).*
- *NOTE – This function provides an audio signal to the user of the terminal device. This function is not available in all of the terminal data syntaxes, and cannot be simulated in a reasonable manner.*

A.3.8.2 Position set

Position-set ::= SEQUENCE { INTEGER, INTEGER }

- *This function provides the equivalent capability of both the Active Position Set command (APS) from Recommendation T.101 DS I, III and the positioning portion of the Active Position Address command (APA) from DS II.*
- *The parameters to establish the new screen active position as a count of “current size” character cells from the “home” upper left position.*

A.3.8.3 Cancel macro

Cancel-Macro ::= GRAPHICSTRING

- *Special C0 character [CAN (Macro)] from Recommendation T.101 DS I, III (C0 set position 1/8).*

A.3.8.4 Non-selective reset

Non-Selective-Reset ::= SEQUENCE { [1] **NSR-Code,**
[2] **Position-Set OPTIONAL }**

NSR-Code ::= GRAPHICSTRING

- *Special C0 character (NSR) from Recommendation T.101 DS I, III (C0 set position 0/15). The positioning parameter sequence is optional.*

A.3.8.5 Cancel row

Cancel-Row ::= GRAPHICSTRING

- *Special C0 characters [CAN (Row)] from Recommendation T.101 DS II (C0 set position 1/8).*

A.3.9 General control characters

The function and parameter category 9 contains general control functions which are used to control the general state of presentation. The meaning of these control characters is very dependent upon the terminal model and display environment of the terminal data syntax in which they are used. Transcoding and conversion is required for each of the functions invoked by these control characters. These control characters have been organized into a number of sub-categories which correspond to the area of functionality being addressed.

General-Control-Characters ::= CHOICE {

- [1] Other-Format-Effectors,**
- [2] Lining-Control,**
- [3] Character-Size-Control,**
- [4] Flash-Control,**
- [5] Conceal-Control,**
- [6] Invert-Control,**
- [7] Window/Box-Control,**
- [8] Marking-Control,**
- [9] Protection-Control,**
- [10] Display-Control,**
- [11] Device-Control,**
- [12] Cursor-Control,**
- [13] Reset-Control }**

This subclause addresses the additional format effector characters which must be specially handled in conversion between the various data syntaxes.

A repeat function is available in all of the data syntaxes; however, the side effects of the function differ between the data syntaxes. Terminal data syntax DS I provides a function which allows the immediately preceding G-set character, or pair of characters in the case of a composite coded graphic character and non-spacing accent character, to be repeated. Both terminal data syntaxes DS II and DS III restrict the character to a graphic character (i.e. an alphanumeric text character or mosaic character from the repertoire, or a DRCS character). These limitations must be considered in establishing the conversion process. Here the repeat function will be considered to repeat any preceding G-set character and testing must be performed in the interpretation of the IDS to eliminate any erroneous cases.

The functions `hold mosaic` and `release mosaic` occur in only one data syntax and require special interpretation since analogous functions do not exist directly in any of the terminal data syntaxes.

A.3.9.1 Other format effectors

$$\text{Other-Format-Effectors} ::= \text{CHOICE} \{ \begin{array}{l} [1] \text{ Repeat-N,} \\ [2] \text{ Repeat-EOL,} \\ [3] \text{ Hold-Mosaic,} \\ [4] \text{ Release-Mosaic} \end{array} \}$$

A.3.9.1.1 Repeat-N

$$\text{Repeat-N} ::= \text{SEQUENCE} \{ \text{SID}, \text{RPT-Par} \}$$

-- Special character indicating the REPEAT Function from Recommendation T.101 DS I [C1 set position 5/8, (9/8)], DS II (C0 set position 1/2), DS III [C1 set position 4/6, (8/6)].

RPT-Par ::= INTEGER

- *Count of repetitions.*

A.3.9.1.2 Repeat EOL

Repeat-EOL ::= SID

-- Special character indicating the REPEAT TO End Of Line Function from Recommendation T.101 DS I [C1 set position 5/8, (9/8) with parameter 0], DS III [C1 set position 4/7, (8/7)].

A.3.9.1.3 Hold mosaic

Hold-Mosaic ::= SID

-- Special character indicating the Hold-Mosaic function (HMS) from Recommendation T.101 DS II [C1 set (serial position 5/14, (9/14)].

A.3.9.1.4 Release mosaic

Release-Mosaic ::= SID

-- Special character indicating the Release-Mosaic (RMS) function from Recommendation T.101 DS II [C1 set (serial position 5/15, (9/15)].

A.3.9.2 Lining control

The lining function permits an underline to be displayed as part of the graphics character shape for alphanumeric characters from Repertoire 1. This underline is considered as a part of the character cell image before any rotation operation is applied. In the special case of the display of mosaic characters, the lining function establishes a “separated mosaic” font. The capability to handle separated mosaics is available in all of the three terminal data syntaxes; however, the level of capability differs. In terminal data syntax DS II, only one size of separation for separated mosaics

is directly available. In terminal data syntaxes DS I and DS III the amount of separation is defined by the line width (drawing point size) parameter (logical pel) in the geometric drawing commands. Basic separated mosaics may be converted directly between each of the data syntaxes. Since the variation in separation cannot be achieved directly in one of the terminal data syntax it must be simulated by the use of DRCS. Of course simulation of separated mosaics in this manner would consume limited DRCS resources and therefore must consider the boundary condition specification.

**Lining-Control ::= INTEGER { start-lining (1),
stop-lining (2) }**

- *Start-Lining is a function from Recommendation T.101 DS I and II [C1 set position 5/10, (9/10)] and (UNDERLINE START) from DS III [C1 set position 5/9, (9/9)].*
- *Stop-Lining is a function from Recommendation T.101 DS I and II [C1 set position 5/9, (9/9)] and (UNDERLINE STOP) from DS III [C1 set position 5/10, (9/10)].*

A.3.9.3 Character size control

The various terminal data syntaxes provide the capability to establish a wide range of character sizes for basic alphanumeric text, mosaics and DRCS characters. In addition, terminal data syntax DS II provides the capability to separately define completely variable character sizes for text defined as part of the geometric part of DS II. Since this “geometric text” data is used only for the annotation of geometric pictures in the optional geometric part of DS II, it is not necessary to consider it as part of the translation of basic alphanumeric text. DS III, on the other hand, provides only one form of text. Therefore it is necessary to handle operations such as dynamic text sizes and rotations as part of the conversion between data syntaxes.

Since the capability to scale text, mosaics and DRCS to arbitrary sizes is not available in all data syntaxes, there will be some degradation of the displayed image when converted from one data syntax to another. It is undesirable to lose any textual information in the conversion process, since this textual information might be of principal importance to the understanding of the videotex page. Also it is not desirable to arbitrarily wrap or scroll textual information since this would corrupt mosaic information. In certain situations the conversion process must automatically choose a smaller size of character cell in order to avoid the loss of information. The commands for character size control indicate the size of the character cell intended in the terminal data syntax used to represent the source data. The resultant character cell in the converted form may be smaller depending upon the capabilities of the target terminal data syntax and the boundary condition in effect.

Two separate functions exist to define characters of double height. This is due to a difference in the definition of the relationship of the double height character cell to the location of the baseline in part of one of the source data syntaxes. Since data syntax DS II provides a capability to define double height characters which both extend up a double height above the baseline, and which extend down below the baseline, two functions are provided here. Since the other two terminal data syntaxes provide only a single double height capability, a conversion involving a repositioning of the baseline is required.

**Character-Size-Control ::= CHOICE { [1] Normal-Size,
[2] Double-Size-Up,
[3] Double-Width,
[4] Double-Height-Up,
[5] Double-Height-Down,
[6] Small-Size,
[7] Medium-Size,
[8] Double-Size-Down }**

A.3.9.3.1 Normal size

Normal-Size ::= SID

- *A function from Recommendation T.101 DS I [C1 set position 4/10, (8/10)], from DS II [C1 set position 4/12, (8/12)] and (NORMAL TEXT) from DS III [C1 set position 4/12, (8/12)].*
- *NOTE – The “Normal-Size” of text is defined by the boundary conditions of each of the terminal data syntaxes and is not the same in any of the terminal data syntaxes. Although the width of the “normal” character cell size is 1/40 of the screen width in DS II and DS III, the screen width is not exactly the same. The “normal” character cell size in DS I is by default 1/31 of the width; however, it may be redefined by the DS I P-TEXT command. Similarly the*

vertical height of a character cell differs between the various terminal data syntaxes. This command indicates that the source terminal data syntax intended to use “normal” size implicit in that terminal data syntax. This will require conversion to the normal size implicit in that resultant terminal data syntax. The value of “normal size” should be communicated explicitly in the state vector associated with this command.

A.3.9.3.2 Double size up

Double-Size-Up ::= SID

- A function from Recommendation T.101 DS II [C1 set position 4/15, (8/15)], (DOUBLE SIZE) from DS III [C1 set position 4/15, (8/15)], and (DBS 4/5) from DS I [C1 set position 4/11, (8/11) followed by parameter 4/5].
- NOTE – The character cell width and height are twice that defined by the control command “Normal-Size”.

A.3.9.3.3 Double width

Double-Width ::= SID

- A function from Recommendation T.101 DS II [C1 set position 4/14, (8/14)], and (DBW 4/4) from DS I [C1 set position 4/11, (8/11) followed by parameter 4/4].
- NOTE – The character cell width is twice that defined by the control command “Normal-Size”.

A.3.9.3.4 Double height up

Double-Height-Up ::= SID

- A function from Recommendation T.101 DS II [C1 set position 4/13, (8/13)], (DOUBLE HEIGHT) from DS III [C1 set position 4/13, (8/13)], and (DBH 4/1) from DS I [C1 set position 4/11, (8/11) followed by parameter 4/1].
- NOTE – The character cell height is twice that defined by the control command “Normal-Size” and extends up two character cell heights above the baseline.

A.3.9.3.5 Double height down

Double-Height-Down ::= SID

- A function from Recommendation T.101 DS II [C1 set position 4/13, (8/13)].
- NOTE – The character cell height is twice that defined by the control command “Normal-Size” and the double height character extends one cell height above the baseline and one cell height below the baseline.

A.3.9.3.6 Small size

Small-Size ::= SID

- A function from Recommendation T.101 DS I [C1 set position 4/8, (8/8)] and (SMALL TEXT) from DS III [C1 set position 4/10, (8/10)].
- NOTE – The character cell width and height are half that defined by the control command “Normal-Size”.

A.3.9.3.7 Medium size

Medium-Size ::= SID

- A function from Recommendation T.101 DS I [C1 set position 4/9, (8/9)] and (MEDIUM TEXT) from DS III [C1 set position 4/11, (8/11)].
- NOTE – The character cell size is defined to be an intermediate size. This intermediate size is defined by the boundary conditions of each of the source data syntaxes which use this control function. In data originating from data syntax DS III, medium size is defined to be 1/32 the normalized width of the display area and 3/64 the height of the normalized unit area. In data from data syntax DS I, medium text becomes half the character cell height and the full width defined by the control command “Normal-Size”.

A.3.9.3.8 Double size down

Double-Size-Down ::= SID

-- A function from Recommendation T.101 DS II [C1 set position 4/15, (8/15)].

A.3.9.4 Flash control

The operation of the flash capability is dependent on the terminal model of the particular source data syntax. In a “multi-plane” terminal configuration, character cells may have an implicit foreground and background which alternate during blinking. In a “single-plane” terminal configuration, the flash capability is achieved by use of colour mapping operations. It is possible to convert between these two variants on flashing. In addition to a basic flash capability driven by control characters, each of the terminal data syntaxes also provides the capability to establish complex dynamic important to reference the boundary condition imposed by the number of colours in the colour map and the terminal model plane structure.

Flash-Control ::= SEQUENCE { **Flash-Rate**, **Flash-Mode** }

Flash-Rate ::= CHOICE {[1] **Flash**,
[2] **Steady**,
[3] **Phase1-Flash**,
[4] **Phase2-Flash**,
[5] **Phase3-Flash**,
[6] **Increment-Flash**,
[7] **Decrement-Flash**,
[8] **Blink-Stop** }

Flash-Mode ::= CHOICE {[1] **Normal**,
[2] **Inverted-Flash**,
[3] **Reduced-Intensity-Flash** }

A.3.9.4.1 Flash

Flash ::= SID

- A function from Recommendation T.101 DS II [C1 set position 4/8, (8/8)], (FLC 4/0) from DS I [C1 set position 5/1, (9/1)] followed by parameter 4/0] and (BLINK START) from DS III [C1 set position 4/14, (8/14)].
- NOTE – Establish a 50% cycle flash either from the foreground to the background or between two colour map addresses chosen implicitly to produce the equivalent effect of foreground/background flashing. Although the Flash function is similar in the three source data syntaxes, the rate of flashing is not necessarily the same.

A.3.9.4.2 Steady

Steady ::= SID

- A function from Recommendation T.101 DS II [C1 set position 4/9, (8/9)], (FLC 4/15) from DS I [C1 set position 5/1, (9/1)] followed by parameter 4/15].
- NOTE – Cancel the application of any flashing attribute.

A.3.9.4.3 Inverted flash

Inverted-Flash ::= SID

- A function from Recommendation T.101 DS II (C1 set position CSI 3/0 4/1), (FLC 4/7) from DS I [C1 set position 5/1, (9/1)] followed by parameter 4/7].
- NOTE – Establish an inverted phase 50% cycle flash from the foreground to the background.

A.3.9.4.4 Reduced intensity flash

Reduced-Intensity-Flash ::= SID

- A function from Recommendation T.101 DS II (C1 set position CSI 3/1 4/1), (FLC 4/7) from DS I [C1 set position 5/1, (9/1)] followed by parameter 4/7].
- NOTE – Establish a reduced intensity flash between colour map addresses.

A.3.9.4.5 Phase 1-flash

Phase 1-Flash ::= SID

- A function from Recommendation T.101 DS II (C1 set position CSI 3/2 4/1), (FLC 4/4) from DS I [C1 set position 5/1, (9/1)] followed by parameter 4/4].
- NOTE – Establish a 33% cycle flash from the foreground to the background beginning on phase 1.

A.3.9.4.6 Phase 2-flash

Phase 2-Flash ::= SID

- A function from Recommendation T.101 DS II (C1 set position CSI 3/3 4/1), (FLC 4/2) from DS I [C1 set position 5/1, (9/1)] followed by parameter 4/2].
- NOTE – Establish a 33% cycle flash from the foreground to the background beginning on phase 2.

A.3.9.4.7 Phase 3-flash

Phase 3-Flash ::= SID

- A function from Recommendation T.101 DS II (C1 set position CSI 3/4 4/1), (FLC 4/1) from DS I [C1 set position 5/1, (9/1)] followed by parameter 4/1].
- NOTE – Establish a 33% cycle flash from the foreground to the background beginning on phase 3.

A.3.9.4.8 Increment flash

Increment-Flash ::= SID

- A function from Recommendation T.101 DS II (C1 set position CSI 3/5 4/1).
- NOTE – Establish a 33% cycle flash from the foreground to the background incrementing the phase reference.

A.3.9.4.9 Decrement flash

Decrement-Flash ::= SID

- A function from Recommendation T.101 DS II (C1 set position CSI 3/6 4/1).
- NOTE – Establish a 33% cycle flash from the foreground to the background incrementing the phase reference.

A.3.9.4.10 Blink stop

Blink-Stop ::= SID

- A function from Recommendation T.101 DS III [C1 set position 5/14, (9/14)].
- NOTE – Stop all blink processes.

A.3.9.5 Conceal control

The conceal display function is intended for operation on a terminal model which supports multiple independent planes. Data stored in character cells may be marked as concealed, in which case the background of the character cell will display in the same colour as the background of the cell. A local reveal command would cause the foreground to be displayed in the originally defined colours. A conversion is necessary to handle this function on a single plane terminal. The capability may be simulated either by the use of a key activated macro which contains a definition of the foreground of the concealed character cells or it may be simulated by the colour map. The definition of the key activated macro sequence must be established during a sorting process in the conversion procedure and is limited by the availability of macro memory. Use of the colour map for the simulation of this function consumes colour map resources very quickly. Therefore the handling of the conceal function should be the lowest priority in using colour map resources. The conceal and stop conceal control functions are included here so that they may be handled in the most effective manner by the conversion process.

Conceal-Control ::= CHOICE { [1] Conceal-Display,
[2] Stop-Conceal-Display }

A.3.9.5.1 Conceal display

Conceal-Display ::= SID

- A function from Recommendation T.101 DS II [C1 set position 5/8, (9/8)] and DS I [C1 set position 5/2, (9/2)] followed by parameter 4/0].
- NOTE – Establish a Conceal state attribute.

A.3.9.5.2 Stop conceal display

Stop-Conceal-Display ::= SID

- A function from Recommendation T.101 DS II (C1 set position CSI 4/2) and DS I [C1 set position 5/2, (9/2) followed by parameter 4/15].
- NOTE – Stop applying Conceal state attribute.

A.3.9.6 Invert control

**Invert-Control ::= CHOICE { [1] Invert-Polarity,
[2] Normal-Polarity }**

- NOTE – Invert the application of the foreground and background colour attributes in a multi-plane terminal model environment and invert the overlaying (foreground) and underlaying (background) colours in a single plane terminal environment. These commands have essentially the same effect when generating a presentation in each of the identified terminal model environments; however, there is a great difference in the effect when this command is used to change the attributes of an already displayed graphic character. This must be handled in the conversion by the process which converts the effects of different planes of the terminal model.

A.3.9.6.1 Invert polarity

Invert-Polarity ::= SID

- A function from Recommendation T.101 DS II [C1 set position 5/12, (9/12)] and (REVERSE VIDEO) DS III [C1 set position 4/8, (8/8)].
- NOTE – Establishes Invert Polarity attribute.

A.3.9.6.2 Normal polarity

Normal-Polarity ::= SID

- A function from Recommendation T.101 DS II [C1 set position 5/13, (9/13)] and (NORMAL VIDEO) DS III [C1 set position 4/9, (8/9)].
- NOTE – Establishes Normal Polarity attribute.

A.3.9.7 Window/box control

The window/box capability establishes a special background colour for a character cell which is transparent to a video image which may underlay the display. This capability is provided directly by two control commands in one of the source terminal data syntaxes. The same capability is provided in a more complex manner in all of the data syntaxes by the establishment of a special transparent colour which may be used together with other presentation commands.

**Window/Box-Control ::= INTEGER { start-box (1),
end-box (2) }**

- Start-Box is a function from Recommendation T.101 DS II [C1 set position 4/10, (8/10)].
- NOTE – Establish the Boxing attribute.
- End-Box is a function from Recommendation T.101 DS II [C1 set position 4/11, (8/11)].
- NOTE – Stop applying the Boxing attribute.

A.3.9.8 Marking control

The marking control capability marks character cell locations for further action. This function depends upon the availability of a character cell-oriented memory in the terminal model. It cannot be converted to other data syntaxes.

**Marking-Control ::= INTEGER { marked-mode-start (1),
marked-mode-stop (2) }**

- Marked-Mode-Start is a function from Recommendation T.101 DS II (C1 set position CSI 3/0 5/3, CSI 3/1 5/3 or CSI 3/2 5/3).
- NOTE – Apply the Marking attribute.
- Marked-Mode-Stop is a function from Recommendation T.101 DS II (C1 set position CSI 3/0 5/4, CSI 3/1 5/4 or CSI 3/2 5/4).
- NOTE – Stop applying Marking attribute.

A.3.9.9 Protection control

The manner in which selective input control is handled in the three source terminal data syntaxes differs greatly. Not only are the procedures different but the input processes are bounded by different boundary conditions. For example, in one case input is associated with the character cell memory of the multi-plane terminal model, whereas in another case, such input data is bounded by a storage limit on the number and cumulative size of such input fields. Since such input processes are fundamentally different, the commands which control them are included here separately. This will permit the conversion process to simulate one set of functions in a different terminal environment.

Protection-Control ::= INTEGER { **unprotect-field** (1),
 protect-field (2),
 protect-mode-start (3),
 protect-mode-cancel (4),
 protect-mode-idle (5),
 unprotect-block (6),
 protect-block (7) **}**

- *Unprotect-Field is a function from Recommendation T.101 DS III [C1 set position 5/15, (9/15)].*
- *NOTE – Unprotect a given area of the display screen, defined by the FIELD geometric command, to allow the input of characters into the unprotected field buffer when the cursor is in the unprotected area.*
- *Protect-Field is a function from Recommendation T.101 DS III [C1 set position 5/0, (9/0)].*
- *NOTE – Protect a given area of the display screen to prevent the input of characters into the unprotected field buffer when the cursor is in the unprotected area. The entire screen area is protected by default.*
- *Protect-Mode-Start is a function from Recommendation T.101 DS II (C1 set position CSI 3/0 5/0, CSI 3/1 5/0 or CSI 3/2 5/0).*
- *NOTE – Apply the protected attribute to character cell positions preventing overwriting.*
- *Protect-Mode-Cancel is a function from Recommendation T.101 DS II (C1 set position CSI 3/0 5/1, CSI 3/1 5/1 or CSI 3/2 5/1).*
- *NOTE – Cancel the protected attribute to character cell positions allowing overwriting.*
- *Protect-Mode-Idle is a function from Recommendation T.101 DS II (C1 set position CSI 3/2 5/2).*
- *NOTE – Stop the application of the protect mode attribute.*
- *Unprotect-Block is a function from Recommendation T.101 DS I [C1 set position 5/14, (9/14)].*
- *NOTE – Remove protection of character cell positions against alteration.*
- *Protect-Block is a function from Recommendation T.101 DS I [C1 set position 5/15, (9/15)].*
- *NOTE – Protect character cell positions against alteration.*

A.3.9.10 Display control

The display control subcategory of commands contains functions which affect the manner in which the display device presents information. This includes configuration of the display memory available in a particular terminal model, includes whether the contents of that display memory is to be scrolled, and includes the overwriting of information in the display memory.

Display-Control ::= CHOICE { **[1] Plane-Configuration-Control,**
 [2] Scroll-Control,
 [3] Overwrite-Mode }

The terminal models used in each of the three terminal data syntaxes differ significantly from one another. In two of the cases the terminal model structures are fixed. In the case of data syntax DS I the terminal model structure can be altered dynamically. The amount of display memory assigned to each display plane and the presentation (overlay) order of the planes may be altered. These functions are highly dependent upon the display hardware used to realize the particular display model which underlies data syntax DS I and the dynamic effects which may be generated using these functions cannot be converted to either of the other data syntaxes. However, these commands must be interpreted by the conversion process in order to establish the criteria for sorting other display information to achieve the mapping from the data syntax DS I multi plane terminal model to the different data syntax DS II multi-plane terminal model or to the data syntax DS III single plane terminal model.

A.3.9.10.1 Plane configuration control

Plane-Configuration-Control ::= CHOICE { [1] **Frame-Area,**
[2] **Set-Frame,**
[3] **Assign-Frame,**
[4] **Header-Area,**
[5] **Body-Area }**

A.3.9.10.1.1 Frame area

Frame-Area ::= SEQUENCE { Area-Origin, Area-Dimensions }

-- *The Frame Area Function is from Recommendation T.101 DS I [Display Control command set position 2/5, (10/5)].*

-- *NOTE – The Display Control Command G Set has final character 3/8 within DS I.*

Area-Origin ::= SEQUENCE { REAL, REAL }

-- *Specification of the origin of the Frame Area.*

Area-Dimensions ::= SEQUENCE { REAL, REAL }

-- *Specification of the dimensions of the Frame Area.*

-- *Coordinates are specified as normalized fractions of the unit screen area represented in a signed integer field with an implied binary point in the most significant place.*

A.3.9.10.1.2 Set frame

Set-Frame ::= SEQUENCE OF { Set-Frame-Index,
Set-Frame-Memory-Assignment }

-- *The Frame Area Function is from Recommendation T.101 DS I [Display Control command set position 2/6, (10/6)].*

Set-Frame-Index ::= INTEGER

-- *Frame Area Index.*

Set-Frame-Dimensions ::= INTEGER

-- *Number of bits of raster memory allocated to the frame.*

A.3.9.10.1.3 Assign frame

Assign-Frame ::= INTEGER

-- *A function from Recommendation T.101 DS I [Display Control command set position 2/7, (10/7)].*

A.3.9.10.1.4 Header area

Some of the terminal oriented Videotex data syntaxes provide a capability to present information in a special message area as well as in the main display area. This message area would contain service oriented messages. The content of these messages would doubtless change in international interworking between Videotex systems. Data syntax DS I provides special commands which control this message header. In DS I the raster and header raster commands control the display of presentation information in the main display area or the header message area. The raster commands also establish the initial colour values in data syntax DS I. These commands are included here so that header information can be identified and properly converted.

Header-Area ::= SEQUENCE { Raster-Colour-Value }

-- *A function from Recommendation T.101 DS I [Display Control command set position 3/9, (11/9)].*

-- *NOTE – The Display Control Command G Set has final character 3/8 within DS I.*

Raster-Colour-Values ::= SEQUENCE { INTEGER, INTEGER, INTEGER }

-- *Specification of the initial raster header colour for Green, Red, and Blue respectively.*

-- *Colour values are specified as normalized fractions of the unit range of colours represented in a signed integer field with an implied binary point in the most significant place.*

A.3.9.10.1.5 Body area

Body-Area ::= SEQUENCE { Body-Opcode, Raster-Colour-Values }

-- A function from Recommendation T.101 DS I [Display Control command set position 3/8, (11/8)].

A.3.9.10.2 Scroll control

Scrolling may occur on a whole screen basis or on a partial screen basis. There is a major difference between scrolling on a multi-plane terminal model and on a single plane terminal model. As well the assignment of functions to the various planes in a multi-plane terminal model also makes a tremendous difference to the result of scrolling. In some cases the underlying graphics information moves with the scrolling characters and in other cases it remains in place. In DS I the multi-plane motion capability permits dynamic motions and plane assignments which greatly affect how scrolling operates. In general it is not possible to convert all dynamic operations such as scrolling between terminal data syntaxes; however, the results of scrolling affects the final presentation. The conversion process must buffer data and post-process it so that the final image is correct. Since the scroll operations in each of the three terminal data syntaxes are fundamentally different, they are all included here so that the conversion process can handle them.

**Scroll-Control ::= CHOICE { scroll-on [1] NULL,
scroll-off [2] NULL,
scroll-up [3] NULL,
scroll-down [4] NULL,
activate-implicit-scrolling [5] NULL,
deactivate-implicit-scrolling [6] NULL,
create-scroll-area [7] Create-Scroll-Area,
delete-scroll-area [8] Delete-Scroll-Area,
scroll-display-mode-on [9] NULL,
scroll-display-mode-off [10] NULL }**

-- Scroll-On is a function from Recommendation T.101 DS III [C1 set position 5/7, (9/7)].

-- NOTE – Enable single plane scroll within an active display Field.

-- Scroll-Off is a function from Recommendation T.101 DS III [C1 set position 5/8, (9/8)].

-- NOTE – Disable single plane scroll.

-- Scroll-Up is a function from Recommendation T.101 DS II (C1 set position CSI 3/0 6/0).

-- NOTE – Cause the scrolling area to scroll up.

-- Scroll-Down is a function from Recommendation T.101 DS II (C1 set position CSI 3/1 6/0).

-- NOTE – Cause the scrolling area to scroll down.

-- Activate-Implicit-Scrolling is a function from Recommendation T.101 DS II (C1 set position CSI 3/2 6/0).

-- NOTE – Cause the scrolling area to scroll implicitly on encountering scroll area boundary.

-- Deactivate-Implicit-Scrolling is a function from Recommendation T.101 DS II (C1 set position CSI 3/3 6/0).

-- NOTE – Cause the scrolling area not to scroll implicitly.

-- Scroll-Display-Mode-On is a function from Recommendation T.101 DS I (Display Control Command G Set position 2/4 with parameter $b_6 = 1$).

-- NOTE – The Display Control Command G Set has final character 3/8 within DS I.

-- NOTE – Establish the Scroll attribute of the Display Mode.

-- Scroll-Display-Mode-Off is a function from Recommendation T.101 DS I (Display Control Command G Set position 2/4 with parameter $b_6 = 0$).

-- NOTE – Disable the Scroll attribute of the Display Mode.

A.3.9.10.2.1 Create scroll area

Create-Scroll-Area ::= SEQUENCE { Upper-Par, Lower-Par }

-- A function from Recommendation T.101 DS II (5/5).

-- NOTE – Create a scrolling area.

Upper-Par ::= SEQUENCE { INTEGER, INTEGER, INTEGER }

-- Parameters <URH> <URT> <URU> defining the upper boundary row of the scrolling area.

Lower-Par ::= SEQUENCE { INTEGER, INTEGER, INTEGER }

-- Parameters <LRH> <LRT> <LRU> defining the lower boundary row of the scrolling area.

A.3.9.10.2.2 Delete scroll area

Delete-Scroll-Area ::= SEQUENCE { Upper-Par, Lower-Par }

-- A function from Recommendation T.101 DS II (5/6).

-- NOTE – Delete a scrolling area.

A.3.9.10.3 Overwrite mode

In conjunction with control over the terminal model memory configuration, one of the terminal data syntaxes provides a unique capability of controlling how data builds up in a particular display plane. Data syntax DS I allows the overwriting of memory to be dependent upon the current contents of memory. The new data may either replace the old contents of memory, or perform a logical “OR”, logical “AND”, or logical “XOR” (eXclusive OR) with the old contents of the memory before replacing it. This function is extremely difficult to simulate in either of the other two data syntaxes in the general case since it requires operations at the bit level within a particular terminal model dependent memory. It is included here so that the conversion process can perform the best simulation possible.

Overwrite-Mode ::= SEQUENCE { Overwrite-Par }

-- A function from Recommendation T.101 DS I (Display Control Command G Set position 2/4).

-- NOTE – The Display Control Command G Set has final character 3/8 within DS I.

Overwrite-Par ::= INTEGER { replace (1),
or (2),
and (3),
xor (4) }

A.3.9.11 Device control

Except for the display device on or off commands, the device-control commands control other than presentation display functions and are outside the scope of the interworking data syntax.

Device-Control ::= INTEGER { display-device-on (1),
display-device-off (2) }

-- Display-Device-On is a function from Recommendation T.101 DS II (Control Sequence ESC 3/12).

-- Display-Device-Off is a function from Recommendation T.101 DS II (Control Sequence ESC 3/13).

A.3.9.12 Cursor control

The display cursor is controlled explicitly in each of the terminal data syntaxes as well as implicitly in one. In addition the explicit cursor control commands do not have the same coding in any of the terminal data syntaxes. In the implicit case of cursor control, the display cursor is controlled by the unprotected field protect mode control in terminal data syntax DS III. Conversion is required between each of these control functions.

Cursor-Control ::= CHOICE { Cursor-On (1),
Cursor-Flash (2),
Cursor-Off (3) }

A.3.9.12.1 Cursor-on

Cursor-On ::= SID

-- A function from Recommendation T.101 DS II (C0 set position 1/1), DS I [C1 set position 4/14, (8/14)] and (CURSOR STEADY) from DS III [C1 set position 5/12, (9/12)].

A.3.9.12.2 Cursor flash

Cursor-Flash ::= SID

-- A function from Recommendation T.101 DS III [C1 set position 5/11, (9/11)].

A.3.9.12.3 Cursor-off

Cursor-Off ::= SID

-- A function from Recommendation T.101 DS II (C0 set position 1/14), DS I [C1 set position 4/15, (8/15)] and (CURSOR OFF) from DS III [C1 set position 5/13, (9/13)].

A.3.9.13 Reset control

Each of the source Videotex data syntaxes provides the capability to reset the states of the display environment supporting that particular data syntax to a predefined set of values. The parameters which may be altered by the various reset functions in the different source data syntaxes are quite different. Some of the reset functions provide the capability to reset particular parameters selectively while others reset a data syntax dependent predefined list of parameters. The interworking data syntax must support reset functions in two different ways. Firstly, an indication of the particular reset command must be communicated as a syntactic element within the IDS. The various reset functions are included here so that the presentation affect of the reset function may be effected in the conversion. Secondly, a reset function greatly effects the global presentation states. These states are kept track of in the conversion process so that the conversion process need not understand the interrelationship between presentation commands. This means that the conversion process does not have to simulate a terminal of the source data syntax in order to handle the conversion of its elements. Therefore, along with an IDS reset control command, it is necessary to include a special form of the state vector which re-establishes the global variables.

Reset-Control ::= CHOICE { [1] Reset-Type-I,
[2] Reset-Type-II,
[3] Reset-Type-III }

A.3.9.13.1 Reset type-I

Reset-Type-I ::= SEQUENCE { P-Reset-Par OPTIONAL }

-- A function from Recommendation T.101 DS I [Display Control Command G Set position 2/1, (10/1)].

-- NOTE – The Display Control Command G Set has final character 3/8 within DS I.

A.3.9.13.1.1 P-reset par

P-Reset-Par ::= SEQUENCE { macro-reset BOOLEAN,
blink-reset BOOLEAN,
lut-reset BOOLEAN,
screen-reset BOOLEAN }

-- Selectively reset the identified parameters.

-- NOTE – Data Syntax DS I also includes the NSR reset function which is identified separately above.

A.3.9.13.2 Reset type-II

Reset-Type-II ::= SEQUENCE { US-Reset-Operation,
US-Reset-Parameter }

-- A function from Recommendation T.101 DS II (C0 set position 1/15) followed by fixed character 2/15.

A.3.9.13.2.1 US-reset operation

US-Reset-Operation ::= CHOICE { us-reset-mosaic-1 [1] NULL,
us-reset-mosaic-2 [2] NULL,
us-reset-mosaic-1-limited [3] NULL,
us-reset-mosaic-2-limited [4] NULL,
us-reset-service-break [5] US-Reset-Service-Break,
us-reset-to-previous-state [6] NULL }

-- US-Reset-Mosaic-1 is represented by US Reset Identifier Character (4/1), and resets to defaults and invokes serial C1 set.

-- US-Reset-Mosaic-2 is represented by US Reset Identifier Character (4/2), and resets to defaults and invokes parallel C1 set.

- *US-Reset-Mosaic-1-Limited* is represented by *US Reset Identifier Character (4/3)*, and resets to limited defaults and invokes parallel C1 set.
- *US-Reset-Mosaic-2-Limited* is represented by *US Reset Identifier Character (4/4)*, and resets to limited defaults and invokes parallel C1 set.
- *US-Reset-to-Previous-State* is represented by *US Reset Identifier Character (4/15)*, and resets to previous state after a reset to service break.

A.3.9.13.2.2 US-reset service break

**US-Reset-Service-Break ::= SEQUENCE { INTEGER { break-to-row-serial (1),
break-to-row-parallel (2) }, row-designator }**

- *Break-to-Row-Serial* is represented by *US Reset Identifier Character (4/0)*, and service breaks to row serial C1 set.
- *Break-to-Row-Parallel* is represented by *US Reset Identifier Character (4/5)*, and service breaks to row parallel C1 set.
- *Row-Designator* is represented by *US Reset Row Designator Parameter Character*, where 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.

A.3.9.13.3 Reset type-III

**Reset-Type-III ::= SEQUENCE { [1] Reset-Par1 OPTIONAL,
[2] Reset-Par2 OPTIONAL }**

- *A function from Recommendation T.101 DS III [PD1 G Set position 2/0, (10/0)].*

**Reset-Par1 ::= SEQUENCE { INTEGER {
colour-mode-1 (1),
colour-mode-2 (2),
colour-mode-3 (3), }
INTEGER {
display-to-nominal-black (1),
display-to-current-colour (2),
border-to-nominal-black (3),
border-to-current-colour (4),
display-and-border-to-current-colour (5),
display-to-current-colour-and-border-to-nominal-black (6),
display-and-border-to-nominal-black (7) },
domain BOOLEAN }**

**Reset-Par2 ::= SEQUENCE {
drcs-reset BOOLEAN,
macro-pdi-reset BOOLEAN,
texture-reset BOOLEAN,
unprotected-field-reset BOOLEAN,
blink-pdi-reset BOOLEAN,
text-pdi-reset BOOLEAN }**

- *Selectively reset the identified parameters.*
- *NOTE – Data Syntax DS III also includes the NSR reset function which is identified separately above.*

A.3.10 Geometric string

All of the source terminal data syntaxes provide a geometric capability, however the capabilities available in each of the geometric systems are quite different. The interworking data syntax groups the common geometric commands together. Recommendation F.300 has identified the various categories into which geometric functions may be organized. These categories are used below. The IDS uses normalized coordinates for all geometric commands. Also the IDS uses relative coordinate specifications for all lists of coordinates, except for the set-position and marker-point commands which are absolute. However, in certain cases there may be a choice of either absolute or relative coordinates. All other forms of coordinates used within any of the terminal oriented data syntaxes, such as the general use of absolute or incremental, will be converted to the forms indicated above.

**Geometric-String ::= CHOICE { [1] Geometric-Drawing-Command,
[2] Geometric-Control-Command }**

A.3.10.1 Geometric drawing command

**Geometric-Drawing-Command ::= CHOICE { [1] Marker-Point,
[2] Line,
[3] Arc-Circle,
[4] Rectangle,
[5] Polygon,
[6] Spline,
[7] Pixel-Array }**

Some of the source terminal data syntaxes provide a method of optionally carrying on from one primitive to another in a relative manner. This provides a level of efficiency in certain situations, however the multiplicity of equivalent formats would make the interworking data syntax more complex. Therefore, the interworking data syntax requires the specification of the initial position of a drawing command as part of the string of parameters for each command. In some situations in converting from data syntaxes which permit the relative association of commands, it will be necessary for the conversion process to calculate the current position in effect at the beginning of a command and include that data as part of the parameter string. There is no direct equivalent in the IDS of the data syntaxes I and III set position command. This information is carried as the initial parameter of each of the other drawing commands.

A.3.10.1.1 Marker point

The various terminal data syntaxes differ in their capability to present a marker shape at a point. Data syntaxes DS I and DS III provide only the capability to draw a dot, whereas data syntax DS II also provides the capability to draw a marker shape at a specific point. The conversion process can easily simulate the marker point functionality in converting to data syntax DS I or DS III by the use of more than one presentation function, possibly included in a MACRO command for efficiency. The dot-point or shape-point command is identified by the context tag in the CHOICE statement. The shape of the shape point (marker) is defined by a geometric control command.

**Marker-Point ::= CHOICE { [1] Dot-Point,
[2] Shape-Point }**

Dot-Point ::= SEQUENCE OF { Abs-Coord }

-- *This command carries the functionality of the Data Syntax I, and III SET POINT command and of the Data Syntax II POLYMARKER command, with the marker in the shape of a dot.*

Shape-Point ::= SEQUENCE OF { Abs-Coord }

-- *This command carries the functionality of the Data Syntax II POLYMARKER command with a general marker shape.*

A.3.10.1.2 Line

All of the terminal data syntaxes provide the capability to draw a single or a series of lines. Minor differences exist with respect to the manner in which boundary conditions are handled, however in general a direct conversion is possible.

Line ::= SEQUENCE OF { Abs-Coord, SEQUENCE OF { Rel-Coord } }

-- *This command carries the functionality of the Data Syntax I, and III LINE command and of the Data Syntax II POLYLINE command.*

A.3.10.1.3 Arc-circle

The capability to draw an arc or a circle differs somewhat between the various data syntaxes. In each of the various data syntaxes the circle/arc function has been optimized to such an extent that it provides an efficient manner of communicating arc or circle information in the context of that data syntax. The interworking data syntax is less concerned about efficiency than it is about carrying sufficient information to permit conversion to take place. Therefore, alternate ways of carrying the same parameters will not be addressed by the IDS, however, the IDS will include all the functions available in the circle-arc capability in the various data syntaxes.

**Arc-Circle ::= CHOICE { [1] Circle,
[2] Arc-3-Point,
[3] Arc-3-Point-Chord,
[4] Arc-3-Point-Pie,
[5] Ellipse,**

- [6] Elliptic-Arc,
- [7] Elliptic-Arc-Chord,
- [8] Elliptic-Arc-Pie,
- [9] Arc-Centre-Cord,
- [10] Arc-Centre-Pie }

A.3.10.1.3.1 Circle

Circle ::= SEQUENCE { Abs-Coord, Coord }

- This command carries the functionality of the Data Syntax I, and III ARC command (circle form) and of the Data Syntax II GDP (circle) command.
- The absolute coordinate defines the initial position of the circle. The other coordinate defines the diameter of the circle by specifying a point on the opposite side.

A.3.10.1.3.2 Arc-3 point

Arc-3-Point ::= SEQUENCE { Abs-Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax I, and III ARC command (outline form) and of the Data Syntax II GDP (circular arc 3 point) command.
- The absolute coordinate defines the initial position of the arc. The two other coordinate parameters define a point on the arc and the final position of the arc respectively.

A.3.10.1.3.3 Arc-3 point chord

Arc-3-Point-Chord ::= SEQUENCE { Abs-Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax I, and III ARC command (chord fill form) and of the Data Syntax II GDP (circular arc 3 point chord) command.
- The absolute coordinate defines the initial position of the arc. The two other coordinate parameters define a point on the arc and the final position of the arc respectively. A Chord is drawn from the initial to the final position of the arc.

A.3.10.1.3.4 Arc-3 point pie

Arc-3-Point-Pie ::= SEQUENCE { Abs-Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax II GDP (circular arc 3 point pie) command.
- The absolute coordinate defines the initial position of the arc. The two other coordinate parameters define a point on the arc and the final position of the arc respectively. Two lines are drawn from the initial to the geometric centre of the arc and then to the final position of the arc to form a pie shape. Although a pie filled arc is not directly available in data syntax DS I or DS III the conversion process can simulate the function by the use of an arc and two lines.

A.3.10.1.3.5 Ellipse

Ellipse ::= SEQUENCE { Abs-Coord, Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax II GDP (ellipse) command.
- The absolute coordinate defines the initial position of the ellipse. A second coordinate parameters defines a point on the opposite side of the arc which establishes the major axis diameter. The third and fourth parameters define the minor axis diameter. Although an ellipse or elliptic arc are not directly available in data syntax DS I or DS III the conversion process can simulate the function in a piecewise manner or by fitting a spline curve.

A.3.10.1.3.6 Elliptic arc

Elliptic-Arc ::= SEQUENCE { Abs-Coord, Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax II GDP (elliptic arc) command.
- The absolute coordinate defines the initial position of the arc. A second coordinate parameters defines a point on the opposite side of the arc which establishes the major axis diameter. A third parameter defines the minor axis diameter. A fourth parameter defines the final position of the arc.

A.3.10.1.3.7 Elliptic arc chord

Elliptic-Arc-Chord ::= SEQUENCE { Abs-Coord, Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax II GDP (elliptic arc chord) command.

- The absolute coordinate defines the initial position of the arc. A second coordinate parameters defines a point on the opposite side of the arc which establishes the major axis diameter. A third parameter defines the minor axis diameter. A fourth parameter defines the final position of the arc. A Chord is drawn from the initial to the final position of the arc.

A.3.10.1.3.8 Elliptic arc pie

Elliptic-Arc-Pie ::= SEQUENCE { Abs-Coord, Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax II GDP (elliptic arc pie) command.
- The absolute coordinate defines the initial position of the arc. A second coordinate parameters defines a point on the opposite side of the arc which establishes the major axis diameter. A third parameter defines the minor axis diameter. A fourth parameter defines the final position of the arc. Two lines are drawn from the initial to the geometric centre of the arc and then to the final position of the arc to form a pie shape.

A.3.10.1.3.9 Arc centre cord

Arc-Centre-Chord ::= SEQUENCE { Abs-Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax II GDP (arc-centre-chord) command.
- The absolute coordinate defines the initial position of the arc. The other coordinate parameters define the start and end points of the arc.

A.3.10.1.3.10 Arc centre pie

Arc-Centre-Pie ::= SEQUENCE { Abs-Coord, Coord, Coord }

- This command carries the functionality of the Data Syntax II GDP (arc-centre-pie) command.
- The absolute coordinate defines the centre of the arc. The other coordinate parameters define the start and end points of the arc.

A.3.10.1.4 Rectangle

Rectangle ::= SEQUENCE { Abs-Coord, Rel-Coord }

- This command carries the functionality of the Data Syntax I and III RECTANGLE command and the Data Syntax II GDP (rectangle) command.
- The absolute coordinate defines the initial position of the rectangle. A relative coordinate parameters defines a point on the diagonally opposite side of the rectangle which establishes the size of the rectangle.

A.3.10.1.5 Polygon

Polygon ::= SEQUENCE { Abs-Coord, SEQUENCE OF { Rel-Coord } }

- This command carries the functionality of the Data Syntax I and III POLYGON (filled) command and the Data Syntax II FILL AREA command. Data Syntax I and III also provide a POLYGON (outline) command which can be carried through the IDS by a LINE command with a repetition of the initial points as the final point.
- The absolute coordinate defines the initial position of the polygon. The sequence of relative coordinate defines the vertices of the polygon. A polygon is always closed and the final position is the same as the initial position.

A.3.10.1.6 Spline

Spline ::= SEQUENCE { Abs-Coord, SEQUENCE OF { Rel-Coord } }

- This command carries the functionality of the Data Syntax I and III ARC (spline) command and the Data Syntax II GDP (spline) command.
- The absolute coordinate defines the initial position of the poly curve. The sequence of relative coordinates (greater than 3) defines the curve.
- NOTE – The various terminal data syntaxes do not use exactly the same definition of the type and/or parameters for the spline generating function, however all of the source terminal data syntaxes tend to use a spline function of some type. Although potentially this could cause significant differences in the resultant picture after conversion, it is still the closest result than can be generated in a reasonable manner.

A.3.10.1.7 Pixel array

Pixel-Array ::= SEQUENCE {
 first-point Abs-Coord,
 second-point Abs-Coord,
 third-point Rel-Coord,

- These 3 points define the pixel area which in general could be a parallelogram. The first two points are the end points of a diagonal.

cells-first-direction INTEGER,
cells-second-direction INTEGER,

- *These values divide the pixel area in a grid with equal dimensions, to represent the intended (logical) resolution. The first direction is considered from the first to the third point. The second direction is from the first point to the unspecified point. These values can easily be derived, e.g. from the logical pel in case of INCREMENTAL POINT.*

Pixel-Array-Data }

Pixel-Array-Data ::= CHOICE { [1] IMPLICIT SEQUENCE OF Basic-Colour-Selection,
[2] IMPLICIT SEQUENCE OF Direct-Colour-Selection,
[3] IMPLICIT SEQUENCE OF Indexed-Colour-Selection }

- *The colour list is defined according to the 'Colour-Control-String'. Auxiliary colour selection is not meaningful for this definition. The first colour is mapped to the cell associated with the first point. The colour elements are mapped within rows running from the first to the third point, and with rows incrementing in order from the third to the second point.*

The various source terminal Videotex data syntaxes contain commands to efficiently code line and polygon data in an incremental fashion to achieve greater efficiency. The incremental capability differs greatly between the different data syntaxes, and no intermediate format could be developed which would be suitable in all the different environments. Since efficiency is of secondary importance, incremental lines and polygons should be communicated in terms of the general line and polygon functions above.

A.3.10.2 Geometric control commands

A large number of control commands are available in each of the terminal data syntaxes to control the geometric drawing functions. Although many of the geometric control commands defined in each of the data syntaxes may appear to be the same, they differ in side effect. For this reason all of the geometric control commands which appear in the various data syntaxes are included here. Only where the control commands are identical, such as a number of the geometric control commands in data syntaxes DS I and DS III, is a common control command definition used below.

Geometric-Control-Command ::= CHOICE { [1] Geo-Control-Command-1,
[2] Geo-Control-Command-2 }

- *Two types of geometric control commands are included in the IDS in order to accommodate the two different approaches taken in Data Syntax II and in Data Syntax I, III. These commands are grouped separately since they would never be received in combination.*

A.3.10.2.1 Geo control command-1

Geometric-Control-Command-1 ::= CHOICE { [1] Numeric-Precision,
[2] Drawing-Point-Size,
[3] Line-Style,
[4] Highlight,
[5] Fill,
[6] Field,
[7] Blink-Process,
[8] Wait }

- *Geometric control commands analogous to those in Data Syntax I and III.*

A.3.10.2.1.1 Numeric precision

Numeric-Precision ::= SEQUENCE { REAL, REAL }

- *This command carries the functionality of the Data Syntax I, and III DOMAIN geometric control command.*
- *Define the nominal numeric precision in use by the source data syntax. Since the ASN.1 encoding rules permit any precision of data to be communicated, this control command does not affect the precision of data communicated. It is used to inform the conversion process of the nominal precision being used by the source data syntax. The first parameter carries the precision, expressed as a number of significant bits, for single-value operands. Similarly the second parameter carries the number of significant bits for multi-valued (2nd and 3rd) operands.*

A.3.10.2.1.2 Drawing point size

Drawing-Point-Size ::= Rel-Coord

- This command carries the functionality of the Data Syntax I, and III DOMAIN (logical pel size) geometric control command.
- This geometric control function establishes the size of the logical drawing point (LOGICAL PEL) as a fraction of the unit screen dimensions. The special case of zero is interpreted as being the smallest size possible on a given presentation device.

A.3.10.2.1.3 Line style

**Line-Style ::= INTEGER { solid (1),
dotted (2),
dashed (3),
dot-dashed (4) }**

- Establish the style for presenting lines from a fixed set of line styles.
- This command carries the functionality of the Data Syntax I, and III TEXTURE (line texture) geometric control command.

A.3.10.2.1.4 Highlight

Highlight ::= BOOLEAN

- Establish whether filled areas are drawn in highlight mode, in which the perimeter is drawn in BLACK or a contrasting colour to the fill.
- This command carries the functionality of the Data Syntax I, and III TEXTURE (highlight) geometric control command.

A.3.10.2.1.5 Fill

Fill ::= BOOLEAN

- Establish whether polygons, closed arcs, ellipses or rectangles are to be filled. For efficiency this control is coded as part of the opcode identifying the drawing primitive in some of the source terminal data syntaxes. This function has been separated here in order to ease conversion between data syntaxes.
- This command carries the functionality of the Data Syntax I, and III TEXTURE (fill texture pattern) geometric control command.

A.3.10.2.1.6 Field

Field ::= Rel-Coord

- Define the dimensions of the active area on the display screen. The field command establishes boundaries which "contain" text; that is boundaries for scroll areas, and to which the format effector characters operate. The initial position is defined by the current geometric drawing position. The relative coordinate parameters define a point on the diagonally opposite side of the field which establishes the size of the field rectangular area.
- This command carries the functionality of the Data Syntax I, and III FIELD geometric control command.

A.3.10.2.1.7 Blink process

**Blink-Process ::= SEQUENCE { [1] INTEGER,
[2] INTEGER OPTIONAL,
[3] INTEGER OPTIONAL,
[4] INTEGER OPTIONAL }**

- Establish a Blink process in which the colour map is dynamically altered for a specified interval and phase. The first integer represents the colour map address of the Blink To colour, then the On interval, the Off interval and the Phase Delay in 1/10 of a second respectively. The capability to handle Blink processes is very terminal model dependent. In general Blink processes can be used to simulate any other blink capability available in any data syntax, within the limits of the available memory assigned for such operations, as specified in boundary value conditions. However Blink processes cannot easily be simulated in display environments which do not present sufficient capabilities.
- This command carries the functionality of the Data Syntax I, and III BLINK geometric control command.

A.3.10.2.1.8 Wait

Wait ::= INTEGER

- Establish a time delay in processing presentation data for the time specified in units of 1/10 of a second. Although the Wait command is very simple, it provides very great problems in conversion. This is because the wait command is a dynamic control command. Presentation dynamics cannot be guaranteed in conversion because the order of presentation commands may have to be altered to accommodate for differences in the terminal model between two data syntaxes. Conversion of the wait command should only be attempted when the source and target presentation processes are in synchronization, i.e. when no sorting of presentation commands is necessary in the conversion, or at the end of a unit (page) of data.

A.3.10.2.2 Geo control command-2

Geo-Control-Command-2 ::= CHOICE { [1] Display-Element-Attributes,
[2] Control-Element-Attributes }

- Geometric control commands analogous to those in Data Syntax II.
- Display Element Attributes pertain to the output display primitives. Some of this primitives may be similar to those in Geo-Control-Command-1 section, however the side effects are different for these commands.
- Control Element Attributes establish the display transformation, clipping and work station control functions which are unique to the display environment associated with Data Syntax II.
- The use of bundle facilities is for further study.

A.3.10.2.2.1 Display element attributes

Display-Element-Attributes ::= CHOICE {
[1] IMPLICIT Line-Attributes,
[2] IMPLICIT Marker-Attributes,
[3] IMPLICIT Fill-Area-Attributes }

Line-Attributes ::= SET {
[1] IMPLICIT Line-Type OPTIONAL,
[2] IMPLICIT Line-Width-Scale-Factor OPTIONAL,
[3] IMPLICIT Polyline-Colour-Index OPTIONAL }

Line-Type ::= INTEGER {
solid (0),
dashed (1),
dotted (2),
dashed-dotted (3),
implementation dependent (4) }

Line-Width-Scale-Factor ::= REAL

Polyline-Colour-Index ::= Colour-Index

Marker-Attributes ::= SET {
[1] IMPLICIT Marker-Type OPTIONAL,
[2] IMPLICIT Marker-Size-Scale-Factor OPTIONAL,
[3] IMPLICIT Polymarker-Colour-Index OPTIONAL }

Marker-Type ::= INTEGER {
dot (0),
plus (1),
asterisk (2),
circle (3),
diagonal-cross (4) }

Marker-Size-Scale-Factor ::= REAL

Polymarker-Colour-Index ::= Colour-Index

Fill-Area-Attributes ::= SET {
[1] IMPLICIT Fill-Area-Interior-Style OPTIONAL,
[2] IMPLICIT Fill-Area-Colour-Style OPTIONAL,
[3] IMPLICIT Fill-Area-Style-Index OPTIONAL,
[4] IMPLICIT Pattern-Reference-Point OPTIONAL,
[5] IMPLICIT Pattern-Vectors OPTIONAL }

Fill-Area-Interior-Style	::= INTEGER {	hollow	(0),
		solid	(1),
		pattern	(2),
		hatch	(3) }
Fill-Area-Colour-Index	::= Colour-Index		
Fill-Area-Style-Index	::= INTEGER {		
<i>-- For interior style pattern the fill area style index selects a pattern defined by “Fill pattern control string”.</i>			
<i>-- For interior style hatch the following styles are selected:</i>			
		vertical-lines	(0),
		horizontal-lines	(1),
		slope-45-degree-lines	(2),
		slope-45-degree-lines	(3),
		crossed-lines-vertical-and-horizontal-lines	(4),
		crossed-lines-45-and-45-degrees	(5) }
Pattern-Reference-Point	::= Abs-Coord		
Pattern-Vectors	::= SEQUENCE { Abs-Coord, Abs-Coord }		
<i>-- The origin of the NDC space and the first point define the pattern height vector. The origin of the NDC space and the second point define the pattern width vector.</i>			
Colour-Index	::= CHOICE {		
		[1] IMPLICIT Basic-Colour-Selection,	
		[2] IMPLICIT Indexed-Colour-Selection }	
A.3.10.2.2.2 Control element attributes			
Control-Element-Attributes	::= CHOICE {		
		[1] WS-Management-Primitives,	
		[2] Transformation-Primitives }	
WS-Management-Primitives	::= CHOICE {		
open-workstation		[1] IMPLICIT INTEGER,	
		<i>-- WS Identifier</i>	
close-workstation		[2] IMPLICIT INTEGER,	
		<i>-- WS Identifier</i>	
activate-workstation		[3] IMPLICIT INTEGER,	
		<i>-- WS Identifier</i>	
deactivate-workstation		[4] IMPLICIT INTEGER,	
		<i>-- WS Identifier</i>	
clear-workstation		[5] IMPLICIT INTEGER,	
		<i>-- WS Identifier</i>	
set-defaults		[6] IMPLICIT NULL,	
update-workstation		[7] IMPLICIT Update-WS,	
deferral-state		[8] IMPLICIT Deferral-State }	
Update-WS	::= SEQUENCE {		
workstation-identifier		INTEGER,	
regeneration-flag		INTEGER {perform	(0),
		postpone	(1) } }
Deferral-State	::= SEQUENCE {		
workstation-identifier		INTEGER,	
deferral-mode		INTEGER {asap	(0),
		bnil	(1),
		bnig	(2),
		asti	(3) }
implicit-regeneration		INTEGER {suppressed	(0),
		allowed	(1) } }
Transformation-Primitives	::= SET {		
		[1] IMPLICIT WS-Window OPTIONAL,	
		[2] IMPLICIT WS-Viewport OPTIONAL,	
		[3] IMPLICIT Clipping-Rectangle OPTIONAL }	

```

WS-Window ::= SEQUENCE {
workstation-identifier INTEGER,
first-point Abs-Coord,
second-point Abs-Coord }

WS-Viewport ::= SEQUENCE {
workstation-identifier INTEGER,
xmin REAL,
xmax REAL,
ymin REAL,
ymax REAL }

Clipping-Rectangle ::= SEQUENCE {
first-point Abs-Coord,
second-point Abs-Coord }

```

A.3.10.3 Geometric coordinates

Coordinate data for geometric operations is stored in terms of normalized display coordinates in all three source data syntaxes. However, the exact details of the number format differ significantly between the approach taken in data syntaxes DS I and DS III and that taken in data syntax DS II. Since the purpose of the IDS is for interworking, differences with respect to the number format should be avoided. Therefore, within the IDS a simple numbering scheme based on the ASN.1 signed REAL data type is used. ASN.1 REAL numbers are self-delimiting and of arbitrary length, so there is no difficulty with precision and no need to assign special bit fields to determine the length of the number. A coordinate can therefore be represented as a pair of numbers. The mapping of a real data field to a numeric data field in any of the data syntaxes is dependent upon that particular data syntax. For the case of data syntaxes DS I and DS III the normalized unit display area is mapped to the fractional part (i.e. mantissa part) of the real number field. For DS II both the mantissa and exponent of the real number are used.

Since three dimensional coordinate specifications are optionally available in all of the data syntaxes, an integer triplet is optionally provided below. Because three dimensional operation is optional, the projection to two dimensions must be defined so that three dimensional information may be viewed in a two dimensional environment through interworking. A plane projection which assumes $Z = 0$ is used.

```

Coord ::= IMPLICIT CHOICE { Abs-Coord, Rel-Coord }

Abs-Coord ::= CHOICE { [1] X-Y,
                        [2] X-Y-Z }

X-Y ::= SEQUENCE { REAL, REAL }
-- Absolute X, Y Coordinates

X-Y-Z ::= SEQUENCE { REAL, REAL, REAL }
-- Absolute X, Y, Z Coordinates

Rel-Coord ::= CHOICE {
                        [3] DX-DY,
                        [4] DX-DY-DZ }

DX-DY ::= SEQUENCE { REAL, REAL }
-- Relative DX, DY Coordinates

DX-DY-DZ ::= SEQUENCE { REAL, REAL, REAL }
-- Relative DX, DY, DZ Coordinates

```

A.3.11 Animation control string

The capability to achieve dynamic or animated effects on the presentation device is highly dependent on the terminal model and display environment. Several of the terminal data syntaxes provide some specialized capabilities to achieve dynamic effects. For example, data syntaxes DS I and DS II include three phase flash (blink) capability and data syntax DS III includes a colour map phased blink function. The dynamic effects generated by these special functions will not in general be preserved in conversion. This is especially true since the order of the display of presentation entities may be altered by the conversion process to account for differences in the terminal model. Except for flash (blink) it is necessary for the conversion process to take into account dynamic effects even though it cannot convert them faithfully, since they may significantly alter the final resultant picture.

A sophisticated terminal model dependent animation capability is available in data syntax DS I. This capability makes use of a multi-plane terminal model in which the order and relative position of the various planes may be altered. The effects which may be generated by this capability are unique to the environment in which they were defined. The animation control commands from data syntax DS I must, however, be included in the interworking data syntax, since they affect the final result of the display. The conversion process must generate the correct final resultant picture.

Animation-Control-String ::= CHOICE { mvi-start [1] NULL,
mvi-stop [2] NULL,
mvi-repeat-start [3] MVI-Repeat-Start,
mvi-repeat-end [4] NULL,
mvi-move [5] MVI-Move }

-- MVI-Start is a function from Recommendation T.101 DS I (MVI Code set position 2/0)
-- MVI-Stop is a function from Recommendation T.101 DS I (MVI Code set position 2/1)

A.3.11.1 MVI-repeat start

MVI-Repeat-Start ::= SEQUENCE { GRAPHICSTRING, INTEGER }

-- General character (REPEAT START) from Recommendation T.101 DS I
-- (MVI Code set position 3/12 or 11/12), followed by a count of the number of repetitions
-- MVI-Repeat-End is a function from Recommendation T.101 DS I
-- (MVI Code set position 3/13 or 11/13)

A.3.11.2 MVI-move

MVI-Move ::= SEQUENCE { Move-Origin, Move-Termination, Move-Time }

-- MVI-Move is a function from Recommendation T.101 DS I
-- (MVI Code set position 3/10 or 11/10)

Move-Origin ::= Abs-Coord
-- X, Y Parameters codes as packed binary fractions

Move-Termination ::= OCTETSTRING
-- X, Y Parameters codes as packed binary fractions

Move-Time ::= INTEGER
-- Numeric count of the time period for the move operation in units of 1/10 of a second

A.3.12 Segment control string

Data syntax II provides an optional segment storage and editing capability. One or two storage memories for display segments are retained. Editing commands may produce dynamic effects by altering the stored display segment and causing the redisplay of the picture. A display segment may contain any geometric string data as well as the special segment attributes as described below.

Segment control is similar to animation control in that it provides functions which control special display environment dependent capabilities. Since analogous functions are not available in either data syntax I or III, these functions must be handled in the conversion process. For the conversion of information from data syntax II into data syntax I or III only one "Workstation" (or display screen) is used.

Segment-Control-String ::= CHOICE { [1] Work-Station-Dependent,
[2] Work-Station-Independent }

Work-Station-Dependent ::= CHOICE { [1] W-Create,
[2] W-Close,
[3] W-Rename,
[4] W-Delete-1,
[5] W-Delete-2,
[6] W-Redraw,
[7] W-Set-Highlight,
[8] W-Set-Visibility,
[9] W-Set-Seg-Transparent,
[10] W-Set-Priority }

A.3.12.1.1 W-create

W-Create ::= INTEGER

-- Open the identified segment.

A.3.12.1.2 W-close

W-Close ::= INTEGER

-- Close the identified segment.

A.3.12.1.3 W-rename

W-Rename ::= SEQUENCE {
 old-segment-number [1] INTEGER,
 new-segment-number [2] INTEGER }

-- Rename old segment number to new segment number.

A.3.12.1.4 W-delete-1

W-Delete-1 ::= SEQUENCE {
 work-station-id [1] INTEGER,
 segment-number [2] INTEGER }

-- Delete identified segment from workstation.

A.3.12.1.5 W-delete-2

W-Delete-2 ::= INTEGER

-- Delete the identified segment from all workstations.

A.3.12.1.6 W-redraw

W-Redraw ::= INTEGER

-- Redraw the identified workstation.

A.3.12.1.7 W-set highlight

W-Set-Highlight ::= SEQUENCE {
 highlight-segment-number [1] INTEGER,
 highlight-attribute [2] INTEGER }

-- Set highlight attribute of identified segment.

A.3.12.1.8 W-set visibility

W-Set-Visibility ::= SEQUENCE {
 visibility-segment-number [1] INTEGER,
 visibility-attribute [2] INTEGER }

-- Set visibility attribute of identified segment.

A.3.12.1.9 W-set segment transparent

W-Set-Seg-Transparent ::= SEQUENCE { transparent-segment-number [1] INTEGER,
 transform-matrix [2] MAT }

-- Set transformation matrix attributes for the identified segment.

MAT ::= SET { matrix-element-11 [11] REAL,
 matrix-element-12 [12] REAL,
 matrix-element-13 [13] REAL,
 matrix-element-21 [21] REAL,
 matrix-element-22 [22] REAL,
 matrix-element-23 [23] REAL }

-- Transform Matrix Definition.

A.3.12.1.10 W-set priority

W-Set-Priority ::= SEQUENCE { priority-segment-number [1] INTEGER,
 priority-value [2] REAL }

-- Set segment priority attribute for the identified segment.
This is analogous to display order priority.

A.3.12.2 Work station independent

Work-Station-Independent ::= CHOICE { [1] W-Associated,
[2] W-Copy,
[3] W-Insert }

A.3.12.2.1 W-associated

W-Associated ::= SEQUENCE { associated-w-station-id [1] INTEGER,
associated-segment-number [2] INTEGER }
-- Associate the identified segment with the identified work station.

A.3.12.2.2 W-copy

W-Copy ::= SEQUENCE { copy-w-station-id [1] INTEGER,
copy-segment-number [2] INTEGER }
-- Copy the primitives of the identified work station.

A.3.12.2.3 W-insert

W-Insert ::= SEQUENCE { insert-segment-number [1] INTEGER,
insert-transform-matrix-ref [2] MAT }
-- Transform and display segment.

A.3.13 Colour control string

All of the source terminal data syntaxes provide the capability to define colour and have available, at least optionally, a colour map capability. However the colour model which is used by each of the source terminal data syntaxes differs significantly. In order to provide a neutral basis for colour, the colour model developed for ISO 8613 Text and Offices Systems – Office Document Architecture is used here.

The basic colour model used in ISO 8613 is a colour cube in terms of the three Red, Green, Blue basic vectors. A colour in the colour model is represented by a three tuple of RGB components. Logically these colours are normalized from 0 (minimum) to 1 (maximum). Therefore 'Black' is the three tuple <0,0,0> and 'White' is the three tuple <1,1,1>. Since all of the videotex terminal data syntaxes support colour models which are different from this, the mapping of the specific colour models to the basic RGB colour cube must be understood by the conversion process.

Two colour indexing modes are available: Direct and Indexed. In direct colour selection, the colour is defined by providing a three tuple of discrete values for the RGB components. In the indexed colour selection mode, the colour is defined by an index into a single colour table of discrete colour values. The number of colours which may be defined in the colour table is terminal model dependent. The limit assumed in the definition of a particular set of data is specified in the Boundary Value Definition section. If a receiving system cannot image the range of colour values specified by a direct colour value or the colour value indexed by a colour index, then a 'closest match' is assumed according to the criteria stated in ISO 8613. A variant of the indexed colour mode called 'auxiliary colour mode' is used to define a colour for the background of a text or mosaic character cell.

Colour-Control-String ::= CHOICE {[1] Basic-Colour-Selection,
[2] Direct-Colour-Selection,
[3] Indexed-Colour-Selection,
[4] Auxiliary-Colour-Selection,
[5] Colour-Index-Setup }

A.3.13.1 Basic colour selection

Basic-Colour-Selection ::= INTEGER { black (0),
red (1),
green (2),
yellow (3),
blue (4),
magenta (5),
cyan (6),
white (7),
auxiliary-black (8),

auxiliary-red	(9),
auxiliary-green	(10),
auxiliary-yellow	(11),
auxiliary-blue	(12),
auxiliary-magenta	(13),
auxiliary-cyan	(14),
auxiliary-white	(15),
auxiliary-foreground	(16) }

- Several of the terminal data syntaxes provide a simplified way to access the basic primary colours by the use of C1 set codes. The various C1 control sets differ in a fundamental manner with respect to how the colour command interacts with other attributes. In order to avoid the difficulty in interworking, only the basic colour commands themselves are identified here. The range of a colour specification is terminal model dependent. This tendency has been avoided by defining all the colour selection commands in terms of the colour model specified in ISO 8613. Colour range dependencies (serial row attributes or parallel cell attributes) should be expressed in terms of the abstract colour by the conversion process. That is, all the rules inherent in the serial attribute method of specifying basic colours, or in the parallel attribute method, should be resolved by the conversion process which creates the IDS colour commands.
- The auxiliary colour commands specify the background colour for text and mosaics. The command 'auxiliary foreground' specifies that the background colour should be set to the current foreground colour.

A.3.13.2 Direct colour selection

Direct-Colour-Selection ::= SEQUENCE { REAL, REAL, REAL }

- Direct colour selection permits colours to be specified in terms of the Red Green Blue components of the colour model. The ASN.1 REAL data type is used since this form of number is self-delimiting and of arbitrary length. The real number parameters are relative to the maximum colour value for each component. The parameters are Red, Green and Blue respectively.

A.3.13.3 Indexed colour selection

Indexed-Colour-Selection ::= INTEGER

- Indexed colour selection permits colours to be specified as an index into an indirect colour map, which contains actual Red, Green and Blue colour specifications for each colour. The length of the colour map and the number of colour maps available is terminal model dependent. The INTEGER parameter is interpreted with respect to the current size of the colour map specified in Boundary Value Definition. In order to accommodate the rules for accommodating differences in the colour value extent, as specified in ISO 8613, the INTEGER parameter is interpreted as a normalized fraction of the specified map length. Some terminal data syntaxes provide the capability of multiple colour maps. Multiple maps are logically equivalent to one large map encompassing a number of submaps. In the IDS, the use of several colour maps is handled by arbitrarily partitioning the single IDS colour map.

A.3.13.4 Auxiliary colour selection

Auxiliary-Colour-Selection ::= INTEGER

- Auxiliary colour selection permits colours to be specified for the background of Text or Mosaics character cells. The operation of this command is similar to the Indexed Colour selection above, except that the current background colour is established.

A.3.13.5 Colour index set-up

Colour-Index-Set-up ::= SEQUENCE { INTEGER, REAL, REAL, REAL }

- The Colour Index set-up command defines the contents of the colour map. The first parameter takes indexes into the colour map in a similar manner to the Indexed Colour Selection command. The remaining three parameters define the Red, Green and Blue colour values in a manner similar to the Direct Colour Specification command.

A.3.14 Text colour string

The manner in which text is presented and the specialized attributes and constraints which pertain to the presentation of texts differs between each of the terminal data syntaxes.

Text-Control-String ::= CHOICE { [1] General-Text-Control, [2] Word-Wrap-Control }

A.3.14.1 General text control

General-Text-Control ::= SEQUENCE { [1] **General-Text-Control-Code,**
[2] **G-Text-Par1 OPTIONAL,**
[3] **G-Text-Par2 OPTIONAL,**
[4] **Rel-Coord OPTIONAL,**
[5] **Abs-Coord OPTIONAL }**

General-Text-Control-Code ::= GRAPHICSTRING

-- General control function from Recommendation T.101 DS III [PDI G Set position 2/2, (10/2)].

-- NOTE – PDI G Set has final character 5/7 within DS III.

G-Text-Par1 ::= SET { [1] Char-Rotation OPTIONAL,
[2] IMPLICIT Char-Path OPTIONAL,
[3] Char-Spacing OPTIONAL,
[4] IMPLICIT Text-Precision OPTIONAL,
[5] IMPLICIT Char-Expansion-Factor OPTIONAL,
[6] Text-Colour-Index OPTIONAL,
[7] IMPLICIT Text-Alignment OPTIONAL }

```
Char-Rotation ::= CHOICE { predefined [1] IMPLICIT INTEGER {
    char-rotation-0      (0),
    char-rotation-90     (1),
    char-rotation-180    (2),
    char-rotation-270    (3) }
    continuous [2] IMPLICIT SEQUENCE {
        height-vector Abs-Coord,
        width-vector Abs-Coord } }
```

Char-Path	::= INTEGER { char-path-right	(0),
	char-path-left	(1),
	char-path-up	(2),
	char-path-down	(3) }

Char-Spacing	::= CHOICE { predefined [1] IMPLICIT INTEGER {	
	char-spacing-1	(0),
	char-spacing-5/4	(1),
	char-spacing-3/2	(2) }
	continuous [2] IMPLICIT REAL }	

```

Text-Precision ::= INTEGER { string (0),
                                char (1),
                                stroke (2) }

```

Char-Expansion-Factor ::= REAL

[illegible]

Text-Alignment ::= SEQUENCE { **Horizontal-Alignment**,
Vertical-Alignment }

Horizontal-Alignment	::= INTEGER { normal	(0),
	left	(1),
	centre	(2),
	right	(3) }

Vertical-Alignment	::= INTEGER { normal	(0),
	top	(1),
	cap	(2),
	half	(3),
	base	(4),
	bottom	(5) }

G-Text-Par2	::= SEQUENCE { INTEGER {	cursor-style-underscore	(0),
		cursor-style-block	(1),
		cursor-style-cross-hair	(2),
		cursor-style-custom	(3) }

INTEGER { cursor-&-geometric-drawing-position-together (0),
 cursor-leads-geometric-drawing-position (1),
 geometric-drawing-position-leads-cursor (1),
 cursor-&-geometric-drawing-position-separate (3) }

INTEGER { char-interrow-spacing-1 (0),
 char-interrow-spacing-5/4 (1),
 char-interrow-spacing-3/2 (2),
 char-interrow-spacing-2 (3) }

Char-Block-Dimension }

-- The relative coordinates define the size of the character field.

Char-Block-Dimensions ::= Rel-Coord

A.3.14.2 Word wrap control

The capability to wrap the presentation of characters on a word boundary rather than on a character boundary is available in one of the terminal data syntaxes. This capability cannot be directly converted to other data syntaxes; however, the effect can be achieved in the converter by issuing appropriate format effector characters.

Word-Wrap-Control ::= INTEGER { Word-Wrap-On (1),
 Word-Wrap-Off (2) }

-- Word-Wrap-On is a function from Recommendation T.101 DS III [C1 set position 5/5, (9/5)].

-- Word-Wrap-Off is a function from Recommendation T.101 DS III [C1 set position 5/6, (9/6)].

A.3.15 Photographic string synthetic image

All of the terminal data syntaxes provide a method of handling an array of pixels. Some of the data syntaxes also provide general photographic capabilities which provide more efficient methods of encoding the same type of data. This means that interworking between all of the terminal data syntaxes is possible for photographic data, even though it may be inefficient in some cases.

Two classes of photographic images are identified below. They are the Synthetic and the Natural Image forms of photographic. The synthetic form of photographic corresponds to the photographic capabilities of data Syntax I. Natural Image photographic coding is for further study.

Photo-Graphic-String-Synthetic-Image ::= CHOICE {[1] Line-Dot-Pattern,
 [2] Line-Dot-Pattern-Comp,
 [3] Field-Dot-Pattern,
 [4] Colouring-Block,
 [5] Colouring-Block-Comp,
 [6] Field-Colouring-Block,
 [7] Field-Colouring-Block-Comp,
 [8] Free-Format-Colouring-Block }

-- Photographic Synthetic Image functions correspond to Recommendation T.101 Data Syntax I.
 These functions are suitable for displaying synthetic images such as Kanji characters, graphics, etc.

A.3.15.1 Line dot pattern

Line-Dot-Pattern ::= SEQUENCE { y-origin-point-coordinate-Idp Abs-Coord,
 dot-pattern-data-Idp BITSTRING }

-- Line-Dot-Pattern functions indicates a selection of two colours which are defined by a colouring Block, Field Colouring Block, etc. This function gives dot pattern data of one or several lines at a time.

A.3.15.2 Line dot pattern comp

Line-Dot-Pattern-Comp ::= SEQUENCE { y-origin-point-coordinate-Idpc Abs-Coord,
 mh-run-length coded-data BITSTRING }

-- The Line-Dot-Pattern-Comp function is equivalent to the Line-Dot-Pattern function except that the dot patterns are encoded in a compressed manner using the M.H. Run Length Code.

A.3.15.3 Field dot pattern

Field-Dot-Pattern ::= SEQUENCE {
 xy-origin-point-coordinate Abs-Coord,
 dx-dy-field-dimensions Rel-Coord,
 dot-pattern-data-fdp BITSTRING }

-- The Field-Dot-Pattern function is equivalent to the Line-Dot-Pattern function except that this function defines the dot pattern in a rectangular area.

A.3.15.4 Colouring block

Colouring-Block ::= SEQUENCE {
 fg-bg-da-existence-indicator INTEGER,
 y-origin-point-coordinate-cb Abs-Coord,
 SEQUENCE OF { SEQUENCE {
 fg-colour BITSTRING,
 bg-colour BITSTRING,
 display-attributes-cb BITSTRING } } }

-- The Colouring-Block function defines a photographic image by specifying the foreground colour (FG), background colour (BG), and display attributes of certain blocks ahead of which is indicated by the parameter y-origin-point-coordinate.

A.3.15.5 Colouring block comp

Colouring-Block-Comp ::= SEQUENCE {
 colouring-block-comp-function-id INTEGER,
 fg-bg-da-existence-indicator-cbc INTEGER,
 y-origin-point-coordinate-cbs Abs-Coord,
 SEQUENCE OF { SEQUENCE {
 fg-comp-colour BITSTRING,
 fg-runlength BITSTRING,
 bg-comp-colour BITSTRING,
 bg-runlength BITSTRING,
 display-attributes-cbc BITSTRING,
 da-runlength BITSTRING } } }

-- The Colouring-Block-Comp function is equivalent to that of the Colouring-Block function except that colour and display attributes data are encoded by compressed manner as run-length code.

A.3.15.6 Field colouring block

Field-Colouring-Block ::= SEQUENCE {
 field-colouring-block-function-id INTEGER,
 fg-bg-da-existence-indicator-fcb INTEGER,
 xy-origin-point-coordinate-fcb Abs-Coord,
 dx-dy-field-dimensions-fcb Rel-Coord,
 SEQUENCE OF { SEQUENCE {
 fg-colour-fcb BITSTRING,
 bg-colour-fcb BITSTRING,
 display-attributes-fcb BITSTRING } } }

-- The Field-Colouring-Block function defines a photographic image by specifying the foreground colour (FG), background colour (BG), and display attributes of certain blocks which are contained in the the field allocated by xy-origin-point-coordinate and the dx-dy-field-dimensions.

A.3.15.7 Field colouring block comp

Field-Colouring-Block-Comp ::= SEQUENCE {
 field-colouring-block-comp-function-id INTEGER,
 fg-bg-da-existence-indicator-fcbc INTEGER,
 xy-origin-point-coordinate-fcbc Abs-Coord,
 dx-dy-field-dimensions-fcbc Rel-Coord,
 SEQUENCE OF { SEQUENCE {
 fg-colour-fcbc BITSTRING,
 fg-runlength-fcbc BITSTRING,
 bg-comp-colour-fcbc BITSTRING,
 bg-runlength-fcbc BITSTRING,
 display-attributes-fcbc BITSTRING,
 da-runlength-fcbc BITSTRING } } }

-- The Field-Colouring-Block-Comp function is equivalent to that of the Field-Colouring-Block function except that colour and display attributes data are encoded by compressed manner as run-length code.

A.3.15.8 Free format colouring block

```
Free-Format-Colouring-Block ::= SEQUENCE { fg-bg-da-existence-indicator-ffcb INTEGER,
                                           fg-bg-da-code-length INTEGER,
                                           run-length-code-length-ffcb INTEGER,
                                           xy-origin-point-coordinate-ffcb Abs-Coord,
                                           dx-dy-field-dimensions-ffcb Rel-Coord,
                                           SEQUENCE OF { SEQUENCE {
                                                         fg-colour-ffcb BITSTRING,
                                                         runlength-ffcb BITSTRING,
                                                         bg-comp-colour-ffcb BITSTRING,
                                                         bg-runlength-ffcb BITSTRING,
                                                         display-attributes-ffcb BITSTRING,
                                                         da-runlength-ffcb BITSTRING } } }
```

-- The Free-Format-Colouring-Block function is equivalent to that of the Field-Colouring-Block-Comp function except that the code length of the Foreground, Background, Display Attributes and Run Length can be arbitrarily set.

A.3.16 Photographic string natural image

```
Photo-Graphic-String-Natural-Image ::= CHOICE { [0] IMPLICIT Header,
                                                  [1] IMPLICIT Transfer,
                                                  [2] IMPLICIT Table-Header,
                                                  [3] IMPLICIT Table-Transfer }
```

```
Header ::= SET {
    CHOICE {
        [0] IMPLICIT Components OPTIONAL,
        [1] IMPLICIT Resolution OPTIONAL,
        [2] IMPLICIT PixelPair OPTIONAL }
    CHOICE {
        [3] IMPLICIT BitsPerDisplay OPTIONAL,
        [4] IMPLICIT SamplingStructure OPTIONAL,
        [5] IMPLICIT Adpcm OPTIONAL,
        [6] IMPLICIT Adct OPTIONAL } }
```

```
Components ::= INTEGER { colorYU*V* (0),
                         monochrome (1) }
```

```
Resolution ::= INTEGER { 4-2-2 (0),
                         2-1-1 (1) }
```

```
PixelPair ::= SEQUENCE { PixHor, PixVer }
```

```
PixHor ::= INTEGER
```

-- Number of horizontal pixels.

```
PixVer ::= INTEGER
```

-- Number of vertical pixels.

```
BitsPerDisplay ::= SEQUENCE OF INTEGER { 8 bits/pixel (0),
                                           1 bit/pixel (1),
                                           2 bits/pixel (2),
                                           ... 9 bits/pixel (9),... }
```

-- One value per component, gives the number or grey or colours a pixel may have.

```
SamplingStructure ::= SEQUENCE {
    spatial { INTEGER { line and orthogonal (0),
                       line and orthogonal field quincunx (1),
                       line quincunx field orthogonal (2),
                       line orthogonal single field (3),
                       line quincunx single field (4) } } }
```

```
temporal { INTEGER { coincident (0),
                    alternate samples (1),
                    sequential line (2) } }
```

```
Adpcm ::= SEQUENCE { INTEGER { Type dpcm (1) },
                     INTEGER { Subtype 1 dimension (0) } }
```

```
Adct ::= SEQUENCE { INTEGER { Type transform (2) },
                     INTEGER { Subtype Cosine (1) },
                     INTEGER { Subtype 2 dimension (0) } }
```

Transfer ::= SET { Origin, Area, Data }

Origin ::= [0] IMPLICIT PixelPair OPTIONAL

Area ::= [1] IMPLICIT PixelPair OPTIONAL

Data ::= CHOICE { [2] IMPLICIT OCTETSTRING OPTIONAL,
-- Any value from 4/0 to 7/F.
[3] IMPLICIT OCTETSTRING OPTIONAL }
-- Transparent mode 8 bit/octet.

TableHeader ::= SET { TableSet, TableSize }

TableSet ::= [0] IMPLICIT SEQUENCE { type ::= INTEGER,
number ::= INTEGER }

TableSize ::= [1] IMPLICIT SEQUENCE { depth ::= INTEGER,
height ::= INTEGER,
width ::= INTEGER OPTIONAL }

TableTransfer ::= SET { TableSet, Position, Data }

Position ::= TableSize

A.3.17 Macro

A Macro capability is available within the syntax of two of the three terminal data syntaxes. This capability permits strings of presentation data to be grouped together, so that it may be executed by the reference to a single command. In essence both terminal data syntax DS I and DS III provide the same Macro capability; however, a Macro in one data syntax cannot in general be converted to a Macro in another data syntax. This is because a Macro may contain any string of presentation data. Since the Terminal Models of the various data syntaxes differ, it is often necessary to sort the commands in the data stream in order to achieve the intended presentation effect. The arbitrary grouping of information into Macros prevents general sorting. Since the purpose of regular Macro functions are to achieve communications efficiency by eliminating the communication of repetitious code, it is possible to expand Macros in the conversion process. The conversion of a Macro is therefore the string of presentation data which it represents.

Two special forms of Macros in data syntax DS I and DS III are Key Activated Macros and Transit Macros. Key Activated Macros link the execution of the Macro function to a local Key on the terminal. Since this operation depends upon the interaction of the user, the contents of the Macro cannot be expanded in the converter ahead of time. The converter must re-transmit the entire page of information to the terminal with the contents of the Key Activated Macro sorted and factored into the page. This problem must be handled by the Interworking Presentation Architecture. Similarly Transit Macro provides a problem in conversion. The contents of a Transit Macro must be sent back to the source upon a user interaction. In interworking this could mean that data syntax DS I data might be contained within a Transit Macro in a data syntax DS III terminal after a conversion so that it might be sent back to the source unchanged. It is necessary to be able to identify entire coding environments or to identify uniquely each code table in each Data Syntax in order to avoid confusion.

MACRO-String ::= CHOICE { [1] Define-Macro,
[2] Define-and-Execute-Macro,
[3] Define-Transmit-Macro,
[4] Define-End-of-Macro-Definition,
[5] Macro-Invocation }
-- Key Activated Macros are Macros with reference numbers 0 to 7 in data syntax DS III.

A.3.17.1 Define macro

Define-Macro ::= SEQUENCE { SID, INTEGER }

-- General control character (DEF MACRO) from Recommendation T.101 DS III [C1 set position 4/0, (8/0)] and (P-DEF MACRO) from DS I [C1 set position 5/5, (9/5) followed by parameter 4/0].

-- Integer number from 0 to 95 corresponding to the Macro reference number of the Macro being defined.

A.3.17.2 Define and execute macro

Define-and-Execute-Macro ::= SEQUENCE { SID, INTEGER }

- General control character (DEFP MACRO) from Recommendation T.101 DS III [C1 set position 4/1, (8/1)] and (P-DEFP MACRO) from DS I [C1 set position 5/5, (9/5) followed by parameter 4/1].
- Integer number from 0 to 95 corresponding to the Macro reference number of the Macro being defined.

A.3.17.3 Define transit macro

Define-Transmit-Macro ::= SEQUENCE { SID, INTEGER }

- General control character (DEFT MACRO) from Recommendation T.101 DS III [C1 set position 4/2, (8/2)] and (P-DEFT MACRO) from DS I [C1 set position 5/5, (9/5) followed by parameter 4/2].
- Integer number from 0 to 95 corresponding to the Macro reference number of the Macro being defined.

A.3.17.4 Define end-of-macro definition

Define-End-of-Macro-Definition ::= SID

- General control character [END (Macro)] from Recommendation T.101 DS III [C1 set position 4/5, (8/5)] and (END MACRO) from DS I [C1 set position 5/5, (9/5) followed by parameter 4/15].

A.3.17.5 Macro invocation

Macro-Invocation ::= INTEGER

- Integer number from 0 to 95 corresponding to the Macro reference number of the Macro being invoked.
- NOTE – Macros may invoke other Macros at any time and to any depth.

A.3.18 DRCS string

The Dynamically Redefinable Character Set (DRCS) capability allows additional text or mosaic characters to be defined and used as regular alphanumeric text or mosaics. All three of the terminal data syntaxes include a form of DRCS capability; however, the operation of DRCS is quite different in the various Display Environments. In general, it is not possible to convert exactly from one type of DRCS to another because of the boundary conditions imposed by each of the Terminal Data Syntaxes. Different limits exist on the number of DRCS characters which may be defined or the amount of memory which may be used to store DRCS characters. The definition of DRCS characters is a particular difficulty. One of the source terminal data syntaxes takes the approach of allowing any presentation information to be used in the definition of a DRCS character, including geometric drawing commands, bit (photographic) and text and even other DRCS characters. The other two source data syntaxes define DRCS characters using a bit oriented (photographic) approach. Even the two photographic approaches to the definition of DRCS are not equivalent since they have different pixel densities and serious quantization errors may result from mapping an array of pixels to another array of a different size. Three forms of DRCS definition are included in the Interworking Data Syntax to accommodate the requirements of the three source data syntaxes. The conversion process would therefore have sufficient information to make the best conversion possible.

DRCS-String ::= CHOICE { [1] Define-DRCS-Type-I-1byte,
[2] Define-DRCS-Type-I-2byte,
[3] Define-DRCS-Type-II,
[4] Define-DRCS-Type-III,
[5] End-of-DRCS-Definition-Type-III,
[6] DRCS-Invocation,
[7] DRCS-Invocation-2byte }

A.3.18.1 Define DRCS Type-I 1 byte

Define-DRCS-Type-I-1byte ::= SEQUENCE { DRCS-I-Char-Size,
DRCS-I-Code,
DRCS-I-Data }

DRCS-I-Char-Size ::= INTEGER { normal-size (1),
medium-size (2),
small-size (3) }

DRCS-I-Code ::= INTEGER

- Integer number from 0 to 95 corresponding to the DRCS reference number of the 1 byte DRCS being invoked.

DRCS-I-Data ::= BITSTRING

A.3.18.2 Define DRCS Type-I 2 byte

Define-DRCS-Type-I-2byte ::= SEQUENCE { **DRCS-I-Char-Size**,
DRCS-I-Code,
DRCS-I-Data }

-- This structure is the same as "Define-DRCS-Type-I-1byte" except that "DRCS-I-Code" is an integer number from 0 to 8835 corresponding to the DRCS reference number of the 2-byte DRCS being invoked.

A.3.18.3 Define DRCS Type-II

Define-DRCS-Type-II ::= SEQUENCE { [1] **IMPLICIT DRCS-Header OPTIONAL**,

-- Description of general properties of the DRCS to be loaded. It is applied for all subsequent DRCS-pattern transfer units.

[2] **IMPLICIT DRCS-Pattern OPTIONAL** }

-- Actual pattern data.

DRCS-Header ::= SEQUENCE { **Identification-of-Char-Set**,
Select-Dot-Composition }

Identification-of-Char-Set ::= SEQUENCE { **repertory-info SET** {
repertory-# INTEGER { **first repertory** (1),
second repertory (2) },
delete-existing-dracs BOOLEAN,
registration-info CHOICE {
iso-registration [1] **IMPLICIT GRAPHICSTRING**,
private-dracs-# [2] **IMPLICIT INTEGER** } } }

Select-Dot-Composition ::= SEQUENCE { **Character-Cell-Structure**,
Blocking-Factor,
Pixel-Characteristics }

Character-Cell-Structure ::= CHOICE { **matrix-dimensions** [1] **IMPLICIT SEQUENCE** {
horizontal **INTEGER**,
vertical **INTEGER** },

-- According to SDC Type 1.

predefined-matrices [2] **IMPLICIT INTEGER** {
n16*24 (0),
n16*20 (1),
n16*12 (2),
n16*10 (3),
n12*24 (4),
n12*20 (5),
n12*12 (6),
n12*10 (7),
n8*12 (8),
n8*10 (9),
n6*12 (10),
n6*10 (11),
n6*5 (12),
n4*10 (13),
n4*5 (14),
n6*6 (15) } }

-- According to SDC Type 2.

Blocking-Factor ::= SEQUENCE { **horizontal INTEGER**,
vertical INTEGER }

-- Grouping of character cells, which are considered as a single character cell during character description.

Pixel-Characteristics ::= CHOICE { **number of bits** [1] **IMPLICIT INTEGER**,
predefined-numbers [2] **IMPLICIT INTEGER**
basic-DRCS (1),

-- 1 bit/dot.

four-colour-DRCS (4),

-- Black, red, green, yellow from 'Basic-Colour-Selection'.

eight-colour-DRCS (8),

-- First 8 colours from 'Basic-Colour-Selection'.

sixteen-colour-DRCS (16) }

-- 16 redefinable colours.

-- This data type describes the pattern for the characters of the down-loaded DRCS, according the last transmitted header unit. It contains no compression for the pattern data. Data Syntax I and III have no similar encodings and this method can be used for an adequate mapping. All encoding different from the direct method and the codes for improvement of the efficiency (S-bytes) have to be transformed to the following description.

DRCS-Pattern ::= SEQUENCE { first character GRAPHICSTRING,

-- Code of the first character or character block

pattern-units SEQUENCE OF {
pattern-block-# SEQUENCE OF INTEGER,
pattern-block BIT STRING } }

-- Each pattern block contains one bit of each of the dots, starting from the top left hand corner, running row by row from left to right. The pattern block numbers are ordered from the least significant bit on. If the pattern block is preceded by two or more block numbers, the pattern block is applied to all of them. The block numbers are in the range of 0 to 'pixel-characteristics'-1. The length of the pattern block equals to the number of pixels in the block.

A.3.18.4 Define DRCS Type-III

Define-DRCS-Type-III ::= INTEGER

-- A function from Recommendation T.101 DS III [C1 set position 4/3, (8/3)].

-- Integer number from 0 to 95 corresponding to the DRCS reference number of the DRCS character being defined to be followed by data string.

A.3.18.5 End-of-DRCS definition Type-III

End-of-DRCS-Definition-Type-III ::= GRAPHICSTRING

-- General control character [END (DRCS)] from Recommendation T.101 DS III [C1 set position 4/5, (8/5)].

A.3.18.6 DRCS invocation

DRCS-Invocation ::= INTEGER

-- Integer number from 0 to 95 corresponding to the DRCS reference number of the DRCS being invoked.

A.3.18.7 DRCS invocation 2 byte

DRCS-Invocation-2byte ::= INTEGER

-- Integer number from 0 to 8835 corresponding to the DRCS reference number of the 2-byte DRCS being invoked.

A.3.19 Fill pattern control string

The capability to fill a geometrically defined area with an arbitrary Fill Pattern, interior style, hatch or texture, is provided in two of the source videotex data syntaxes. Since one of the terminal Videotex data syntaxes, data syntax DS I, does not provide this capability it must be accommodated in the conversion process by assigning distinguishing colours or other means to indicate the difference between patterned areas. The method by which this capability is supported in the other two source data syntaxes is quite different. Data syntax DS III provides four predefined texture patterns, including solid fill, and four redefinable texture masks. These masks are rectilinear and are referenced to the origin of the normalized display area. This means that abutting areas filled with the same pattern will align perfectly. In Interior style patterns defined in data syntax DS II, the pattern may be defined on a parallelogram shaped area and is referenced to the origin of the area. Data syntax DS II also provides eight predefined fill patterns (hatch patterns). In general, any texture of interior style pattern may be simulated in the conversion process; however, secondary effects such as exact alignment of patterns cannot be guaranteed. Texture patterns in data syntax DS III are defined by including any string of presentation data in the definition of the pattern whereas interior styles defined in data syntax DS II are defined in terms of a cell array. The conversion process must resolve the pattern before the conversion. Limits to global variables, such as the available amount of texture memory, is defined by the boundary value condition indicators in the state vector.

A.3.20.2 Music code sequence

Music-Control-Sequence ::= GRAPHICSTRING

-- Control characters from Recommendation T.101 DS I (Musical Control C1 Set). The Musical Control set contains the functions: Start Music Sequence, End Music Sequence, Start Melody Part, Start Rythm Part, End Part, Music Label, Jump to Part, Music Repeat, Music Branch, Sound Level, Change of Timbre, Long Duration Rest or Tone. Reference is made to this Recommendation since this code table has not yet been registered.

A.3.21 Telesoftware string

Telesoftware-String ::= Further Study.

A.3.22 Audio data string

Audio-Data-String ::= Further Study.

Appendix I

(to Recommendation T.101, Annex A)

Text and mosaic character repertoires

In the Interworking Data Syntax all text and mosaic characters are assigned a code name so that they may be uniquely identified. An exhaustive Repertoire of all of the text and mosaic characters used in the data syntaxes specified in this Recommendation is presented below. This simplifies many of the references to graphic character sets used in the IDS since only the code names need be used in the body of the ASN.1 description of the Interworking Data Syntax. None of the videotex data syntaxes make use of all of the text and mosaic characters identified below. There are areas where there is a large amount of overlap between the various data syntaxes. In order to aid transcoding and conversions several categories have been identified. Separate repertoires have been defined for each of these categories.

I.1 Repertoire I – Common alphanumeric text characters

Repertoire I contains the common repertoire of basic alphanumeric text characters. These characters are taken from the primary and supplementary characters from Recommendation T.51, with registered final characters 4/0 and 6/2 respectively. In addition, this includes the SPACE character (SP01) and the DELETE character (SM34). The descriptive names given to characters differ between the various terminal data syntaxes defined in this Recommendation and between the repertoire of alphanumeric characters defined in ISO Standards ISO 6937. Composite names are used here, which endeavour to include the full range of meanings specified in the different terminal data syntaxes identified in this Recommendation, and achieve the maximum level of commonality with ISO 6937.

I.1.1 Latin alphabetic characters

Name Code	Descriptive name	LA28	Capital A with ring
		LA31	Small a with macron
		LA32	Capital A with macron
LA01	Small a	LA43	Small a with ogonek
LA02	Capital A	LA44	Capital A with ogonek
LA11	Small a with acute accent	LA51	Small æ diphthong
LA12	Capital A with acute accent	LA52	Capital AE diphthong
LA13	Small a with grave accent	LB01	Small b
LA14	Capital A with grave accent	LB02	Capital B
LA15	Small a with circumflex accent	LC01	Small c
LA16	Capital A with circumflex accent	LC02	Capital C
LA17	Small a with diaeresis or umlaut mark	LC11	Small c with acute accent
LA18	Capital A with diaeresis or umlaut mark	LC12	Capital C with acute accent
LA19	Small a with tilde	LC15	Small c with circumflex accent
LA20	Capital A with tilde	LC16	Capital C with circumflex accent
LA23	Small a with breve		
LA24	Capital A with breve		
LA27	Small a with ring		

Name
Code

Descriptive name

<i>Name Code</i>	<i>Descriptive name</i>	<i>Name Code</i>	<i>Descriptive name</i>
LC21	Small c with caron	LI18	Capital I with diaeresis or umlaut mark
LC22	Capital C with caron	LI19	Small i with tilde
LC29	Small c with dot above	LI20	Capital I with tilde
LC30	Capital C with dot above	LI30	Capital I with dot above
LC41	Small c with cedilla	LI31	Small i with macron
LC42	Capital C with cedilla	LI32	Capital I with macron
LD01	Small d	LI43	Small i with ogonek
LD02	Capital D	LI44	Capital I with ogonek
LD21	Small d with caron	LI51	Small ij ligature
LD22	Capital D with caron	LI52	Capital IJ ligature
LD61	Small d with stroke	LI61	Small i without dot
LD62	Capital D with stroke (Icelandic eth)	LJ01	Small j
LD63	Small eth, Icelandic	LJ02	Capital J
LE01	Small e	LJ15	Small j with circumflex accent
LE02	Capital E	LJ16	Capital J with circumflex accent
LE11	Small e with acute accent	LK01	Small k
LE12	Capital E with acute accent	LK02	Capital K
LE13	Small e with grave accent	LK41	Small k with cedilla
LE14	Capital E with grave accent	LK42	Capital K with cedilla
LE15	Small e with circumflex accent	LK61	Small k, Greenlandic
LE16	Capital E with circumflex accent	LL01	Small l
LE17	Small e with diaeresis or umlaut	LL02	Capital L
LE18	Capital E with diaeresis or umlaut	LL11	Small l with acute accent
LE21	Small e with caron	LL12	Capital L with acute accent
LE22	Capital E with caron	LL21	Small l with caron
LE29	Small e with dot above	LL22	Capital L with caron
LE30	Capital E with dot above	LL41	Small l with cedilla
LE31	Small e with macron	LL42	Capital L with cedilla
LE32	Capital E with macron	LL61	Small l with stroke
LE43	Small e with ogonek	LL62	Capital L with stroke
LE44	Capital E with ogonek	LL63	Small l with middle dot
LF01	Small f	LL64	Capital L with middle dot
LF02	Capital F	LM01	Small m
LG01	Small g	LM02	Capital M
LG02	Capital G	LN01	Small n
LG11	Small g with acute accent	LN02	Capital N
LG15	Small g with circumflex accent	LN11	Small n with acute accent
LG16	Capital G with circumflex accent	LN12	Capital N with acute accent
LG23	Small g with breve	LN19	Small n with tilde
LG24	Capital G with breve	LN20	Capital N with tilde
LG29	Small g with dot above	LN21	Small n with caron
LG30	Capital G with dot above	LN22	Capital N with caron
LG42	Capital G with cedilla	LN41	Small n with cedilla
LH01	Small h	LN42	Capital N with cedilla
LH02	Capital H	LN61	Small eng, Lapp
LH15	Small h with circumflex accent	LN62	Capital eng, Lapp
LH16	Capital H with circumflex accent	LN63	Small n with apostrophe
LH61	Small h with stroke	LO01	Small o
LH62	Capital H with stroke	LO02	Capital O
LI01	Small i	LO11	Small o with acute accent
LI02	Capital I	LO12	Capital O with acute accent
LI11	Small i with acute accent	LO13	Small o with grave accent
LI12	Capital I with acute accent	LO14	Capital O with grave accent
LI13	Small i with grave accent	LO15	Small o with circumflex accent
LI14	Capital I with grave accent	LO16	Capital O with circumflex accent
LI15	Small i with circumflex accent	LO17	Small o with diaeresis or umlaut mark
LI16	Capital I with circumflex accent	LO18	Capital O with diaeresis or umlaut mark
LI17	Small i with diaeresis or umlaut mark	LO19	Small o with tilde

<i>Name Code</i>	<i>Descriptive name</i>
LO20	Capital O with tilde
LO25	Small o with double acute accent
LO26	Capital O with double acute accent
LO31	Small o with macron
LO32	Capital O with macron
LO51	Small œ ligature
LO52	Capital OE ligature
LO61	Small o with slash
LO62	Capital O with slash
LP01	Small p
LP02	Capital P
LQ01	Small q
LQ02	Capital Q
LR01	Small r
LR02	Capital R
LR11	Small r with acute accent
LR12	Capital R with acute accent
LR21	Small r with caron
LR22	Capital R with caron
LR41	Small r with cedilla
LR42	Capital R with cedilla
LS01	Small s
LS02	Capital S
LS11	Small s with acute accent
LS12	Capital S with acute accent
LS15	Small s with circumflex accent
LS16	Capital S with circumflex accent
LS21	Small s with caron
LS22	Capital S with caron
LS41	Small s with cedilla
LS42	Capital S with cedilla
LS61	Small sharp s, German
LT01	Small t
LT02	Capital T
LT21	Small t with caron
LT22	Capital T with caron
LT41	Small t with cedilla
LT42	Capital T with cedilla
LT61	Small t with stroke
LT62	Capital T with stroke
LT63	Small thorn, Icelandic
LT64	Capital thorn, Icelandic
LU01	Small u
LU02	Capital U

I.1.2 Non-alphabetic characters

I.1.2.1 Decimal digits

<i>Name Code</i>	<i>Descriptive name</i>
ND01	Digit 1
ND02	Digit 2
ND03	Digit 3
ND04	Digit 4
ND05	Digit 5

<i>Name Code</i>	<i>Descriptive name</i>
LU11	Small u with acute accent
LU12	Capital U with acute accent
LU13	Small u with grave accent
LU14	Capital U with grave accent
LU15	Small u with circumflex accent
LU16	Capital U with circumflex accent
LU17	Small u with diaeresis or umlaut mark
LU18	Capital U with diaeresis or umlaut mark
LU19	Small u with tilde
LU20	Capital U with tilde
LU23	Small u with breve
LU24	Capital U with breve
LU25	Small u with double acute accent
LU26	Capital U with double acute accent
LU27	Small u with ring
LU28	Capital U with ring
LU31	Small u with macron
LU32	Capital U with macron
LU43	Small u with ogonek
LU44	Capital U with ogonek
LV01	Small v
LV02	Capital V
LW01	Small w
LW02	Capital W
LW15	Small w with circumflex accent
LW16	Capital W with circumflex accent
LX01	Small x
LX02	Capital X
LY01	Small y
LY02	Capital Y
LY11	Small y with acute accent
LY12	Capital Y with acute accent
LY15	Small y with circumflex accent
LY16	Capital Y with circumflex accent
LY17	Small y with diaeresis or umlaut mark
LY18	Capital Y with diaeresis or umlaut mark
LZ01	Small z
LZ02	Capital Z
LZ11	Small z with acute accent
LZ12	Capital Z with acute accent
LZ21	Small z with caron
LZ22	Capital Z with caron
LZ29	Small z with dot above
LZ30	Capital Z with dot above

<i>Name Code</i>	<i>Descriptive name</i>
ND06	Digit 6
ND07	Digit 7
ND08	Digit 8
ND09	Digit 9
ND10	Digit 0

I.1.2.2 Currency signs

<i>Name Code</i>	<i>Descriptive name</i>
SC01	General currency sign
SC02	Pound sign
SC03	Dollar sign

<i>Name Code</i>	<i>Descriptive name</i>
SC04	Cent sign
SC05	Yen sign

I.1.2.3 Punctuation marks

<i>Name Code</i>	<i>Descriptive name</i>
SP01	SPACE
SP02	Exclamation mark
SP03	Inverted exclamation mark
SP04	Quotation mark
SP05	Apostrophe
SP06	Opening (left) parenthesis
SP07	Closing (right) parenthesis
SP08	Comma
SP10	Hyphen or minus sign
SP11	Period (or decimal point)
SP12	Solidus (slant)

<i>Name Code</i>	<i>Descriptive name</i>
SP13	Colon
SP14	Semicolon
SP15	Question mark
SP16	Inverted question mark
SP17	Angle quotation mark left
SP18	Angle quotation mark right
SP19	Single quotation mark left
SP20	Single quotation mark right
SP21	Double quotation mark left
SP22	Double quotation mark right

I.1.2.4 Arithmetic signs

<i>Name Code</i>	<i>Descriptive name</i>
SA01	Plus sign
SA02	Plus/minus sign
SA03	Less-than sign
SA04	Equals sign

<i>Name Code</i>	<i>Descriptive name</i>
SA05	Greater-than sign
SA06	Divide sign
SA07	Multiply sign

I.1.2.5 Subscripts and superscripts

<i>Name Code</i>	<i>Descriptive name</i>
NS01	Superscript 1
NS02	Superscript 2

<i>Name Code</i>	<i>Descriptive name</i>
NS03	Superscript 3

I.1.2.6 Fractions

<i>Name Code</i>	<i>Descriptive name</i>
NF01	Fraction 1/2
NF04	Fraction 1/4
NF05	Fraction 3/4
SM39	Fraction 1/8 (equivalent to NF18)

<i>Name Code</i>	<i>Descriptive name</i>
SM40	Fraction 3/8 (equivalent to NF19)
SM41	Fraction 5/8 (equivalent to NF20)
SM42	Fraction 7/8 (equivalent to NF21)

I.1.2.7 Miscellaneous

<i>Name Code</i>	<i>Descriptive name</i>
SM01	Number sign
SM02	Percent sign
SM03	Ampersand
SM04	Star, asterisk
SM05	Commercial at
SM06	Opening (left) square bracket

SM07	Reverse solidus
SM08	Closing (right) square bracket
SM11	Opening brace, left curly bracket
SM12	Central horizontal bar jointive
SM13	Central vertical line jointive
SM14	Closing brace, right curly bracket

<i>Name Code</i>	<i>Descriptive name</i>
----------------------	-------------------------

<i>Name Code</i>	<i>Descriptive name</i>	<i>Name Code</i>	<i>Descriptive name</i>
SM17	Micro sign	SM36	Copyright sign (equivalent to SM52)
SM18	Ohm sign	SM37	Trademark sign (equivalent to SM54)
SM19	Degree sign	SM38	Musical symbol (equivalent to SM93)
SM20	Ordinal indicator, masculine	SM43	Arrowhead upwards, circumflex shape
SM21	Ordinal indicator, feminine	SM44	Upper reverse solidus, grave accent shape
SM24	Section sign	SM47	Upper bar (not jointive) bar or tilde shape
SM25	Paragraph sign, pilcrow	SM48	Lower bar (not jointive) low line, spacing underline (equivalent to SP09 of ISO 6937)
SM26	Middle dot	SM49	Non-spacing underline
SM30	Leftward arrow		
SM31	Rightward arrow		
SM32	Upward arrow		
SM33	Downward arrow		
SM34	DELETE		
SM35	Registered sign (equivalent to SM53)		

NOTE – The characters SM43, SM44, SM47, SM48 have multiple names since the descriptive names for these characters differ significantly between the various terminal data syntaxes defined in this Recommendation. These characters were originally intended as accent characters in the original usage of the IRV code table of ISO 646. This meaning has changed since the introduction of the composite method of coding accented characters defined in CCITT Recommendation T.51, ISO 6937 and several Recommendations. As a result these characters should not be used to generate accented characters. To retain compatibility with previous standards, multiple names are described here.

I.1.2.8 Diacritical marks (as displayed when used in conjunction with SPACE)

<i>Name Code</i>	<i>Descriptive name</i>	<i>Name Code</i>	<i>Descriptive name</i>
SD11	Acute accent	SD25	Double acute accent
SD13	Grave accent	SD27	Ring
SD15	Circumflex accent	SD29	Dot above
SD17	Umlaut or diaeresis	SD31	Macron
SD19	Tilde	SD41	Cedilla
SD21	Caron	SD43	Ogonek
SD23	Breve		

I.2 Repertoire 2 – Special alphanumeric text characters

These characters are unique to one or two of the terminal data syntaxes and the presentation of these characters must be converted so that their presentation effect may be achieved in other terminal data syntax.

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
SM45	Left vertical bar jointive	SM50	Non spacing vector overbar
SM46	Right vertical bar jointive	SM51	Non spacing slant

NOTE – The name codes SM50 and SM51 are introduced here since name codes for these characters are not included in either the ISO registry (Registered code table number 99) or in this Recommendation.

I.3 Repertoire 3 – Kana characters

This entire set of characters is unique to only one of the terminal data syntaxes and therefore the presentation of these characters must be converted so that their presentation effect may be achieved in another terminal data syntax.

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
JA01	Katakana full stop	JA04	Katakana comma
JA02	Katakana opening bracket	JA05	Katakana conjunctive symbol
JA03	Katakana closing bracket	JA06	Katakana WO
<i>Name code</i>	<i>Descriptive name</i>		

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
JA07	Katakana small a	JA36	Katakana TO
JA08	Katakana small i	JA37	Katakana NA
JA09	Katakana small u	JA38	Katakana NI
JA10	Katakana small e	JA39	Katakana NU
JA11	Katakana small o	JA40	Katakana NE
JA12	Katakana ya	JA41	Katakana NO
JA13	Katakana small yu	JA42	Katakana HA
JA14	Katakana yo	JA43	Katakana HI
JA15	Katakana tsu	JA44	Katakana FU
JA16	Prolonged Sound Symbol	JA45	Katakana HE
JA17	Katakana A	JA46	Katakana HO
JA18	Katakana I	JA47	Katakana MA
JA19	Katakana U	JA48	Katakana MI
JA20	Katakana E	JA49	Katakana MU
JA21	Katakana O	JA50	Katakana ME
JA22	Katakana KA	JA51	Katakana MO
JA23	Katakana KI	JA52	Katakana YA
JA24	Katakana KU	JA53	Katakana YU
JA25	Katakana KE	JA54	Katakana YO
JA26	Katakana KO	JA55	Katakana RA
JA27	Katakana SA	JA56	Katakana RI
JA28	Katakana SHI	JA57	Katakana RU
JA29	Katakana SU	JA58	Katakana RE
JA30	Katakana SE	JA59	Katakana RO
JA31	Katakana SO	JA60	Katakana WA
JA32	Katakana TA	JA61	Katakana N or M
JA33	Katakana CHI	JA62	Voiced sound symbol
JA34	Katakana TSU	JA63	semi-voiced sound symbol
JA35	Katakana TE		

NOTE – Since name codes are not included for these characters in either the ISO registry (Registration number 13) or in this Recommendation, codes beginning with JA are introduced here.

I.4 Repertoire 4 – Kanji characters

These characters occur in only one of the terminal data syntaxes although the registered Kanji character code table is sometimes used in combination with the sets of another terminal data syntax. Except in the case when Kanji character capability is available at both ends of an interchange, the presentation of these characters must be converted so that their presentation effect may be achieved in the other terminal data syntax. The Kanji character set is registered as a two byte set. This Recommendation uses a sub-Repertoire of this set which includes some 3639 characters, of which 2980 are Kanji symbol characters. These characters are unique to this Repertoire. The remaining characters, including a number of special characters as well as some alphabets for other languages, overlap with other registered character sets. For example a Cyrillic, Greek, Katakana, and Hiragana alphabet are included, as well as all but 11 of the Latin alphabet characters of Repertoire 1. The overlap characters in this code table have the same *name codes* as the equivalent characters in Repertoire 1 and can therefore be converted directly between data syntaxes. Special handling is required for the other characters. In addition there are 32 drawing characters which overlap those in Repertoire 8. These also share the same *name codes* as those in other character sets and therefore are part of the greater drawing character Repertoire (Repertoire 8).

The Repertoire of Kanji pictorial characters and unique special characters is voluminous. Since conversions from this Repertoire of characters require special handling, it is not necessary to list the Repertoire here. For reference, see Data Syntax I of this Recommendation which is a sub Repertoire of the ISO registration number 87. The name codes Kanji JK01 to JK2980, HK01 to HK83, and JS01 to JS366 have been introduced to identify the Kanji pictorial characters, the Hiragana characters, and the Kanji Special characters respectively.

I.5 Repertoire 5 – Greek characters

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
GA01	Lower case greek letter Alpha	GN02	Upper case greek letter Nu
GA02	Upper case greek letter Alpha	GO01	Lower case greek letter Omicron
GA11	Lower case greek letter Alpha with accent	GO02	Upper case greek letter Omicron
GA12	Upper case greek letter Alpha with accent	GO11	Lower case greek letter Omicron with accent
GB01	Lower case greek letter Beta	GO12	Upper case greek letter Omicron with accent
GB02	Upper case greek letter Beta	GO61	Lower case greek letter Omega
GD01	Lower case greek letter Delta	GO62	Upper case greek letter Omega
GD02	Upper case greek letter Delta	GO63	Lower case greek letter Omega with accent
GE01	Lower case greek letter Epsilon	GO64	Upper case greek letter Omega with accent
GE02	Upper case greek letter Epsilon	GP01	Lower case greek letter Pi
GE11	Lower case greek letter Epsilon with accent	GP02	Upper case greek letter Pi
GE12	Upper case greek letter Epsilon with accent	GP61	Lower case greek letter Psi
GE61	Lower case greek letter Eta	GP62	Upper case greek letter Psi
GE62	Upper case greek letter Eta	GR01	Lower case greek letter Rho
GE63	Lower case greek letter Eta with accent	GR02	Upper case greek letter Rho
GE64	Upper case greek letter Eta with accent	GS01	Lower case greek letter Sigma
GF01	Lower case greek letter Phi	GS02	Upper case greek letter Sigma
GF02	Upper case greek letter Phi	GS61	Lower case greek letter final Sigma
GG01	Lower case greek letter Gamma	GT01	Lower case greek letter Tau
GG02	Upper case greek letter Gamma	GT02	Upper case greek letter Tau
GH01	Lower case greek letter Khi	GT61	Lower case greek letter Theta
GH02	Upper case greek letter Khi	GT62	Upper case greek letter Theta
GI01	Lower case greek letter Iota	GU01	Lower case greek letter Upsilon
GI02	Upper case greek letter Iota	GU02	Upper case greek letter Upsilon
GI11	Lower case greek letter Iota with accent	GU11	Lower case greek letter Upsilon with accent
GI12	Upper case greek letter Iota with accent	GU12	Upper case greek letter Upsilon with accent
GI17	Lower case greek letter Iota with diaeresis	GU17	Lower case Upsilon with diaeresis
GI18	Upper case greek letter Iota with diaeresis	GU18	Upper case Upsilon with diaeresis
GI33	Lower case greek letter Iota with accent and diaeresis	GU33	Lower case Upsilon with accent and diaeresis
GK01	Lower case greek letter Kappa	GX01	Lower case greek letter Xi
GK02	Upper case greek letter Kappa	GX02	Upper case greek letter Xi
GL01	Lower case greek letter Lambda	GZ01	Lower case greek letter Zeta
GL02	Upper case greek letter Lambda	GZ02	Upper case greek letter Zeta
GM01	Lower case greek letter Mu	SD33	Diaeresis and accent sign with space
GM02	Upper case greek letter Mu		
GN01	Lower case greek letter Nu		

I.6 Repertoire 6 – Cyrillic characters

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
KA01	Small cyrillic a	KC61	Small cyrillic chtcha
KA02	Capital cyrillic a	KC62	Capital cyrillic chtcha
KA61	Small cyrillic ja	KD01	Small cyrillic de
KA62	Capital cyrillic ja	KD02	Capital cyrillic de
KB01	Small cyrillic be	KE01	Small cyrillic je
KB02	Capital cyrillic be	KE02	Capital cyrillic je
KC01	Small cyrillic tse	KE17	Small cyrillic jo
KC02	Capital cyrillic tse	KE18	Capital cyrillic jo
KC21	Small cyrillic tche	KE61	Small cyrillic e
KC22	Capital cyrillic tche	KE62	Capital cyrillic e

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
KF01	Small cyrillic eff	KS02	Capital cyrillic es
KF02	Capital cyrillic eff	KS23	Small cyrillic sha
KG01	Small cyrillic gue	KS24	Capital cyrillic sha
KG02	Capital cyrillic gue	KT01	Small cyrillic te
KH01	Small cyrillic kha	KT02	Capital cyrillic te
KH02	Capital cyrillic kha	KU01	Small cyrillic v
KI01	Small cyrillic i	KU02	Capital cyrillic v
KI02	Capital cyrillic i	KV61	Small cyrillic ju
KI23	Small cyrillic breve i	KV62	Capital cyrillic ju
KI24	Capital cyrillic breve i	KV01	Small cyrillic ve
KJ01	Small cyrillic zhe	KV02	Capital cyrillic ve
KJ02	Capital cyrillic zhe	KY01	Small cyrillic ieri
KL01	Small cyrillic el	KY02	Capital cyrillic ieri
KL02	Capital cyrillic el	KY61	Small cyrillic ier (miagkil zhak)
KM01	Small cyrillic em	KY62	Capital cyrillic ier (miagkil zhak)
KM02	Capital cyrillic em	KY63	Small cyrillic ier' (tvjordy zhak)
KN01	Small cyrillic en	KY64	Capital cyrillic ier' (tvjordy zhak)
KN02	Capital cyrillic en	KZ01	Small cyrillic ze
KO01	Small cyrillic o	KZ02	Capital cyrillic ze
KO02	Capital cyrillic o	KK01	Small cyrillic ka
KP01	Small cyrillic Pe	KK02	Capital cyrillic ka
KP02	Capital cyrillic Pe		
KR01	Small cyrillic er		
KR02	Capital cyrillic er		
KS01	Small cyrillic es		

I.7 Repertoire 7 – Block mosaic characters

Block mosaic characters are a sub-Repertoire of the mosaics used in each of the terminal data syntaxes and therefore are directly translatable between data syntaxes. Even the coding of the block mosaic characters is identical except for the Filled Mosaic character. A Filled Mosaic is included in Repertoire 5. Either of the two equivalent codings established in Recommendation T.101 (code table position 5/15 or 7/15) for the Filled Mosaic character is acceptable in a string of characters from Repertoire 5.

NOTE – The following convention is used to describe the shape of a 2 sub-cell wide by 3 sub-cell high mosaic character. The cells are numbered from the upper left, to upper right, centre left, centre right, lower left, and lower right as 1,2,3,4,5,6 respectively. This code is used in identifying each character in the Repertoire.

1	2
3	4
5	6

T0821510-95/d009

Mosaic sub-cell structure

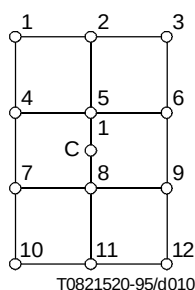
<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
MG00	Mosaic empty character	MG04	Block Mosaic 3
MG01	Block Mosaic 1	MG05	Block Mosaic 1,3
MG02	Block Mosaic 2	MG06	Block Mosaic 2,3
MG03	Block Mosaic 1,2	MG07	Block Mosaic 1,2,3

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
MG08	Block Mosaic 4	MG37	Block Mosaic 1,3,6
MG09	Block Mosaic 1,4	MG38	Block Mosaic 2,3,6
MG10	Block Mosaic 2,4	MG39	Block Mosaic 1,2,3,6
MG11	Block Mosaic 1,2,4	MG40	Block Mosaic 4,6
MG12	Block Mosaic 3,4	MG41	Block Mosaic 1,4,6
MG13	Block Mosaic 1,3,4	MG42	Block Mosaic 2,4,6
MG14	Block Mosaic 2,3,4	MG43	Block Mosaic 1,2,4,6
MG15	Block Mosaic 1,2,3,4	MG44	Block Mosaic 3,4,6
MG16	Block Mosaic 5	MG45	Block Mosaic 1,3,4,6
MG17	Block Mosaic 1,5	MG46	Block Mosaic 2,3,4,6
MG18	Block Mosaic 2,5	MG47	Block Mosaic 1,2,3,4,6
MG19	Block Mosaic 1,2,5	MG48	Block Mosaic 5,6
MG20	Block Mosaic 3,5	MG49	Block Mosaic 1,5,6
MG21	Block Mosaic 1,3,5	MG50	Block Mosaic 2,5,6
MG22	Block Mosaic 2,3,5	MG51	Block Mosaic 1,2,5,6
MG23	Block Mosaic 1,2,3,5	MG52	Block Mosaic 3,5,6
MG24	Block Mosaic 4,5	MG53	Block Mosaic 1,3,5,6
MG25	Block Mosaic 1,4,5	MG54	Block Mosaic 2,3,5,6
MG26	Block Mosaic 2,4,5	MG55	Block Mosaic 1,2,3,5,6
MG27	Block Mosaic 1,2,4,5	MG56	Block Mosaic 4,5,6
MG28	Block Mosaic 3,4,5	MG57	Block Mosaic 1,4,5,6
MG29	Block Mosaic 1,3,4,5	MG58	Block Mosaic 2,4,5,6
MG30	Block Mosaic 2,3,4,5	MG59	Block Mosaic 1,2,4,5,6
MG31	Block Mosaic 1,2,3,4,5	MG60	Block Mosaic 3,4,5,6
MG32	Block Mosaic 6	MG61	Block Mosaic 1,3,4,5,6
MG33	Block Mosaic 1,6	MG62	Block Mosaic 2,3,4,5,6
MG34	Block Mosaic 2,6	MG63	Filled Mosaic 1,2,3,4,5,6 (also equivalent to MG64-T.101 DSIII)
MG35	Block Mosaic 1,2,6		
MG36	Block Mosaic 3,6		

I.8 Repertoire 8 – Sub cell aligned smooth mosaics 1

A general set of Sub Cell Aligned Smooth Mosaic characters are part of the Repertoire of mosaic characters used in two of the terminal data syntaxes, and have the same coding. These characters are identified separately as a Repertoire of the interworking data syntax since these characters are directly translatable between two of the data syntaxes. The presentation of these characters must be converted so that their presentation effect may be achieved in a terminal using a data syntax which does not support these characters.

NOTE – The following convention is used to describe the shape of a 2 sub-cell wide by 3 sub-cell high mosaic character containing Sub Cell Aligned Smooth Mosaics. The vertices of the sub-cells are numbered from the upper sub-cell upper left corner as illustrated below. The smooth mosaic is filled below (B), above (A) to the right (R) or the left (L) of a line or lines through the indicated vertices.



Mosaic sub-cell vertices

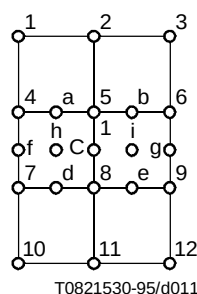
<i>Name code</i>	<i>Descriptive name</i>		
SG01	Sub Cell Aligned Smooth Mosaic B-7,11	SG29	Sub Cell Aligned Smooth Mosaic A-7,11
SG02	Sub Cell Aligned Smooth Mosaic B-7,12	SG30	Sub Cell Aligned Smooth Mosaic A-7,12
SG03	Sub Cell Aligned Smooth Mosaic B-4,11	SG31	Sub Cell Aligned Smooth Mosaic A-4,11
SG04	Sub Cell Aligned Smooth Mosaic B-4,12	SG32	Sub Cell Aligned Smooth Mosaic A-4,12
SG05	Sub Cell Aligned Smooth Mosaic B-1,11	SG33	Sub Cell Aligned Smooth Mosaic A-1,11
SG06	Sub Cell Aligned Smooth Mosaic B-1,12	SG34	Sub Cell Aligned Smooth Mosaic A-1,12
SG07	Sub Cell Aligned Smooth Mosaic B-4,2	SG35	Sub Cell Aligned Smooth Mosaic A-4,2
SG08	Sub Cell Aligned Smooth Mosaic B-4,3	SG36	Sub Cell Aligned Smooth Mosaic A-4,3
SG09	Sub Cell Aligned Smooth Mosaic B-7,2	SG37	Sub Cell Aligned Smooth Mosaic A-7,2
SG10	Sub Cell Aligned Smooth Mosaic B-7,3	SG38	Sub Cell Aligned Smooth Mosaic A-7,3
SG11	Sub Cell Aligned Smooth Mosaic B-10,2	SG39	Sub Cell Aligned Smooth Mosaic A-10,2
SG12	Sub Cell Aligned Smooth Mosaic B-7,6	SG40	Sub Cell Aligned Smooth Mosaic A-7,6
SG13	Sub Cell Aligned Smooth Mosaic R-1,c,10	SG41	Sub Cell Aligned Smooth Mosaic L-1,c,10
SG14	Sub Cell Aligned Smooth Mosaic B-1,c,3	SG42	Sub Cell Aligned Smooth Mosaic A-1,c,3
SG15	Sub Cell Aligned Smooth Mosaic A-10,c,12	SG43	Sub Cell Aligned Smooth Mosaic A-11,9
SG16	Sub Cell Aligned Smooth Mosaic L-3,c,12	SG44	Sub Cell Aligned Smooth Mosaic A-10,9
SG17	Sub Cell Aligned Smooth Mosaic B-4,9	SG45	Sub Cell Aligned Smooth Mosaic A-11,6
SG18	Sub Cell Aligned Smooth Mosaic B-2,12	SG46	Sub Cell Aligned Smooth Mosaic A-10,6
SG19	Sub Cell Aligned Smooth Mosaic B-1,9	SG47	Sub Cell Aligned Smooth Mosaic A-11,3
SG20	Sub Cell Aligned Smooth Mosaic B-2,9	SG48	Sub Cell Aligned Smooth Mosaic A-10,3
SG21	Sub Cell Aligned Smooth Mosaic B-1,6	SG49	Sub Cell Aligned Smooth Mosaic A-2,6
SG22	Sub Cell Aligned Smooth Mosaic B-2,6	SG50	Sub Cell Aligned Smooth Mosaic A-1,6
SG23	Sub Cell Aligned Smooth Mosaic B-10,3	SG51	Sub Cell Aligned Smooth Mosaic A-2,9
SG24	Sub Cell Aligned Smooth Mosaic B-11,3	SG52	Sub Cell Aligned Smooth Mosaic A-1,9
SG25	Sub Cell Aligned Smooth Mosaic B-10,6	SG53	Sub Cell Aligned Smooth Mosaic A-2,12
SG26	Sub Cell Aligned Smooth Mosaic B-11,6	SG54	Sub Cell Aligned Smooth Mosaic A-4,9
SG27	Sub Cell Aligned Smooth Mosaic B-10,9	SG55	Sub Cell Aligned Smooth Mosaic R-3,c,12
SG28	Sub Cell Aligned Smooth Mosaic B-11,9	SG56	Sub Cell Aligned Smooth Mosaic B-10,c,12

*Name
code* *Descriptive name*

I.9 Repertoire 9 – General smooth mosaics

An additional form of Smooth Mosaic characters is available in only one of the terminal data syntaxes. These smooth mosaic characters align with half sub-cell boundaries. Since these characters are unique to only one of the terminal data syntaxes, the presentation of these characters must be converted so that their presentation may be achieved in another terminal data syntax.

NOTE – The notation used to describe these characters is the same as that used in Repertoire 6 except that intermediate points which identify to the half sub cell are introduced. In the case that four numbers are given, such as for the characters (SG58 and SG71), the area is bounded on all sides by the four sub-cell positions. In two special cases, for the characters (SG68 and SG81), two areas are filled within the mosaic cell.



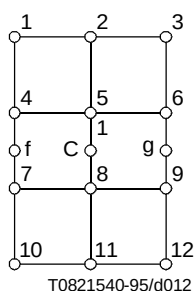
**Mosaic sub-cell vertices
and intermediate points**

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
MS01	General Smooth Mosaic B-f,c	MS17	General Smooth Mosaic B-c,g
MS03	General Smooth Mosaic B-h,i	MS18	General Smooth Mosaic A-e,d & B-a,d
MS04	General Smooth Mosaic B-F,C,2	MS19	General Smooth Mosaic A-h,i
MS05	General Smooth Mosaic A-f,11	MS20	General Smooth Mosaic L-2,c,g
MS06	General Smooth Mosaic A-1,g	MS21	General Smooth Mosaic A-11,g
MS07	General Smooth Mosaic L-1,g,10	MS22	General Smooth Mosaic A-f,3
MS08	General Smooth Mosaic B-f,12	MS23	General Smooth Mosaic R-3,f,12
MS09	General Smooth Mosaic A-f,11,g	MS24	General Smooth Mosaic B-10,g
MS10	General Smooth Mosaic L-3,f,12	MS25	General Smooth Mosaic B-f,2,g
MS11	General Smooth Mosaic A-f,2,g	MS26	General Smooth Mosaic R-1,g,10
MS12	General Smooth Mosaic L-1,c,10 & R-3,c,12	MS27	General Smooth Mosaic B-f,11,g
MS16	General Smooth Mosaic B-f,g	MS28	General Smooth Mosaic A-1,c,3 & B-10,c,12

I.10 Repertoire 10 – Drawing characters

A number of drawing characters, consisting primarily of line drawing characters, are available in the various terminal data syntaxes. These consist of the drawing characters (DG01 to DG04, DG13 to DG24, and DG32) which are common between DS I and DS II of this Recommendation, drawing characters (DG35 to DG50) which are unique to DS I, drawing characters (DG05 to DG12, and DG25 to DG31) which are unique to DS II and drawing characters (DG33 and DG34) which are unique to DS III.

NOTE 1 – The notation used to describe these characters identifies the character sub-cell vertices and intermediate points in a similar manner to the way smooth mosaics are identified in Repertoire 7; however, only lines are used here to connect the vertex and intermediate points. Some of the drawing graphics contain more than one line component. This is identified by the description of two line elements separated by “&”. In addition, the line weight (or width) may vary on some drawing characters. Heavy weight (wider) line elements are indicated by (W). Special drawing graphics are described in words.



**Character sub-cell vertices
and intermediate points**

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
DG01	Drawing Character W-f,g & c,2	DG27	Drawing Character 2,11 (with arrow head on top)
DG02	Drawing Character W-f,g & c,11	DG28	Drawing Character 2,11 (with arrow head on bottom)
DG03	Drawing Character 2,11 & W-c,g	DG29	Drawing Character (centre small dot)
DG04	Drawing Character 2,11 & W-f,c	DG30	Drawing Character (centre large dot)
DG05	Drawing Character 2,f,11	DG31	Drawing Character (centre large hollow dot)
DG06	Drawing Character 2,g,11	DG32	Drawing Character (speckled character)
DG07	Drawing Character f,11,g	DG33	Drawing Character 3,10
DG08	Drawing Character f,2,g	DG34	Drawing Character 1,12
DG09	Drawing Character f,2	DG35	Drawing Character W-2,11
DG10	Drawing Character g,2	DG36	Drawing Character W-f,g
DG11	Drawing Character f,11	DG37	Drawing Character W-11,c,g
DG12	Drawing Character g,11	DG38	Drawing Character W-f,c,11
DG13	Drawing Character W-f,g & 2,11	DG39	Drawing Character W-2,c,g
DG14	Drawing Character 2,11	DG40	Drawing Character W-f,c,2
DG15	Drawing Character f,g	DG41	Drawing Character W-2,11 & W-c,g
DG16	Drawing Character 11,c,g	DG42	Drawing Character W-2,11 & W-f,c
DG17	Drawing Character f,c,11	DG43	Drawing Character W-f,g & W-c,11
DG18	Drawing Character 2,c,g	DG44	Drawing Character W-f,g & W-c,2
DG19	Drawing Character f,c,2	DG45	Drawing Character W-f,g & W-2,11
DG20	Drawing Character 2,11 & c,g	DG46	Drawing Character W-2,11 & c,g
DG21	Drawing Character 2,11 & f,c	DG47	Drawing Character W-2,11 & f,c
DG22	Drawing Character f,g & c, 11	DG48	Drawing Character f,g & W-c,11
DG23	Drawing Character f,g & c,2	DG49	Drawing Character f,g & W-c,2
DG24	Drawing Character f,g & 2,11	DG50	Drawing Character W-2, 11 & f,g
DG25	Drawing Character f,g (with arrow head on right)		
DG26	Drawing Character f,g (with arrow head on left)		

NOTE 2 – The name codes DG33 to DG50 are introduced here since name codes for these characters are not included in either the ISO registry or in this Recommendation.

I.11 Repertoire 11 – Special mosaics

A number of special mosaic shapes exist in the various data syntaxes. These special mosaic shapes are generally unique and occur in only one of the terminal data syntaxes. The presentation of these characters must be converted so that their presentation effect may be achieved in another terminal data syntax.

<i>Name code</i>	<i>Descriptive name</i>	<i>Name code</i>	<i>Descriptive name</i>
MS02	Centre-Small-Square	MS29	Open-Right-Half-Oval
MS13	Open-Left-Half-Oval	MS30	Filled-Right-Half-Oval
MS14	Filled-Left-Half-Oval	MS31	Reverse-Right-Half-Oval
MS15	Reverse-Left-Half-Oval		

Appendix II

(to Recommendation T.101, Annex A)

Default states of the interworking data syntax

The Interworking Data Syntax permits a choice of the default states to be established. These defaults correspond to the default states of the terminal oriented data syntaxes corresponding to the source of the data. Referencing Tables II.1 to II.3 provides a convenient way of establishing the states required for interworking with a particular source of data. Only three default tables are described below. If a particular profile of a data syntax does not correspond exactly to the default table given below, then explicit state vector commands may be used to alter the particular states which are different.

The default tables include the default values for the function and parameters of the data syntax as well as the default boundary conditions. The Terminal Model Precedence indicator is always assumed to be initially “1”, and of course at the initial point, or at a reset, all global variables are assumed to be altered.

NOTE – In Table II.2 below a question mark “(?)” implies that the information for that entry in the table needs to be confirmed. A question mark alone indicates that the information needs to be supplied in a future version of this Recommendation.

Table II.1/T.101 – Default-set-I – Defaults corresponding to Data Syntax I

<i>Default boundary condition values</i>	
– Screen-Dimensions	– 0.969, 0.797
– Colour-Map-Limit	– 16
– Presentation-Sub-Area	– full screen
– Char-Mode-Constraints	– 15.5, 8
– Coordinate-Limit-Polygon	– 256 (fill form)
– Coordinate-Limit-Spline	– not applicable
– Presentation-Resolution	– 248, 204 (basic rank)
– Macro-Seg-Memory-Limit	– 3072 bytes (min)
– DRCS-Memory-Limit	– 416 bytes (min)
<i>Default presentation parameters</i>	
current-text-position	– upper left corner
current-foreground-colour	– white
current-auxiliary-colour	– blue reduced intensity
lining-state	– off
flash-blink-state	– off
basic-char-size-state	– char = normal 16/256, 24/256
colouring-block-state	– 4/256, 4/256
conceal-state	– off
char-invert-box-state	– off
char-marking-state	– not applicable
screen-protection-state	– protect
display-control-state	– scroll = off
device-control-state	– not applicable
cursor-control-state	– not applicable
current-geometric-position	– lower left corner

Table II.1/T.101 – Default-set-I – Defaults corresponding to Data Syntax I (cont.)

Default presentation parameters (cont.)					
geometric-control-1-state			– line texture = solid, texture pattern = solid highlight = off logical pel = 0,0		
geometric-control-2-state			– not applicable		
wait-state			– none		
general-text-state			– not applicable		
p-text-state			– char rotation = 0° char path = right		
char spacing = 1			cursor = underscore interrow = single space		
geometric-text-state			– not applicable		
DRCS-definition-state			– none		
macro-definition-state			– none		
texture-pattern-state			– not applicable		
music-part-memory-state			– musical tone = fixed – musical control = fixed part memory = none		
animation-configuration-state			– no animation frames frame order = normal		
workstation-configuration-state			– not applicable		
Default colour map – Data Syntax I					
	Map address		Colour value		
			R	G	B
black	0	0000	0001	0001	0001
red	1	0001	1111	0000	0000
green	2	0010	0000	1111	0000
yellow	3	0011	1111	1111	0000
blue	4	0100	0000	0000	1111
magenta	5	0101	1111	0000	1111
cyan	6	0110	0000	1111	1111
white	7	0111	1111	1111	1111
transparent	8	1000	0000	0000	0000
RI red	9	1001	0111	0000	0000
RI green	10	1010	0000	0111	0000
RI yellow	11	1011	0111	0111	0000
RI blue	12	1100	0000	0000	0111
RI magenta	13	1101	0111	0000	0111
RI cyan	14	1110	0000	0111	0111
grey	15	1111	0111	0111	0111

Table II.2/T.101 – Default-set-II – Defaults corresponding to Data Syntax II

<i>Default boundary condition values</i>	
– Screen-Dimensions	– 1, 1
– Colour-Map-Limit	– 32
– Presentation-Sub-Area	– not applicable
– Char-Mode-Constraints	– 40, 24
– Coordinate-Limit-Polygon	– 128
– Coordinate-Limit-Polycurve	– not applicable
– Presentation-Resolution	– (?)
– Macro-Seg-Memory-Limit	– (?)
– DRCS-Memory-Limit	– 2048 bytes
– Direct Colours	– 8 colours
<i>Default presentation parameters</i>	
current-text-position	– upper left corner
current-foreground-colour	– white
current-auxiliary-colour	– transparent
lining-state	– off
flash-blink-state	– off
basic-char-size-state	– 1/40, 5/128, 40 char/24 rows
conceal-state	– off
char-invert-box-state	– off
char-marking-state	– off
screen-protection-state	– not protected
display-control-state	– implicit scrolling activated. No defined scrolling area.
device-control-state	– display device = on auxiliary device = off recording device = stop hard copy device = stop
cursor-control-state	– off
geometric-control-1-state	– not applicable
geometric-control-2-state	0.0, 0.0 and 1.0, 1.0
– Clipping-Rectangle	asap
– deferral-mode	1
– Fill-Area-Colour-Index	hollow
– Fill-Area-Interior-Style	1
– Fill-Area-Style-Index	off
– Set-Highlighting	allowed
– implicit-regeneration	solid
– Line-Type	1.0
– Line-Width-Scale-Factor	1.0
– Marker-Size-Scale-Factor	asterisk
– Marker-Type	0.0, 1.0 and 1.0, 0.0
– Pattern-Vectors	1
– Polyline-Colour-Index	1
– Polymarker-Colour-Index	postpone
– regeneration-flag	the largest square which fits into the display area
– Ws-Viewport	0.0, 0.0 and 1.0, 1.0
– Ws-Window	0
– workstation-identifier	
geometric-text-state	
– Char-Expansion-Factor	1.0
– Char-Spacing(continous)	0.0
– Char-Rotation(continous)	0.0, 7.0/320.0 and 7.0/320.0, 0.0
– Text-Alignment	normal, normal
– Text-Colour-Index	1
– Text-Precision	string
– Char-Path	char-path-right
Fill-Pattern-Control	
– Pattern-Representation	
delta-x	1
delta-y	1
pattern-cell-data	1
pattern-index	1
DRCS-definition-state	– none defined
macro-definition-state	– not applicable
texture-pattern-state	– ?
music-part-memory-state	– not applicable
animation-configuration-state	– not applicable
Segment-Control-State	
– Segment-Priority	0.0
– Segment-Transform-Matrix	1.0; 0.0; 0.0
– segment-visibility	visible

Table II.2/T.101 – Default-set-II – Defaults corresponding to Data Syntax II (cont.)

<i>Default colour map – Data Syntax II</i>					
	Map address		Colour value		
			R	G	B
black	0	00000	000000	000000	000000
red	1	00001	111111	000000	000000
green	2	00010	000000	111111	000000
yellow	3	00011	111111	111111	000000
blue	4	00100	000000	000000	111111
magenta	5	00101	111111	000000	111111
cyan	6	00110	000000	111111	111111
white	7	00111	111111	111111	111111
transparent	8	01000	---	---	---
RI red	9	01001	011111	000000	000000
RI green	10	01010	000000	011111	000000
RI yellow	11	01011	011111	011111	000000
RI blue	12	01100	000000	000000	011111
RI magenta	13	01101	011111	000000	011111
RI cyan	14	01110	000000	011111	011111
grey	15	01111	011111	011111	011111
black	16	10000	000000	000000	000000
red	17	10001	111111	000000	000000
green	18	10010	000000	111111	000000
yellow	19	10011	111111	111111	000000
blue	20	10100	000000	000000	111111
magenta	21	10101	111111	000000	111111
cyan	22	10110	000000	111111	111111
white	23	10111	111111	111111	111111
black	24	11000	000000	000000	000000
red	25	11001	111111	000000	000000
green	26	11010	000000	111111	000000
yellow	27	11011	111111	111111	000000
blue	28	11100	000000	000000	111111
magenta	29	11101	111111	000000	111111
cyan	30	11110	000000	111111	111111
white	31	11111	111111	111111	111111

Table II.3/T.101 – Default-set-III – Defaults corresponding to Data Syntax III

Default boundary condition values					
– Screen-Dimensions		– 0.9999, 0.78125			
– Colour-Map-Limit		– 16			
– Presentation-Sub-Area		– full screen			
– Char-Mode-Constraints		– 40, 20			
– Coordinate-Limit-Polygon		– 256			
– Coordinate-Limit-Polycurve		– 256			
– Presentation-Resolution		– 256, 200 (nominal)			
– Macro-Seg-Memory-Limit		– 3072 bytes			
– DRCS-Memory-Limit		– 3072 bytes (shared with Macro)			
Default presentation parameters					
current-text-position		– lower left corner			
current-foreground-colour		– colour = white			
		– mode = direct			
current-auxiliary-colour		– none			
lining-state		– off			
flash-blink-state		– off			
basic-char-size-state		– dx = 1/40, dy = 1/128			
conceal-state		– not applicable			
char-invert-box-state		– char = normal size, invert = off, box (transparent)=off			
char-marking-state		– not applicable			
screen-protection-state		– protected			
display-control-state		– scroll = off			
device-control-state		– not applicable			
cursor-control-state		– off (invisible)			
geometric-control-1-state		– line texture = solid texture pattern = solid texture mask = 1/40, 5/128 highlight = off logical pel = 0,0			
geometric-control-2-state		– not applicable			
wait-state		– none			
general-text-state		– char rotation = 0° char path = right cursor = underscore interrow = single space			
char spacing = 1					
p-text-state		– not applicable			
geometric-text-state		– not applicable			
DRCS-definition-state		– none defined			
macro-definition-state		– none defined			
texture-pattern-state		– none defined			
music-part-memory-state		– not applicable			
animation-configuration-state		– not applicable			
workstation-configuration-state		– not applicable			
Default colour map – Data Syntax III					
	Map address		Colour value		
			R	G	B
nominal black	0	0000	000	000	000
	1	0001	001	001	001
	2	0010	010	010	010
grey	3	0011	011	011	011
	4	0100	100	100	100
	5	0101	101	101	101
nominal white	6	0110	110	110	110
	7	0111	111	111	111
	8	1000	000	000	111
hues	9	1001	000	101	111
	10	1010	000	111	100
	11	1011	010	111	000
	12	1100	111	111	000
	13	1101	111	010	000
	14	1110	111	000	100
	15	1111	101	000	111

Annex B

Data Syntax I for international interactive Videotex service

(This annex forms an integral part of this Recommendation)

NOTE – This data syntax generally corresponds to the “CAPTAIN” presentation layer data syntax officially adopted by Japan.

Preface

This annex describes the formats, rules and procedures for encoding of textual, pictorial and musical information for videotex applications. This annex conforms to the architecture defined in ISO’s and CCITT’s multi-layered reference model of open systems interconnection as part of a presentation level.

Operation is accommodated in both a 7-bit and an 8-bit environment. For textual information, alphanumerics, Japanese characters and the Dynamically Redefinable Character Sets (DRCs for both one byte and two bytes) are provided. For pictorial information, the photographic coding scheme, which is a distinctive feature of this data syntax, is provided. The mosaic coding and geometric coding are also available for pictorial information. There are two mosaic sets: one mosaic set is compatible with that of Data Syntax II. The geometric coding scheme conforms to Data Syntax III. Additional capabilities include colour look-up table, macros, flashing, unprotected fields, concealed characters, variable character size and scrolling. Musical note encoding and animated picture coding are optionally available.

B.1 General

B.1.1 Scope

This annex describes the Data Syntax I, which provides means for exchanging textual, pictorial and audio information in the international videotex service interworking.

B.1.2 References

This annex is intended to be as closely compatible as possible with the following Recommendations and Standards:

- CCITT Recommendation T.50, *International Alphabet No. 5*.
- CCITT Recommendation T.51, *Coded character sets for telematic services*.
- ITU-T Recommendation T.52, *Non-Latin coded character sets for telematic services*.
- CCITT Recommendation F.300, *Videotex service*.
- ISO Standard 2022-1986, *ISO 7-bit 8-bit coded character sets-code extension techniques*.
- ISO Standard 6937, *Information processing – Coded character sets for text communication*.

B.1.3 Definitions

The following definitions apply:

attribute means a settable parameter to be applied to subsequent textual, pictorial or audio information.

bit combination is an ordered set of bits (binary digits) that represents a character or a control function.

C-set stands for control set. There are two control sets, C0 and C1, each of which comprises 32 character positions arranged in 2 columns by 16 rows.

character block means the rectangular area within which a normal size character is displayed.

character code mode is the mode in which the coding structure is based on the code extension technique of ISO 2022.

character code mode command means a character or a command such as one in the Display Control Commands which is encoded in the character code mode.

character field means the rectangular area within which a character is displayed in the currently defined character size.

code extension means techniques for expanding the absolute character address space of a byte-oriented code into a larger virtual address space.

code table means the set of unambiguous rules that defines the mapping between received bit combinations and presentation level characters.

defined display area means the addressable area of the physical display screen onto which the unit screen or a portion of the unit screen is mapped. Header area is not included in this area.

designate means to identify a given set from the repertory of G-sets as a G0, G1, G2 or G3 set.

display control command set is a G-set. A control command is composed of an opcode followed by zero or more operands and defines an attribute control function or a multi-frame structure.

dynamically redefinable character set (DRCS) is a G-set containing definable characters whose patterns can be downloaded from the host.

escape sequence means a string of two or more bit combinations beginning with the ESC character. Formats and rules regarding the use of the escape sequences are specified in ISO 2022.

frame is the minimum unit of the display structure on which complete information, but not necessarily whole information, can be displayed. Entire information on the screen may be composed of a few frames. Typical implementation of a frame requires a few memory planes.

G-set refers to one of the four sets, G0, G1, G2 and G3, each of which comprises 94 or 96 character positions arranged in 6 columns by 16 rows.

header area is used for the display of system messages and key pad input monitor.

Japanese-Kanji character set is a 2-byte G-set. The total of 7046 characters including Japanese-Kanji characters, Japanese phonetic signs (Katakana and Hiragana), Roman characters, numerics and additional characters are defined.

Katakana character set is a G-set which defines 63 Japanese phonetic signs.

layer is terminology adopted by ISO to describe each individual module of the reference model for open systems interconnection (OSI). (The terms “level” and “layer” are used interchangeably.)

locking shift means an invocation of a code set into the in-use table that remains in effect until another code set is invoked in its place.

mosaic is a rectangular matrix of pre-defined elements that can be used to construct block-style graphic images.

move instruction (MVI) is composed of an opcode followed by zero or more operands and constitutes an executable frame moving or control command.

message mode is the mode in which message information from the network or each center is carried. In this mode the coding structure is based on the code extension technique of ISO 2022.

photographic data unit (PDU) is composed of an opcode with a length indicator (LI), followed by zero, one or more operands, each of which consists of one or more octets of bit sequences. This use of all possible octet patterns are allowed in the operands bit sequences, which results in efficient expression or arbitrary data. The opcode consists of a single octet which indicates the meaning of photographic data contained in the PDU. The LI consists of one or more octets. The value of an LI is a binary number that represents the total length of operands following LI field in octet. One or more octets of parameters are located at the leading part of an operand field. Parameters include the drawing point coordinates where the photographic data should be displayed, and/or the packing format which indicates the way in which photographic data are arranged. Photographic data expressed on a dot-by-dot basis are contained in the remaining part of the operand field.

picture description instruction (PDI) is composed of an opcode followed by zero or more operands and constitutes an executable picture drawing or control command.

plane; memory plane is a component part of a frame, accommodating an information component such as pattern information, foreground colour information, etc.

primary character set is a G-set which defines 52 Roman characters, 10 numerics and 32 marks.

protocol is a set of formats, rules and procedures governing the exchange of information between peer processes at the same level.

service reference model (SRM) defines the recommended features that should be implemented by an ordinary terminal or decoder.

single shift is an invocation of a code set into the in-use table that affects only the interpretation of the next bit combination received. Interpretation then automatically reverts to the previous contents of the table. (This is also referred to as non-locking shift.)

sound control set is a C-set. The control codes are used for controlling sound generation.

sound mode is the mode in which sound information is carried. Sound tones are encoded within the character coding techniques.

sound tone set is a 2-byte G-set which defines pitch and duration of a sound.

terminal equipment is equipment that can exchange coded bit combinations by means of telecommunication or by physical interchange of storage media.

transparent mode is the mode in which all the presentation level bits are used for pictorial, audio or telesoftware data. This mode provides an efficient means of transmitting the relatively large amount of data.

transparent mode command means a command encoded in the transparent mode.

unit screen means the logical display address space within which all drawing operations are executed and alphanumeric characters are deposited. The dimensions of the unit screen are 0 to 1 in the horizontal (X), vertical (Y) and depth (Z) dimensions. (The last is only defined in three-dimensional mode.)

visible display area means the entire physical display screen visible to the user.

B.1.4 Visual information display

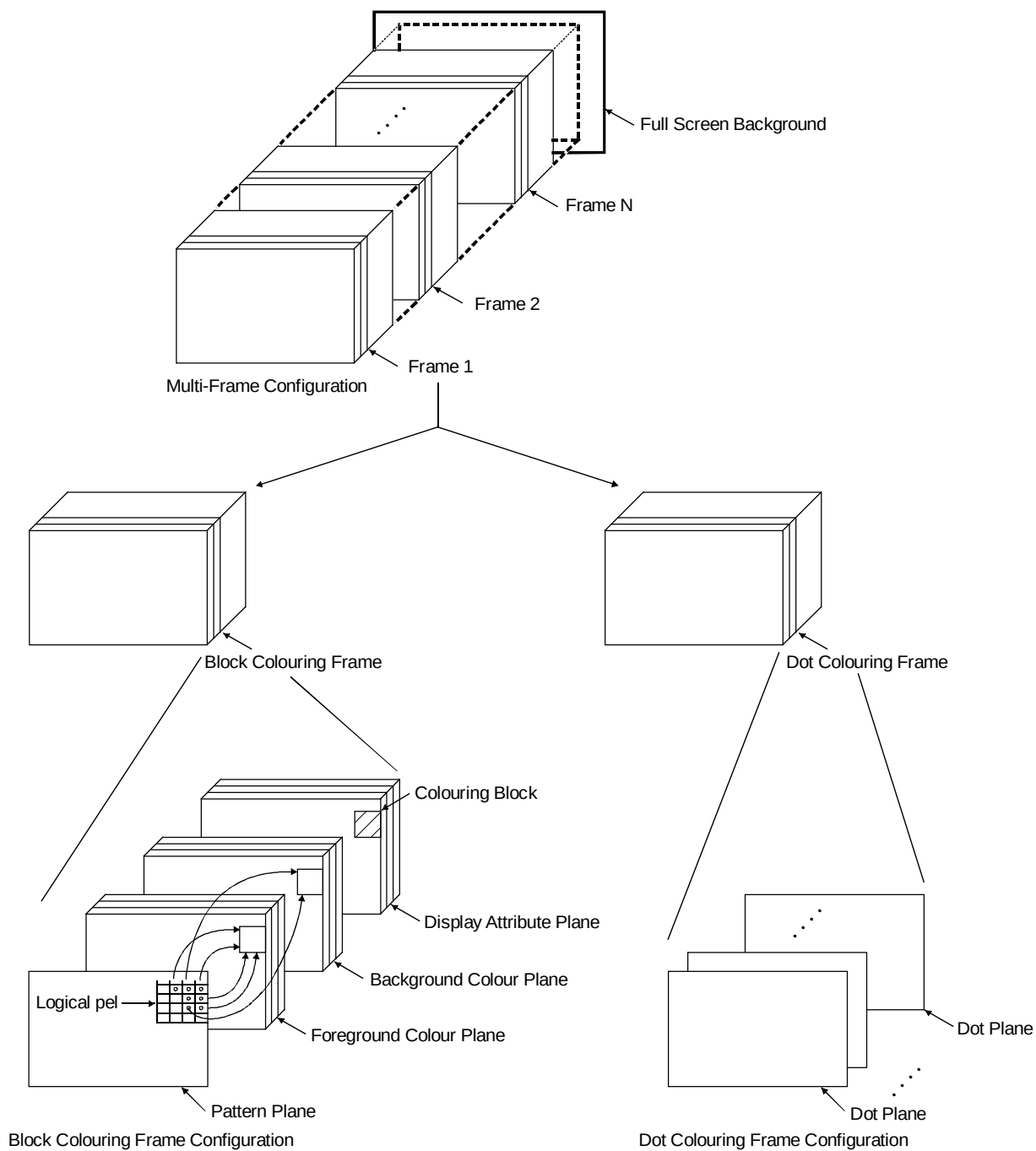
B.1.4.1 Display structure

This annex allows the multi-frame structure which provides superimposition of several frames. Figure B.1-1 illustrates an example of multi-frame structure.

Each frame can be of different constructions: a dot colouring frame has colour information on a dot-by-dot basis, while a block colouring frame has one on a block-by-block basis.

The full screen background layer has the lowest display priority: the Raster colour and the Header Raster colour are displayed only where an upper frame is “transparent” and outside of the defined display area.

Multi-frame manipulation is achieved through some display control commands. Details are given in B.3.3.3.



T0809810-92/d013

Figure B.1-1/T.101 – Multi-Frame Structure Concept

B.1.4.2 Logical frame structure

B.1.4.2.1 Block colouring frame

A block colouring frame is composed of a pattern plane, a foreground colour (FG) plane, a background colour (BG) plane and a display attribute (DA) plane as illustrated in Figure B.1-1.

Colour information in the FG plane and the BG plane is given on a colouring block basis. The colouring block is a rectangular area, normally larger than the pattern plane's pixel, and is a minimum unit of colour definition.

The pattern plane is a one-bit-deep memory plane. Where the bit is '1', the foreground colour is displayed. Where the bit is '0', the background colour is displayed.

A display attribute plane has flags for flashing, conceal and protect attributes on a colouring block basis.¹⁾

The block colouring frame structure economizes both data transmission time and the amount of a raster memory at the expense of colour resolution. Text and pictures are normally drawn on this type of frame.

B.1.4.2.2 Dot colouring frame

A dot colouring frame can store colour information on a pixel basis. Geometrically coded pictures are best reproduced on a dot colouring frame (see Figure B.1-1).

B.1.4.3 Coordinate system

The coordinate system utilized in both photographic and geometric picture data is based on the unit screen concept. A logical pel or a geometric primitive is always located within the unit screen, whose horizontal (X) and vertical (Y) coordinates range from 0 to 1. The origin (0, 0) of the unit screen is mapped to the lower left corner of the physical display screen.

The physical display screen should cover at least the rectangle region $0 \leq x \leq 0.969$, $0 \leq y \leq 0.797$. Normal information is displayed within the region $0 \leq y \leq 0.75$, while the remainder of the screen ($0.75 \leq y \leq 0.797$: Header) is used for the display of system messages and key pad input monitor.

The full screen background layer is divided into two regions: Raster ($y \leq 0.75$) and Header Raster ($0.75 \leq y$).

The concept of the unit screen and full screen background layer is shown in Figures B.1-2 and B.1-3.

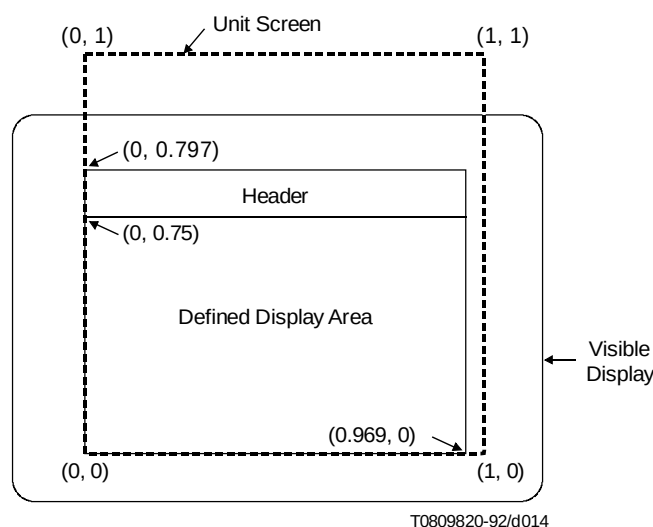


Figure B.1-2/T.101 – Unit Screen Concept

¹⁾ *Flashing*: Patterns are displayed alternately in the foreground colour and the background colour.

Conceal: Patterns are made invisible until the conceal attribute is cancelled.

Protect: A character block or a colouring block is protected against alternation by a user.

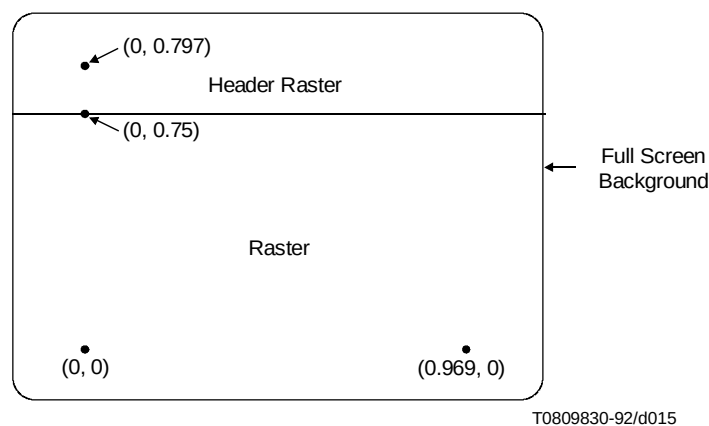


Figure B.1-3/T.101 – Full Screen Background Structure

B.1.5 Sound information

Sound information is encoded into pseudo character codes and is downloaded to a terminal. Then a multi-channel tone generator installed in the terminal can play music.

Mode switching technology is employed to present sound information (see B.2.2).

B.2 Coding architecture

B.2.1 General

Textual information is character coded and geometric primitives, musical notes and display control commands are encoded in the same scheme as character codes. Both 7-bit and 8-bit environments are supported.

B.2.2 Mode switching technique

This annex specifies four separate operation modes:

- the character code mode;
- the transparent mode;
- the sound mode; and
- the message mode.

The character code mode is the mode in which the coding structure is based on the code extension technique of ISO 2022.

In the transparent mode, transparent data can be conveyed.

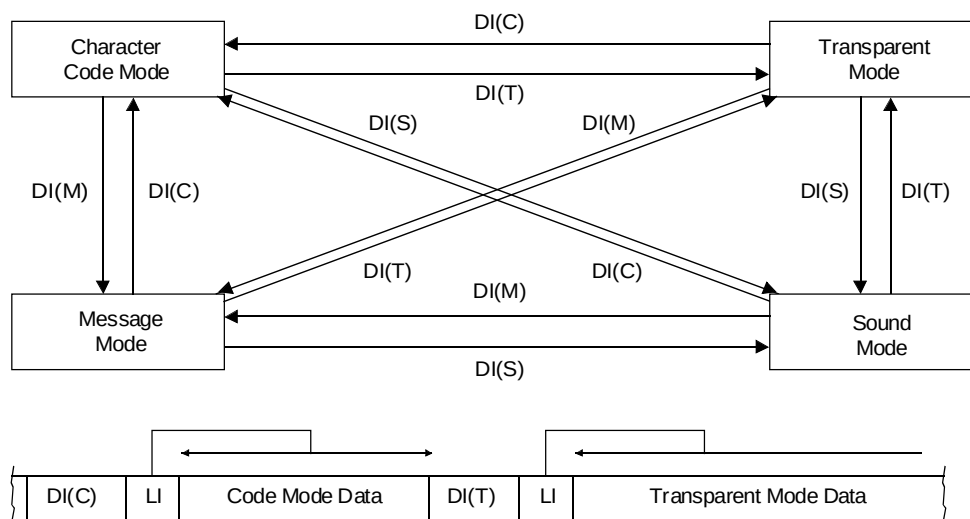
In the sound mode, musical sound notes are encoded and transmitted with the character coding technique; however, a C-set and a G-set are different from those in the character code mode.

In the message mode, the received data is executed prior to the data which is received in the other mode.

Each mode is invoked by the Data Identifier DI and the Length Indicator LI.

LI indicates the number of octets in a mode. If the decimal expression of the LI is 255, then the next two octets (the third and the fourth) indicate the real length.

Figure B.2-1 illustrates the mode switching concept.



Data Identifier of Character Code Mode	DI(C)	0 0 1 0 0 0 0 0
Data Identifier of Transparent Mode	DI(T)	0 0 1 1 0 0 0 0
Data Identifier of Transparent Mode Continued ^{a)}	DI(T-CNT)	0 0 1 1 1 1 1 1
Data Identifier of Sound Mode	DI(S)	0 1 0 0 0 0 0 1
Data Identifier of Message Mode	DI(M)	0 1 0 0 0 0 0 0

a) DI(T-CNT) indicates the data is continued from the previous data which is received in the transparent mode. It is used for transmitting PDUs which are separated into several blocks.

Length Indicator LI	Length
0 0 0 0 0 0 0 0	0 Byte
0 0 0 0 0 0 0 1	1 Byte
0 0 0 0 0 0 1 0	2 Byte
⋮	⋮
1 1 1 1 1 1 1 0	254 Byte
(Extension)	
1 1 1 1 1 1 1 1 0	0 Byte
1 1 1 1 1 1 1 1 0 1	1 Byte
1 1 1 1 1 1 1 1 0 1 0	2 Byte
⋮	⋮
1 0	65, 534 Byte

T0809840-92/d016

Figure B.2-1/T.101 – Mode switching Concept

B.2.3 Coding structure in the character code mode

The coding structure in the character code mode is based on the code extension principles of ISO 2022 for both the 7-bit and 8-bit environments. Figure B.2-2 shows the code extension method of Data Syntax I. Escape sequences for designation of C-sets and G-sets are shown in Table B.2-1.

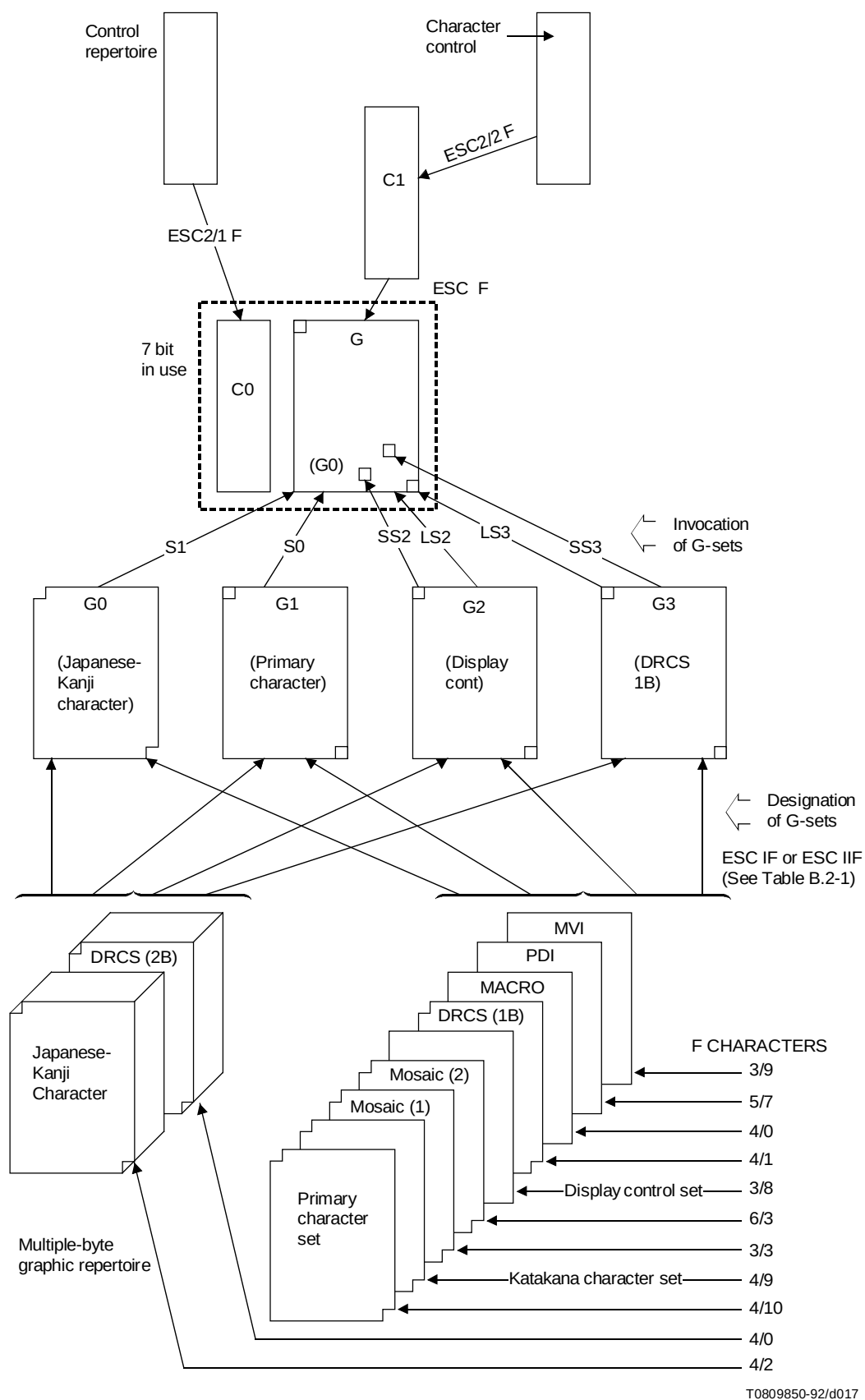


Figure B.2-2 (1)/T.101 – Code Extension in the Character Code Mode (in 7-bit environment)

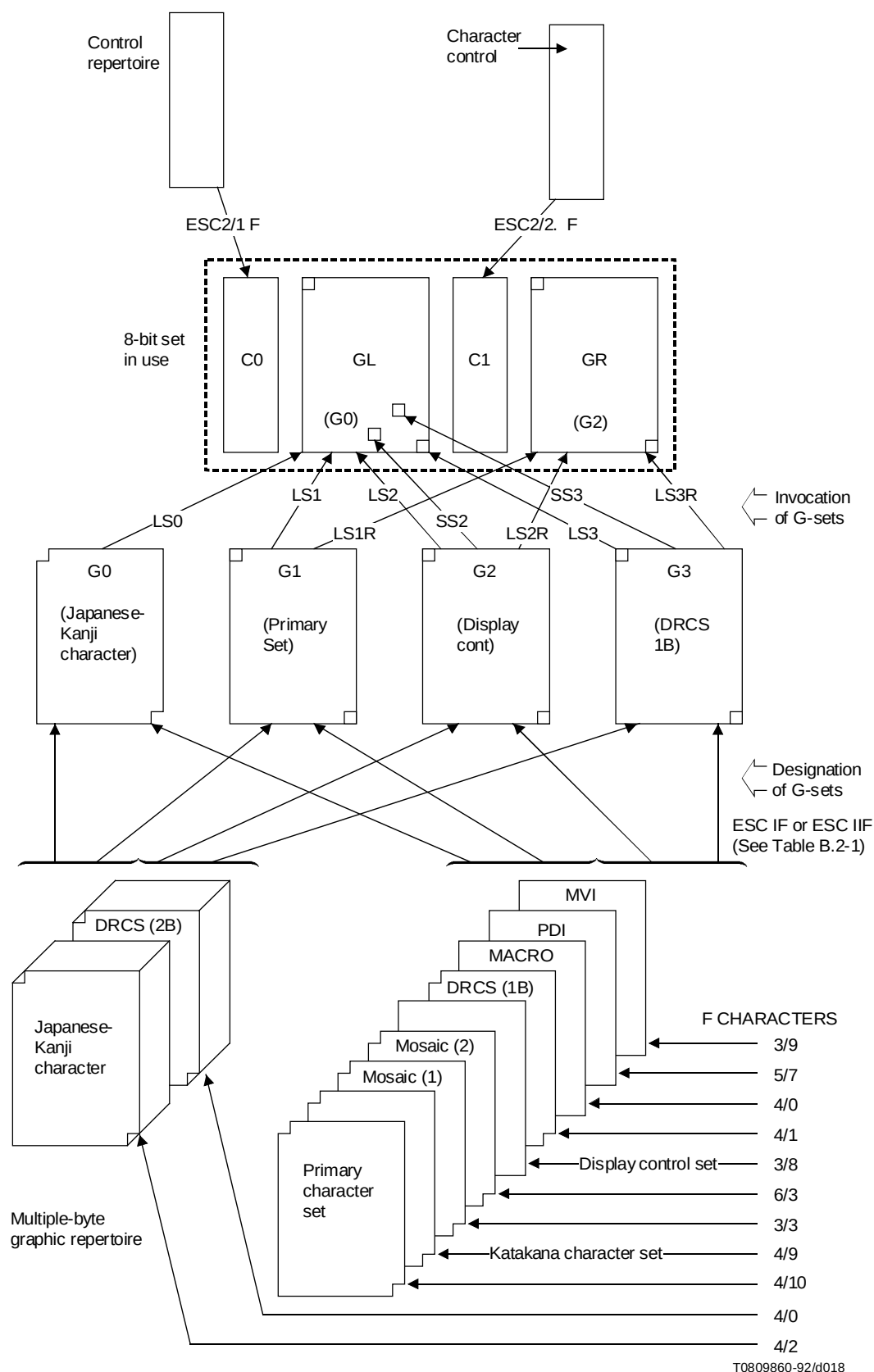


Figure B.2-2 (2)/T.101 – Code Extension in the Character Code Mode (in 8-bit environment)

Table B.2-1/T.101 – Escape Sequences for Designation of C-sets and G-sets

Set to be designated	Escape Sequence
C0 set	ESC 2/1 F
C1 set	ESC 2/2 F

Set to be designated	F
C0 set	4/9
C1 set	4/4

Character set	G0	G1	G2	G3
Primary Character set Katakana Character set Mosaic I set Mosaic II set	ESC 2/8 F	ESC 2/9 F	ESC 2/10 F	ESC 2/11 F
Display Control set PDI set MVI set	————	ESC 2/13 F	ESC 2/14 F	ESC 2/15 F
Kanji Character set	ESC 2/6 4/0 ESC 2/4 F	ESC 2/6 4/0 ESC 2/4 2/9 F	ESC 2/6 4/0 ESC 2/4 2/10 F	ESC 2/6 4/0 ESC 2/4 2/11 F
Macro set	————	ESC 2/13 2/0 F	ESC 2/14 2/0 F	ESC 2/15 2/0 F
DRCS I set	ESC 2/8 2/0 F	ESC 2/9 2/0 F	ESC 2/10 2/0 F	ESC 2/11 2/0 F
DRCS II set	ESC 2/4 2/8 2/0 F	ESC 2/4 2/9 2/0 F	ESC 2/4 2/10 2/0 F	ESC 2/4 2/11 2/0 F

Character set	F
Primary Character set Katakana Character set Mosaic I set Mosaic II set	4/10 4/9 3/3 6/3
Display Control set PDI set MVI set	3/8 5/7 3/9
Kanji Character set	4/2
Macro set	4/0
DRCS I set	4/1
DRCS II set	4/0

NOTE – The entire coding environment described in Data Syntax I is to be designated and invoked by the escape sequence ESC 2/5, 4/3, in accordance with ISO 2022-1986.

B.2.4 Coding structure in the transparent mode

In the transparent mode, transparent data are separated into Photographic Data Units (PDUs). The structure of a PDU is shown in Figure B.2-3. The first octet (8 bits) of a PDU is an opcode which defines the meaning of the PDU. The second octet is a length indicator which indicates the number of octets in a PDU. If the decimal expression of the second octet is 255, then the next two octets (the third and the fourth) indicate the real length. After the length indicator, the necessary bit data sequence follows.

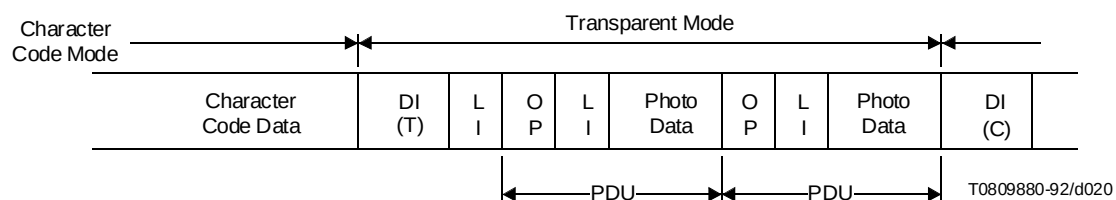


T0809870-92/d019

OP Opcode
LI Length Indicator

Figure B.2-3/T.101 – Photographic Data Unit Format

Data flow is illustrated in Figure B.2-4.



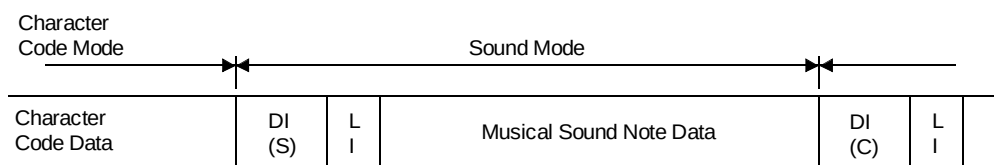
T0809880-92/d020

Figure B.2-4/T.101 – Data Flow (Transparent Mode)

B.2.5 Coding structure in the sound mode

Commands and parameters are character coded in the sound mode. However, a C-set and G-set different from the character code mode are set upon invocation of the mode.

Data flow is illustrated in Figure B.2-5.

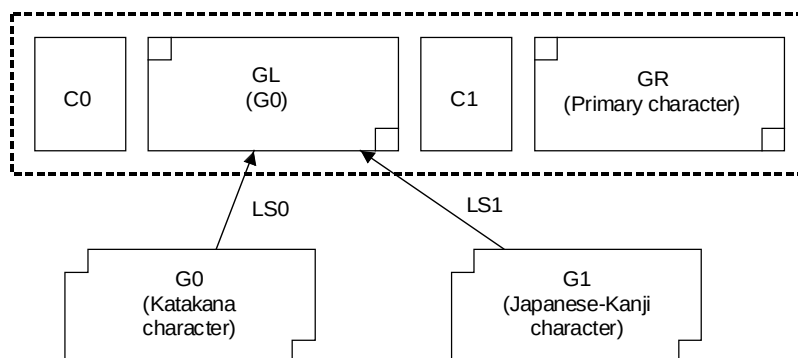


T0809890-92/d021

Figure B.2-5/T.101 – Data Flow (Sound Mode)

B.2.6 Coding structure in the message mode

The coding structure in the message mode is based on the code extension principles of ISO 2022. Figure B.2-6 shows the code extension method in the message mode.



T0809900-92/d022

Figure B.2-6/T.101 – Code extension in the Message Mode

B.3 Coding in the character code mode

B.3.1 C0 Control Set

The C0 control set is shown in Table B.3-1. The functions are as follows:

Table B.3-1 (1)/T.101 – C0 Control Set in the Character Code Mode

					b ₇	0	0
					b ₆	0	0
					b ₅	0	1
b ₄	b ₃	b ₂	b ₁		0		
0	0	0	0	0	NUL		
0	0	0	1	1			
0	0	1	0	2			
0	0	1	1	3			
0	1	0	0	4			KMC ^{a)}
0	1	0	1	5			
0	1	1	0	6			
0	1	1	1	7	BEL		
1	0	0	0	8	APB		
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	APS ^{a)}	
1	1	0	1	13	APR	SS3	
1	1	1	0	14	LS1 (SO)	APH	
1	1	1	1	15	LS0 (SI)	NSR ^{a)}	
^{a)} This code has parameters.							

Table B.3-1 (2)/T.101 – C0 Control Set in the Message Mode

					b ₇	0	0
					b ₆	0	0
					b ₅	0	1
b ₄	b ₃	b ₂	b ₁		0		
0	0	0	0	0	NUL		
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	BEL		
1	0	0	0	8	APB		
1	0	0	1	9	APF		
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	LS1 (SO)	APH	
1	1	1	1	15	LS0 (SI)	NSR ^{a)}	
^{a)} This code has parameters.							

Legend to Table B.3-1:

NUL	(0/0)	NULL character has no effect on syntax data. It may be used for timing adjustment.
BEL	(0/7)	BELL character is used to momentarily ring a bell for attracting the user's attention.
KMC	(1/4)	KMC character is used to conceal the Key-in-monitor area where input data is displayed. It is specified by the one byte parameter immediately following an KMC. KMC 4/0 – Key-in-monitor area is made unconcealed. KMC 4/1 – Key-in-monitor area is made concealed.
APB	(0/8)	ACTIVE POSITION BACKWARD is used to position the cursor a distance equal to the inter-character spacing lying parallel to the character path in the direction opposite to the character path. If such a movement would cause the edge of the defined display area to be crossed, then the cursor is positioned at the opposite edge of the active drawing area and an automatic APU is executed.
APF	(0/9)	ACTIVE POSITION FORWARD is used to position the cursor a distance equal to the inter-character spacing lying parallel to the character path in the direction of the character path. If such a movement would cause the edge of the defined display area to be crossed, then the cursor is positioned at the opposite edge of the defined display area and an automatic APD is executed.
APD	(0/10)	ACTIVE POSITION DOWN is used to position the cursor a distance equal to the inter-row spacing lying perpendicular to the character path in a direction perpendicular to the character path (-90°). If such a movement would cause the edge of the defined display area to be crossed, the cursor is positioned at the opposite edge of the defined display area.
APU	(0/11)	ACTIVE POSITION UP is used to position the cursor a distance equal to the inter-row spacing lying perpendicular to the character path in a direction perpendicular to the character path (90°). If such a movement would cause the edge of the defined display area to be crossed, then the cursor is positioned at the opposite edge of the defined display area.
APR	(0/13)	ACTIVE POSITION RETURN is used to position the cursor to the first character position within the defined display area along the character path and an automatic APD is executed.
APH	(1/14)	ACTIVE POSITION HOME is used to position the cursor to the upper left character position in the defined display area.
APS	(1/12)	ACTIVE POSITION SET is used to set the cursor position which is specified by the two bytes parameter immediately following an APS. The first byte represents the row address and the second byte does the column address. The address is obtained by taking the binary values comprising bits b_6 through b_1 with b_6 being the MSB. (Each byte is from columns 4 through 7.) Exact APS operations are as follows: First, an automatic APH is executed. Then, the cursor is positioned N times inter-row spacing below and M times inter-character spacing right of the home position, where N is the number specified by the first byte, and M is the number specified by the second byte. If any of the two bytes is out of columns 4 through 7, the APS character and the following two bytes are taken as NULs.
NSR	(1/15)	NSR serves two functions. First, NSR resets non-selectively all the display attributes to their default states (see clause 6). Then, the NSR character sets the cursor positions by the two bytes parameter immediately following NSR. The cursor positioning is the same as APS, except that APS sets the cursor according to the current inter-character spacing and inter-row spacing, while NSR sets the cursor according to the default values. Macro definitions and DRCS definitions are not cancelled by NSR.
CS	(0/12)	CS resets all the display attributes to their default states and activates an automatic APH. Then, all macro definitions and DRCS definitions are cancelled. CS clears the entire screen.
ESC	(1/11)	ESC introduces a code extension sequence.
LS0	SI (0/15)	Locking shift 0.
LS1	SI (0/14)	Locking shift 1.
SS2	(1/9)	Single shift 2.
SS3	(1/13)	Single shift 3.

B.3.2 C1 Control Set

The C1 control set is used to allow control over textual display attributes and also to define macros. Table B.3-2 shows the C1 control set table.

**Table B.3-2 (1)/T.101 – C1 Control Set
in the Character Code Mode**

				b ₈	1	1
				b ₇	0	0
				b ₆	0	0
				b ₅	0	1
					8	9
			b ₇	1	1	
			b ₆	0	0	
			b ₅	0	1	
b ₄	b ₃	b ₂	b ₁		4	5
0	0	0	0	0	BKF	COL ^{a)}
0	0	0	1	1	RDF	FLC ^{a)}
0	0	1	0	2	GRF	CDC ^{a)}
0	0	1	1	3	YLF	
0	1	0	0	4	BLF	
0	1	0	1	5	MGF	p ^{b)} _ MACRO ^{a)}
0	1	1	0	6	CNF	
0	1	1	1	7	WHF	
1	0	0	0	8	SSZ	RPC ^{a)}
1	0	0	1	9	MSZ	SPL
1	0	1	0	10	NSZ	STL
1	0	1	1	11	SZX	
1	1	0	0	12		
1	1	0	1	13		
1	1	1	0	14	CON	UNP
1	1	1	1	15	COF	PRT
a) This code has a parameter.						
b) P Photographic						

**Table B.3-2 (2)/T.101 – C1 Control Set
in the Message Mode**

				b ₈	1	1
				b ₇	0	0
				b ₆	0	0
				b ₅	0	1
					8	9
			b ₇	1	1	
			b ₆	0	0	
			b ₅	0	1	
b ₄	b ₃	b ₂	b ₁		4	5
0	0	0	0	0	BKF	COL ^{a)}
0	0	0	1	1	RDF	FLC ^{a)}
0	0	1	0	2	GRF	
0	0	1	1	3	YLF	
0	1	0	0	4	BLF	
0	1	0	1	5	MGF	
0	1	1	0	6	CNF	
0	1	1	1	7	WHF	
1	0	0	0	8		RPC ^{a)}
1	0	0	1	9		SPL
1	0	1	0	10		STL
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		
a) This code has a parameter.						

B.3.2.1 Colour controls

Each character on a screen has the foreground colour (pixels' colour where the value of the pattern plane is 1) and the background colour (pixels' colour where the value of the pattern plane is 0). Both the foreground colour and the background colour are specified by means of a colour look up table (LUT) entry address. However, a code name is derived from the default colour of the LUT.

The default colours include six colours (red, green, yellow, blue, magenta, cyan) and black and white, and their reduced intensity versions (see Table B.3-5).

A full intensity foreground colour is specified by one of the following C1 control codes.

- BKF (Black foreground): Invokes black or 0th colour of the LUT as a foreground colour.
- RDF (Red foreground): Invokes red or 1st colour of the LUT as a foreground colour.
- GRF (Green foreground): Invokes green or 2nd colour of the LUT as a foreground colour.
- YLF (Yellow foreground): Invokes yellow or 3rd colour of the LUT as a foreground colour.
- BLF (Blue foreground): Invokes blue or 4th colour of the LUT as a foreground colour.
- MGF (Magenta foreground): Invokes magenta or 5th colour of the LUT as a foreground colour.
- CNF (Cyan foreground): Invokes cyan or 6th colour of the LUT as a foreground colour.
- WHF (White foreground): Invokes white or 7th colour of the LUT as a foreground colour.

A full intensity background colour, a reduced intensity foreground colour or a reduced intensity background colour is specified by one byte parameter following the COL C1 control code.

- COL 4/8 – TRF (Transparent foreground): Invokes transparent or 8th colour of the LUT as a foreground colour.
- COL 4/9 – RDFR (Red foreground reduced): Invokes reduced intensity red or 9th colour of the LUT as a foreground colour.
- COL 4/10 – GRFR (Green foreground reduced): Invokes reduced intensity green or 10th colour of the LUT as a foreground colour.
- COL 4/11 – YLFR (Yellow foreground reduced): Invokes reduced intensity yellow or 11th colour of the LUT as a foreground colour.
- COL 4/12 – BLFR (Blue foreground reduced): Invokes reduced intensity blue or 12th colour of the LUT as a foreground colour.
- COL 4/13 – MGFR (Magenta foreground reduced): Invokes reduced intensity magenta or 13th colour of the LUT as a foreground colour.
- COL 4/14 – CNFR (Cyan foreground reduced): Invokes reduced intensity cyan or 14th colour of the LUT as a foreground colour.
- COL 4/15 – WHFR (White foreground reduced): Invokes reduced intensity white or 15th colour of the LUT as a foreground colour.
- COL 5/0 – BKB (Black background): Invokes black or 0th colour of the LUT as a background colour.
- COL 5/1 – RDB (Red background): Invokes red or 1st colour of the LUT as a background colour.
- COL 5/2 – GRB (Green background): Invokes green or 2nd colour of the LUT as a background colour.

– COL 5/3 – YLB (Yellow background):	Invokes yellow or 3rd colour of the LUT as a background colour.
– COL 5/4 – BLB (Blue background):	Invokes blue or 4th colour of the LUT as a background colour.
– COL 5/5 – MGB (Magenta background):	Invokes magenta or 5th colour of the LUT as a background colour.
– COL 5/6 – CNB (Cyan background):	Invokes cyan or 6th colour of the LUT as a background colour.
– COL 5/7 – WHB (White background):	Invokes white or 7th colour of the LUT as a background colour.
– COL 5/8 – TRB (Transparent background):	Invokes transparent or 8th colour of the LUT as a background colour.
– COL 5/9 – RDBR (Red background reduced):	Invokes reduced intensity red or 9th colour of the LUT as a background colour.
– COL 5/10 – GRBR (Green background reduced):	Invokes reduced intensity green or 10th colour of the LUT as a background colour.
– COL 5/11 – YLBR (Yellow background reduced):	Invokes reduced intensity yellow or 11th colour of the LUT as a background colour.
– COL 5/12 – BLBR (Blue background reduced):	Invokes reduced intensity blue or 12th colour of the LUT as a background colour.
– COL 5/13 – MGBR (Magenta background reduced):	Invokes reduced intensity magenta or 13th colour of the LUT as a background colour.
– COL 5/14 – CNBR (Cyan background reduced):	Invokes reduced intensity cyan or 14th colour of the LUT as a background colour.
– COL 5/15 – WHBR (White background reduced):	Invokes reduced intensity white or 15th colour of the LUT as a background colour.

A foreground colour and a background colour specified remain unchanged until a new colour is specified. Execution of NSR or CS also resets colour specification.

The default foreground colour is white and the default background colour is transparent.

B.3.2.2 Character size controls

The following codes define a character size to be used. The character size, once defined, remains unchanged until it is altered by following C1 control set, a display control command (P²)-TEXT), NSR or CS.

– NSZ (Normal size):	The character width and height are the same as that of the character block specified by the P-TEXT command described later.
– SSZ (Small size):	The character width and height become half that of NSZ.
– MSZ (Medium size):	The character width becomes half that of NSZ. The character height is the same as that of NSZ.

²) P stands for “photo”.

- SZX (Size control): SZX is the C1 control code which specifies a character size DBH, DBW and DBS by one byte parameter following the SZX code.
- SZX 4/1 – DBH (Double height): The character height becomes double that of NSZ. The character width is the same as that of NSZ.
- SZX 4/4 – DBW (Double width): The character width becomes double that of NSZ. The character height is the same as that of NSZ.
- SZX 4/5 – DBS (Double size): The character width and height become double that of NSZ.

The default dimensions of the character size are 'normal'.

B.3.2.3 Cursor controls

- CON (Cursor ON): The cursor display is made visible.
- COF (Cursor OFF): The cursor display is turned off.

The default state is cursor off.

B.3.2.4 Flashing controls

Flashing is a process where a foreground colour is alternately turned into a background colour. The flashing attribute can be placed on the character size basis or on a colouring block basis. The default state is steady.

The FLC (Flashing control) code, which is one of the C1 control codes, specifies flashing attributes by one byte parameter following the FLC code.

- FLC 4/0 – Normal flash: Applies the normal (50%) flash attributes.
- FLC 4/7 – Inverted flash: Applies the inverted flash attributes.
- FLC 4/4 – Three phase flash 1: Applies the 1st phase of three phase flashing.
- FLC 4/2 – Three phase flash 2: Applies the 2nd phase of three phase flashing.
- FLC 4/1 – Three phase flash 3: Applies the 3rd phase of three phase flashing.
- FLC 4/3 – Three phase inverted flash 1: Applies the inverted 1st phase of three phase flashing.
- FLC 4/5 – Three phase inverted flash 2: Applies the inverted 2nd phase of three phase flashing.
- FLC 4/6 – Three phase inverted flash 3: Applies the inverted 3rd phase of three phase flashing.
- FLC 4/15 – Steady: Cancels the application of any flash attributes.

The sequence FLC 4/0, A, B, C, D, FLC 4/15 causes characters "ABCD" to be normal-flashed.

B.3.2.5 Repeat controls

The RPC (Repeat control) code causes the following transmitted G-set character, if the following character is a non-spacing character, both the non-spacing character and the next character, to be displayed a number of times specified by the following byte. The byte must be from columns 4 through 7. The repeat count is given by the binary number comprising bits b_6 through b_1 with b_6 being the MSB. If the byte following the RPC code is not from columns 4 through 7, the RPC code is not executed. RPC 4/0 has a special meaning that repeat to end of line.

The sequence A, RPC 4/3, B, C causes "A BBB C" to be displayed.

B.3.2.6 Lining controls

The following codes control the application of the lining attribute:

- STL (Start lining): Applies the lining attribute. Characters are displayed with an underline whereas mosaic patterns are displayed in separated font.
- SPL (Stop lining): Resets the lining attribute.

B.3.2.7 Conceal controls

The CDC (Conceal display control) codes control the conceal display attribute. When the conceal display attribute is applied, characters and DRCS are made invisible. The character position takes ordinary advance. Concealed characters are made visible by the specified user action. The CDC code, which is one of the C1 control codes, specifies conceal display attributes by one byte parameter following the CDC code.

- CDC 4/0 – CDY (Conceal display): Applies the conceal display attribute to the following characters.
- CDC 4/15 – SCD (Stop conceal display): Resets the conceal display attribute.

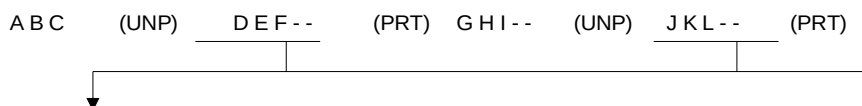
The sequence A, B, C, CDC 4/0, D, E, F, ... , CDC 4/15 causes a terminal to display "ABC" first and "DEF ..." is made visible by a user action.

B.3.2.8 Protect controls

Protected character blocks cannot be altered, manipulated or erased by user action. The entire screen is protected by default.

- UNP (Unprotect): Following character are made unprotected.
- PRT (Protect): Following characters are made protected.

Example:



These characters are unprotected, while the rest are protected.

B.3.2.9 Photo Macro Command

B.3.2.9.1 General

The photo macro command facility provides the capability for a string of any characters and commands including transparent mode commands to be stored within the terminal and to be subsequently executed via a single photo macro call. A photo macro name is one of the 96 characters (from 2/0 to 7/15).

B.3.2.9.2 P-DEF MACRO

The P-DEF MACRO is specified by the parameter 4/0 following the P-MACRO code which is one of the C1 control codes. The P-DEF MACRO command is used to define a photo macro command. The character following the code is the name of the photo macro command. All character codes and transparent mode bit sequences are stored within the terminal under the specified photo macro name. Definition of the photo macro command terminates upon receipt of P-DEF MACRO, P-DEFP MACRO, P-DEFT MACRO and P-MACRO END commands.

Characters and commands following the photo macro name character are not executed at the time of reception. Neither the terminating code nor its preceding ESC code (in a 7-bit environment) is stored as part of the photo macro command.

Definition of a photo macro command replaces any previously existing photo macro command under the same name. If there are no valid characters or commands between the photo macro name and the terminating code (including its preceding ESC code), the photo macro command is deleted. All photo macros are simultaneously deleted with the P-RESET command or CS.

B.3.2.9.3 P-DEFP MACRO

The P-DEFP MACRO is specified by the parameter 4/1 following the P-MACRO C1 control code.

This command is the same as the P-DEF MACRO command except that it simultaneously stores and executes the incoming characters that make up the macro.

B.3.2.9.4 P-DEFT MACRO

The P-DEFT MACRO is specified by the parameter 4/2 following the P-MACRO C1 control code.

This command is used to define a transmit-macro. Transmit-macros, when called, are not executed, but are transmitted back to the host computer.

B.3.2.9.5 P-MACRO END

The P-MACRO END is specified by the parameter 4/15 following the P-MACRO C1 control code.

This command terminates the current P-DEF MACRO, P-DEFP MACRO or P-DEFT MACRO operation.

B.3.3 Display control command set

The display control command set provides control over display attributes, presentation level data format and display structure definition. Display control commands have effect on textual information, photographic information and geometric information.

The display control command set is one of the G-sets. A display control command is composed of one single byte opcode and or more operands if necessary. Each operand consists of one or more bytes of numeric data or bits combination.

There are three types of operands:

- i) fixed format;
- ii) single-value format;
- iii) multi-value format.

The fixed format operands consist of one or more bytes of numeric data or bits combination whose length and interpretation depend on the opcode with which they are used.

The single-value operands consist of one, two, three or four bytes of numeric data, as determined by the P-DOMAIN command described later. They are interpreted as unsigned integers composed of the sequence of concatenated bits taken consecutively (high order bit of b_6 to low order bit or b_1) from the numeric data bytes as shown in Figure B.3-1 (1).

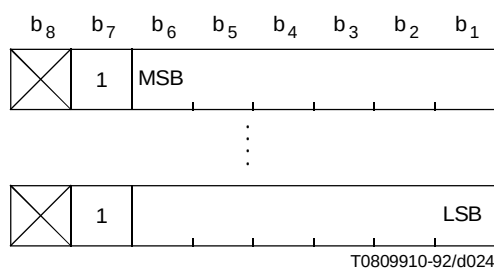


Figure B.3-1 (1)/T.101 – Single-Value Operand Format

The multi-value operands consist of one to eight bytes of numeric data, as determined by the P-DOMAIN command. These operands are used to specify coordinate information, when used to specify dimensions, or colour information, when used in conjunction with the SET LUT command described later.

All coordinate operands are interpreted as signed, two's complement numbers, with the MSB representing 0.5.

When the multi-value operand is used along with the SET LUT command, it specifies an unsigned colour value in the RGB (red-green-blue) colour system. The representation of the colour value within the multi-value operand is shown in Figure B.3-1 (2), where the colour value is given by an unsigned binary decimal. The colour value '0' indicates the lowest intensity, while '1' does the highest intensity.

Table B.3-3 shows the types of operands used by each of the opcodes.

Entire display control command set is shown in Table B.3-4.

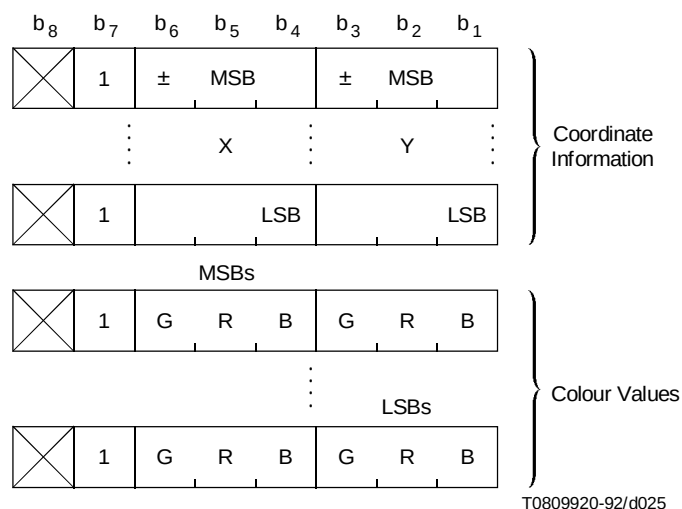


Figure B.3-1 (2)/T.101 – Multi-Value Operand Format

Table B.3-3/T.101 – Types of operands used by display control command

Opcode	Type of operand
P-RESET	Fixed format, One byte
P-DOMAIN	Fixed format, Two bytes
LOGICAL-PEL	Multi-value format (2)
DISPLAY MODE	Fixed format, One byte
P-TEXT	Fixed format, Two bytes / Multi-value format (1)
P-WAIT	Single-value format (1)
RASTER	Multi-value format (1)
HEADER RASTER	Multi-value format (1)
SET LUT	Single-value format (1) / Multi-value format (1)
P-BLINK	Single-value format (2) / Fixed format, Three bytes
AREA	Multi-value format (2)
SET FRAME	Single-value format (2n) n: The number of frames
ASSIGN FRAME	Single-value format (1)
() The number of operands.	

Table B.3-4/T.101 – Display Control Command Set

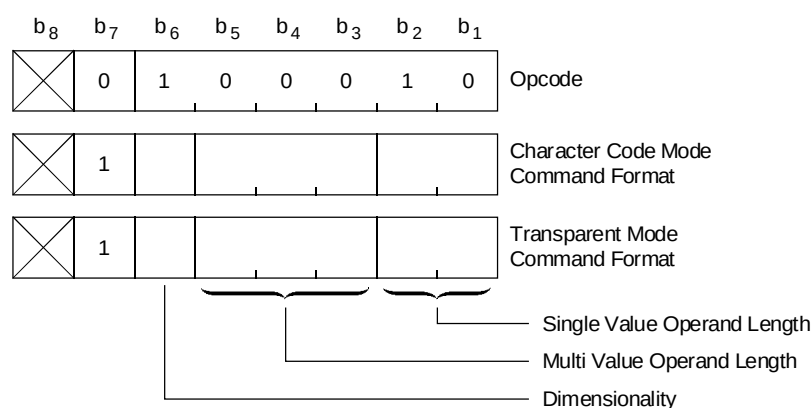
					10	11	12	13	14	15
					b ₇	b ₆	b ₅			
b ₄	b ₃	b ₂	b ₁		2	3	4	5	6	7
0	0	0	0	0			NUMERIC DATA			
0	0	0	1	1	P-RESET					
0	0	1	0	2	P-DOMAIN					
0	0	1	1	3	LOGICAL PEL					
0	1	0	0	4	DISPLAY MODE	P-TEXT				
0	1	0	1	5	AREA					
0	1	1	0	6	SET FRAME					
0	1	1	1	7	ASSIGN FRAME					
1	0	0	0	8		RASTER				
1	0	0	1	9		HEADER RASTER				
1	0	1	0	10		SET LUT				
1	0	1	1	11						
1	1	0	0	12	P-WAIT	P-BLINK				
1	1	0	1	13						
1	1	1	0	14						
1	1	1	1	15						

B.3.3.2 Attribute control functions

B.3.3.2.1 P-DOMAIN

The P-DOMAIN command defines the operand length for both character code mode commands (display control commands, PDI and MVI) and transparent mode commands (photographic commands) and the dimensionality. These parameters, once set, remain unchanged until altered by another P-DOMAIN command or reset by NSR or CS.

The P-DOMAIN command takes two bytes fixed format operands as shown in Figure B.3-2. The first byte defines the character code commands operand length and the second does the transparent mode commands operand length. Bits of both bytes are interpreted as follows.



T0809930-92/d026

Figure B.3-2/T.101 – P-Domain

i) *Single-value operand length*

b ₂	b ₁	Single-value operand length (Bytes)
0	0	1*
0	1	2
1	0	3
1	1	4

* Default value for both character code commands and transparent mode commands.

[The single value transparent mode command operand format is shown in Figure B.3-3 (1).]

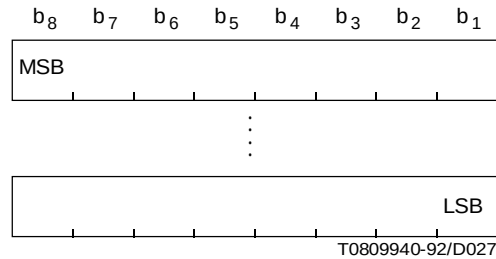


Figure B.3-3 (1)/T.101 – Single-Value Transparent Mode Command Operand

ii) *Multi-value operand length*

b ₅	b ₄	b ₃	Multi-value operand length (Bytes)
0	0	0	1
0	0	1	2 **
0	1	0	3 ***
0	1	1	4
1	0	0	5
1	0	1	6
1	1	0	7
1	1	1	8

** Default value for transparent mode commands.

*** Default value for character code mode commands.

[The multi-value transparent mode command operand format is shown in Figure B.3-3 (2).]

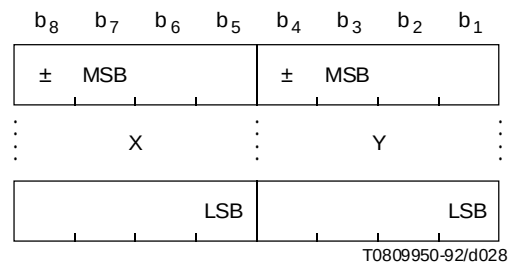


Figure B.3-3 (2)/T.101 – Multi-Value Transparent Mode Command Operand

iii) Dimensionality

Bit 6 of each operand byte determines the dimensionality of the coordinate specification. 0 indicates two dimensional mode, and 1 indicates three dimensional mode; however, the definition of three dimensional mode is reserved for future standardization. Therefore, bit 6 should be always 0 until final standardization.

iv) Operand length

If an operand following a character code command opcode is shorter than the length previously specified by the P-DOMAIN command (or the implicit length in the fixed format case), then the remainder of the operand is padded with zeros, unless otherwise indicated in the definition of the command. If an operand following a character code mode command opcode is longer than the length previously specified by the P-DOMAIN command (or the implicit length), it is taken as an indication to repeat the execution of the command with the subsequent numeric data taken as new operands, unless otherwise indicated.

B.3.3.2.2 LOGICAL PEL

The LOGICAL PEL command defines the size of the logical pel and the colouring block. The LOGICAL PEL command takes two multi-value operands as shown in Figure B.3-4. The first operand specifies the logical pel size. The width (dx) and height (dy) of a logical pel are interpreted as relative coordinate values.

The second multi-value operand specifies the colouring block size. The colouring block size is common for the foreground colour plane, the background colour plane and the display attribute plane, and is normally larger than the logical pel.

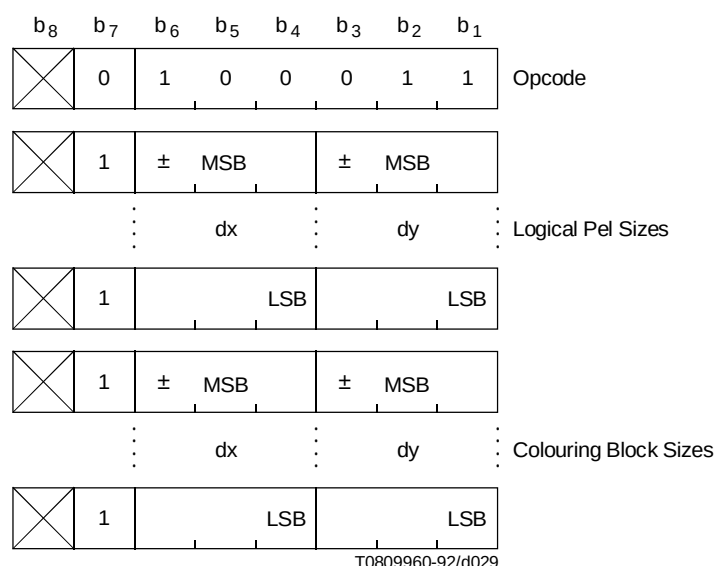


Figure B.3-4/T.101 – Logical pel

B.3.3.2.3 Colour look up table controls

The SET LUT command and the P-BLINK command allow control over the colour look up table (LUT). The block colouring frame's LUT is commonly used for both foreground colours and background colours.

B.3.3.2.3.1 SET LUT

The SET LUT command is used to download RGB colour values into the LUT. The SET LUT command takes one single-value operand followed by one multi-value operand as shown in Figure B.3-5. The single-value operand specifies the LUT entry which is left justified within the operand. The multi-value operand specifies the colour values for the entry assigned. If additional numeric data follows this operand, then the SET LUT command is implicitly repeated with the entry address of the LUT being automatically incremented. The incrementing process starts from the MSB side.

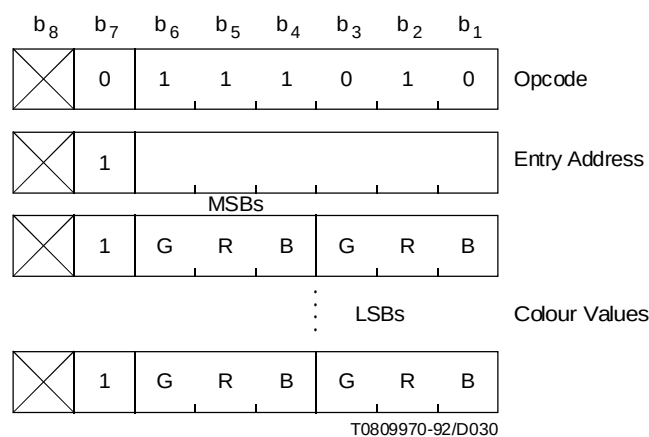


Figure B.3-5/T.101 – SET LUT

If there is no multi-value operand, each value of R, G and B is taken as 0, i.e. transparent.

The default contents of the LUT are shown in Table B.3-5.

Table B.3-5/T.101 – Default Colour Look up Table

Entry Address (RI, B, G, R)		Colour Values			Nominal Colour
		B	G	R	
0	0 0 0 0	0001	0001	0001	Black
1	0 0 0 1	0000	0000	1111	Red
2	0 0 1 0	0000	1111	0000	Green
3	0 0 1 1	0000	1111	1111	Yellow
4	0 1 0 0	1111	0000	0000	Blue
5	0 1 0 1	1111	0000	1111	Magenta
6	0 1 1 0	1111	1111	0000	Cyan
7	0 1 1 1	1111	1111	1111	White
8	1 0 0 0	0000	0000	0000	Transparent
9	1 0 0 1	0000	0000	0111	RI Red
10	1 0 1 0	0000	0111	0000	RI Green
11	1 0 1 1	0000	0111	0111	RI Yellow
12	1 1 0 0	0111	0000	0000	RI Blue
13	1 1 0 1	0111	0000	0111	RI Magenta
14	1 1 1 0	0111	0111	0000	RI Cyan
15	1 1 1 1	0111	0111	0111	Grey

RI Reduced Intensity

B.3.3.2.3.2 P-BLINK

The P-BLINK command causes a blink process at the LUT entry. The blink process periodically overwrites the contents of the specified entry (the “blink-from” colour) and substitutes another entry’s contents in the LUT (the “blink-to” colour).

The P-BLINK command takes two single-value operands followed by three fixed format operands. The first single-value operand is the blink-from colour specification, specified as an LUT entry address. The second single-value operand specifies the blink-to colour.

The first fixed format operand following the second single-value operand specifies the ON interval which is a period of time the blink-to colour is activated. The second fixed format operand specifies the OFF interval which is a period of time the blink-from colour is activated. The third fixed format operand specifies the phase delay. The phase delay refers to the start of the ON interval of the most recently defined active blink process. These values are given in units of 1/10th of a second.

Figure B.3-6 shows the P-BLINK command format.

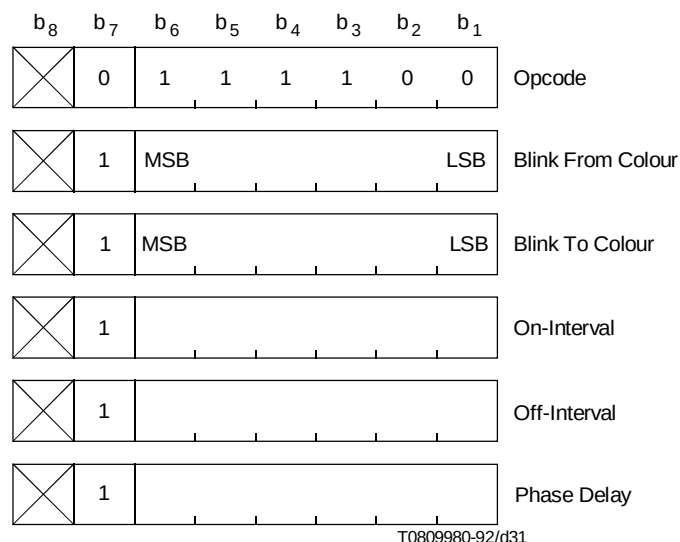


Figure B.3-6/T.101 – P-BLINK

B.3.3.2.4 Raster colour controls

The RASTER command and the HEADER RASTER command specify the raster colour and the header raster colour, respectively. Both commands have the same format shown in Figures B.3-7 and B.3-8. Colour values are specified by a one multi-value operand. Once set, the raster colour or the header raster colour remains unchanged until another RASTER command, another HEADER RASTER COMMAND, or CS is executed. The default colour values of both the raster and the header raster are R = G = 0 and B = 0.0111 --- (= 1/2; Reduced blue colour).

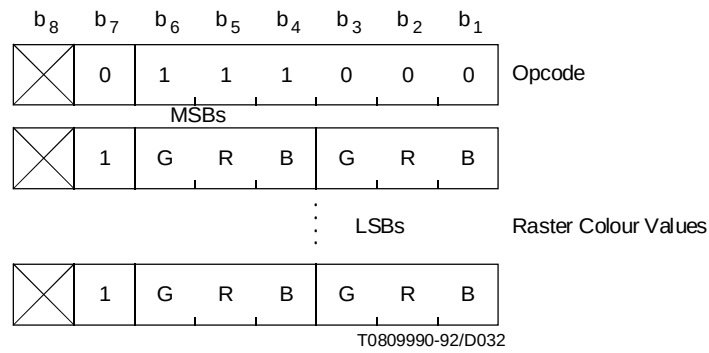


Figure B.3-7/T.101 – RASTER

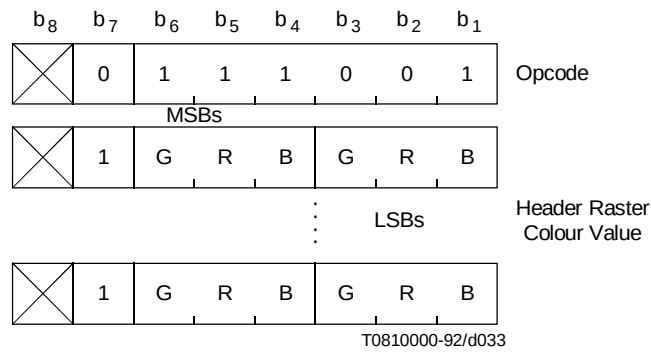


Figure B.3-8/T.101 – HEADER RASTER

B.3.3.2.5 P-RESET

The P-RESET command is used to selectively reset parameters as described below. It takes one single byte fixed format operand (see Figure B.3-9).

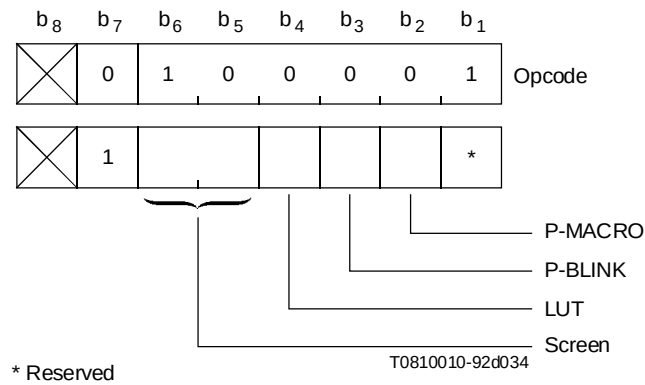


Figure B.3-9/T.101 – P-RESET

b ₆	b ₅	Screen
0	0	No operation.
0	1	The entire pattern plane is reset to 0. The entire foreground colour is set to 7th colour of the LUT (default colour: white), while the entire background colour is set to 8th colour of the LUT (default colour: transparent).

When the screen is cleared (other than b₆ = b₅ = 0), all flashing, conceal and protect attributes are cancelled.

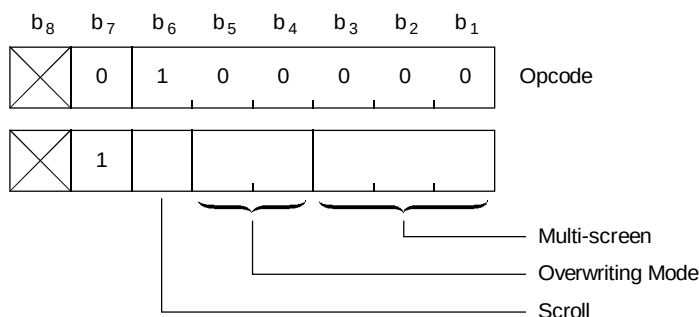
b ₄	LUT
0	No operation.
1	The LUT is set to default colours.
b ₃	P-BLINK
0	No operation.
1	All blink processes are terminated.
b ₂	P-MACRO
0	No operation.
1	All photo-MACROs are cancelled.

B.3.3.2.6 DISPLAY MODE

The DISPLAY MODE command specifies the display attributes. The DISPLAY MODE opcode takes one single byte fixed format operand whose bits (from b₁ to b₆) control individual attributes (see Figure B.3-10).

b ₆	Scroll
0	Scroll off
1	Scroll on

The default state is scroll off.



TISO0810020-92/d035

Figure B.3-10/T.101 – DISPLAY MODE

Operations of scroll on mode and scroll off mode are as follows:

- i) *Scroll off mode*
 - *In case of a photographic command*
Data given are written to the point specified by the command.
 - *In case of text*
When the next character position exceeds the lower boundary of the screen, the next character will be displayed at the upper left corner of the screen.

ii) *Scroll on mode*

– *In case of a photographic command*

When the scroll on mode is activated by the DISPLAY MODE command, a register for storing a Y coordinate value (Y coordinate register) is first preset to 0.75. Upon receipt of new data, full contents of the screen are scrolled upward by the distance given as the difference between the contents of the Y coordinate register and the Y coordinate value indicated by the command, then new data is written at the position $Y = 0$. If the difference becomes 0.75, it is taken as 0. After that, the Y coordinate value indicated by the command is set to the Y coordinate register.

When a photographic command indicates to write data on a block by block basis rather than a line by line basis and the new field block position exceeds the lower boundary of the screen, the full contents of the screen are scrolled upward so that the bottom of the new field block meets the lower boundary of the screen.

– *In case of text*

When the next character position exceeds the boundary of the screen, the full contents of the screen are scrolled so that the next character can be displayed within the screen. Scrolling occurs in a direction perpendicular to the character path.

b_5	b_4	Overwriting mode.
0	0	New data for the pattern plane replaces the old one. $M_i \leftarrow D_i$ (replace mode)
0	1	New data and the old one is ORed and the result is written to the memory. $M_i \leftarrow M_i \quad \text{OR} \quad D_i$
1	0	New data and the old one is ANDed and the result is written to the memory. $M_i \leftarrow M_i \quad \text{AND} \quad D_i$
1	1	New data and the old one is Exclusive ORed and the result is written to the memory. $M_i \leftarrow M_i \quad \text{EOR} \quad D_i$

where M_i is the content of the pattern plane memory, and D_i is new data for it.

The default mode is the replace mode.

b_3	Multi-screen
0	Single-screen (default)
1	Multi-screen

Four smaller frames are assembled into a single split-screen.

b_2	b_1	Multi-screen position (these bits are effective only when $b_3 = 1$).
0	0	The frame is displayed at the upper left part of the screen which is separated into four parts.
0	1	The frame is displayed at the upper right part of the screen which is separated into four parts.
1	0	The frame is displayed at the lower left part of the screen which is separated into four parts.
1	1	The frame is displayed at the lower right part of the screen which is separated into four parts.

B.3.3.2.7 P-TEXT

The P-TEXT command is used to modify parameters which describe the text display manner. The opcode is followed by a two byte fixed format operand and a multi-value operand (see Figure B.3-11).

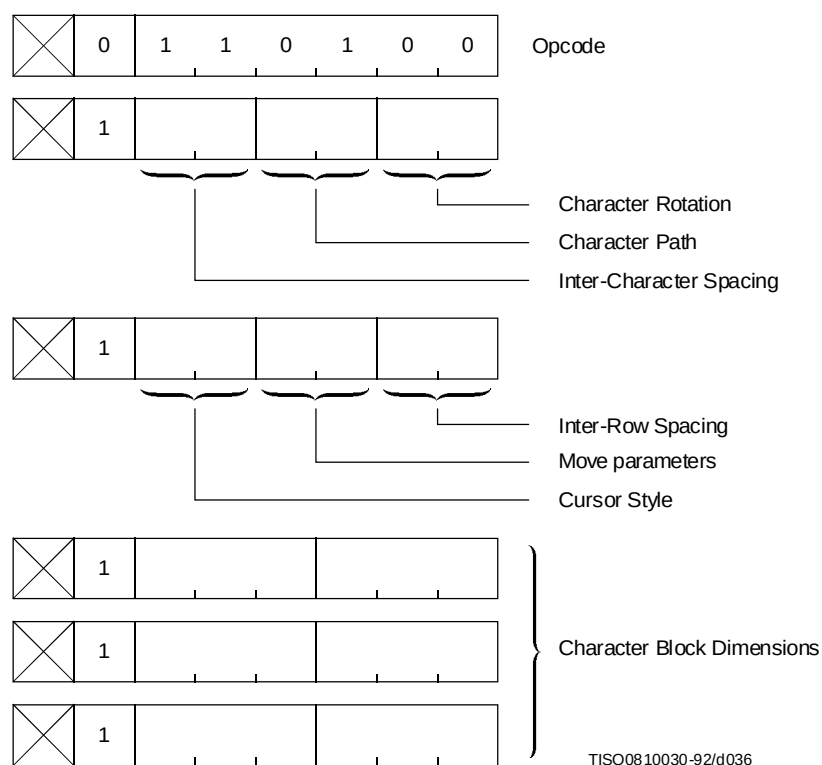
The fixed format operand determines parameters as follows:

Inter-Character Spacing

Bits b_5 and b_6 of byte 1 are used to determine the distance the cursor is moved (in multiples of the character field dimension lying parallel to the character path) after a character is displayed or after a SPACE or APB (backspace) or APF (horizontal tab) character is received.

b ₆	b ₅	Inter-character spacing
0	0	1 (default)
0	1	1.25
1	0	1.5
1	1	Proportional to the width of the character

The three fixed inter-character spacings (1, 1.25 and 1.5) are interpreted as multiplicative functions of the dimension of the current character field lying parallel to the character path that are applied to movements of the cursor. In the proportionally spaced mode, the inter-character spacing is a variable that may be a function of the width of the actual pattern deposited as well as the current character size and font style. The proportional spacing algorithm is implementation dependent. However, each character shall be completely contained within the area defined by the current character field. This means that the exact number of characters per line is not known in proportional spacing mode, but it is at least as many characters per line as would be allowed by the current character field dimensions. The default inter-character spacing is a fixed space of 1.



TISO0810030-92/d036

Figure B.3-11/T.101 – P-TEXT

Character Path Movement

Bits b₃ and b₄ of byte 1 determine the direction of the character path, that is, the direction in which the cursor is automatically advanced after a character is deposited. The character path is defined relative to horizontal within the unit screen and is independent of the character rotation. The default character path is right.

b ₄	b ₃	Character path movement
0	0	right (default)
0	1	left
1	0	up
1	1	down

Character Rotation

Bits b_1 and b_2 of byte 1 are used to specify character rotation.

b_2	b_1	Rotation
0	0	0° (default)
0	1	90°
1	0	180°
1	1	270°

Rotation causes the character field to rotate counter-clockwise about the character field origin. This rotation is measured relative to horizontal within the unit screen and is independent of the character path. The character field origin is the lower left corner of the character field at the default 0° rotation regardless of the sign of the character field dimensions dx and dy . All images within a character field are affected by rotation so that the relative position of the images within the character field is unchanged.

Cursor styles

Bits b_5 and b_6 of byte 2 are used to determine the display style of the cursor symbol.

b_6	b_5	Cursor style
0	0	underscore (default)
0	1	block
1	0	cross hair
1	1	custom

This cursor is located in the position in which the next character is to deposited. The underscore cursor symbol is a single line the width of the current character field at the bottom of the character field. The block cursor symbol is a solid block or outline of block whose size is the size of the current character field. The cross-hair cursor symbol consists of a vertical line and a horizontal line that intersect at the centre of the character field and whose height and width are equal to the height and width of the current character field. The definition of the shape of the custom cursor symbol is implementation independent.

Move Attributes

Bits b_3 and b_4 of byte 2 are used to define the relationship between movement of the cursor and movement of the graphics drawing point.

b_4	b_3	Move attributes
0	0	The cursor and the drawing point move together
0	1	The cursor leads
1	0	The drawing point leads
1	1	The cursor and the drawing point move independently (default)

If the cursor and the drawing point are set to move together (00), then whenever the cursor is moved (such as when characters are displayed) the graphic drawing point is moved with it, maintaining its alignment relative to the cursor. Correspondingly, whenever the drawing point is moved (such as with a geometric drawing primitive) the cursor is also moved so as to maintain its alignment relative to the drawing point.

If the cursor is defined as leading (01), then every time the cursor is moved the drawing point will move along with it but not vice versa.

If the drawing point is set to lead (10) the cursor then every time the drawing point is moved the cursor will move with it but not vice versa.

If the drawing point and the cursor are set to move independently (11), then movement of one will not affect the position of the other.

The alignment of the drawing point corresponds to the character field origin for the underscore cursor and block cursor, and the centre of the character field for the crosshair cursor and custom cursor.

The execution of a P-TEXT command shall cause alignment of the drawing point if the “move together” or “cursor leads” move attribute is in effect after execution. The execution of a P-TEXT command shall have no effect on the position of the character field origin.

Inter-Row Spacing

Bits b_1 and b_2 of byte 2 determine the inter-row spacing of characters, which defines the relative location of the cursor when it is advanced to a new line in a direction perpendicular (-90°) to the character path, either automatically as described below or by the APD (line feed) or APU (vertical tab) characters. Inter-row spacings are interpreted as multiplicative functions of the dimension of the character field that lies perpendicular to the character path.

b_2	b_1	Inter-row spacing
0	0	1 (default)
0	1	1.25
1	0	1.5
1	1	2

When using fixed or proportional inter-character spacing, if the character field origin is advanced along the character path such that any part of the full corresponding character field would exceed the edge of the unit screen, an automatic APR (carriage return) and APD (line feed) are executed. An inter-row spacing of 1, in which the character field on the current row abuts the character field of the previous row, is the default.

Character Block Dimensions

The multi-value operand specifies the width (dx) and the height (dy) of the character block which defines the “normal” character size (default values are $dx = 0.0625$ and $dy = 0.09375$.) The character block size specified remains unchanged until another P-TEXT command, NSR or CS is received.

If the multi-value operand is omitted, the character block size remains unchanged. If more operands follow the specified opcodes, they are discarded.

If dx is negative, the character pixel patterns are reflected about the vertical centre axis of the character field. If dy is negative, the character pixel patterns are reflected about the horizontal centre axis of the character field.

B.3.3.2.8 P-WAIT

The P-WAIT command causes a specific time of delay in processing of text and commands following the command. The command format is shown in Figure B.3-12. The P-WAIT opcode takes a single-value format operand which gives the time delay in units of 1/10th of a second.

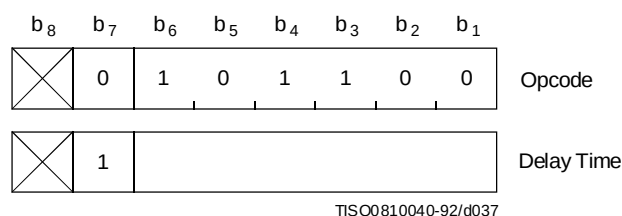


Figure B.3-12/T.101 – P-WAIT

B.3.3.3 Multi-frame control functions

B.3.3.3.1 General

This annex provides the functionality which allows superimposed display of several frames and partition of the screen. The multi-frame display structure realizes display of textual information over pictorial information or animated pictorial information display over background pictorial information.

Screen partition provides various display structure: different multi-frame structure and/or frame composition can be realized for different parts of the screen. This function enables to display different types of information such as photographic information and geometric information in a single screen with minimum raster memory requirement. For example, area A in Figure B.3-13 has two photographic frames. Each frame consists of M-bits per pixel, while the rest of the screen (area B) has three N-bits frames. If $2M$ equals to $3N$, both area A and B can be accommodated in the same raster memory.

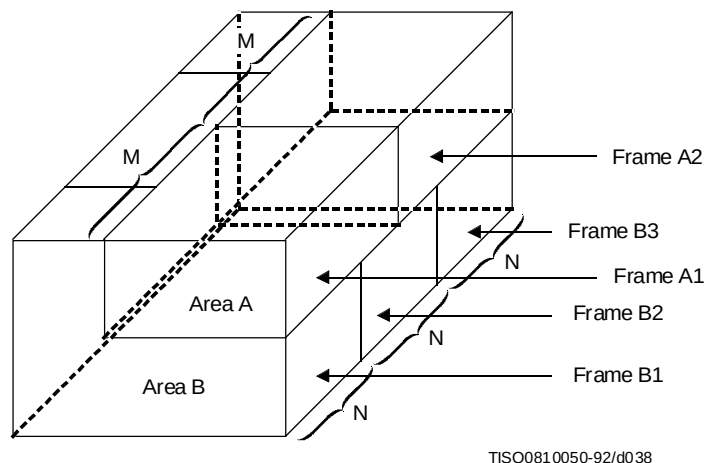


Figure B.3-13/T.101 – Screen Partition Concept

B.3.3.3.2 General flow of multi-frame display operation

The general flow for multi-frame display operations is shown in Figure B.3-14.

At first, an active drawing area is defined. The following operations like bit assignment, frame selection and drawing commands are only effective to a defined area. Secondly, a multi-frame structure and bit-assignment for each frame is specified. Finally, a desired frame is selected. After that, general drawing commands follow.

B.3.3.3.3 AREA

The AREA command defines an active drawing area. Subsequent drawing commands are only effective in the defined area. The rest of the screen remains unchanged. For example, the scrolling of a picture can be executed within an area. There is only one active drawing area on the screen at a time. Area definition is cancelled, upon reception of a new AREA command.

An area is rectangular in shape, and defined by the originating position (x , y) and width (dx) and height (dy) (see Figure B.3-15). The command format is shown in Figure B.3-16. The AREA command itself has no effect on the displayed picture.

B.3.3.3.4 SET FRAME

The SET FRAME command specifies the raster memory configuration relevant to the defined area. The command format is:

[Opcode a1, b1, a2, b2, . . .]

as shown in Figure B.3-17. Here 'ai' is the frame index and 'bi' is the number of bits per raster memories allocated to the frame. The frame index indicates relative display priority. If 'ai' is smaller than 'aj', the 'ai' frame has a higher display priority than the 'aj' frame. Any frame, however, has a higher display priority than the full screen background layer.

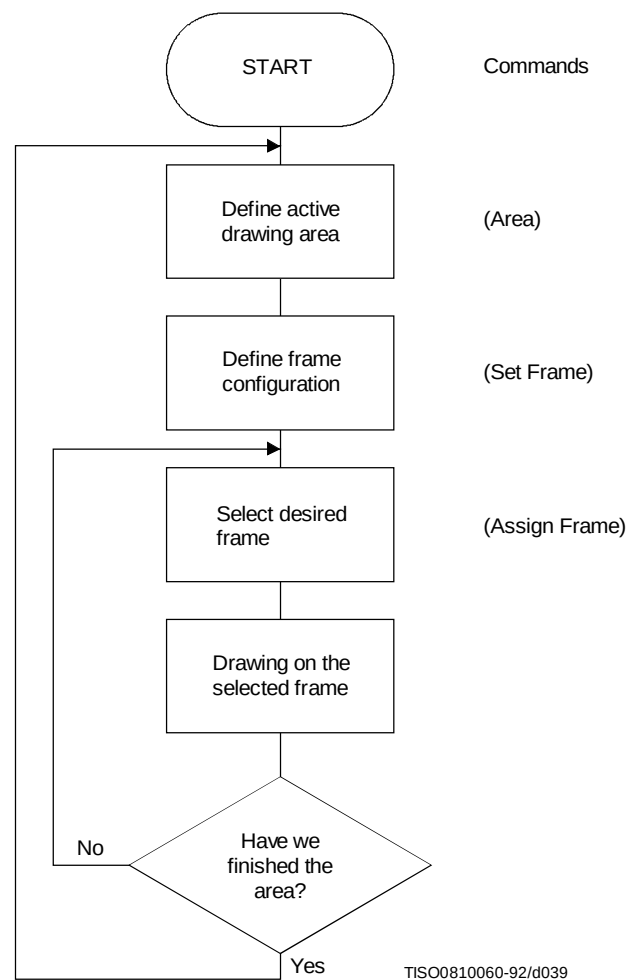


Figure B.3-14/T.101 – General Flow of Multi-Frame Display Operations

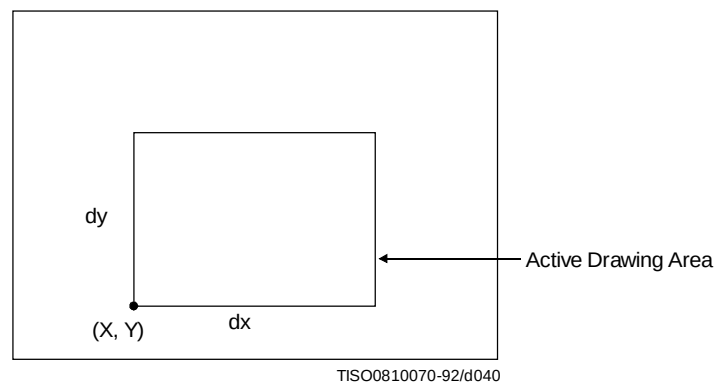


Figure B.3-15/T.101 – Active Drawing Area Definition

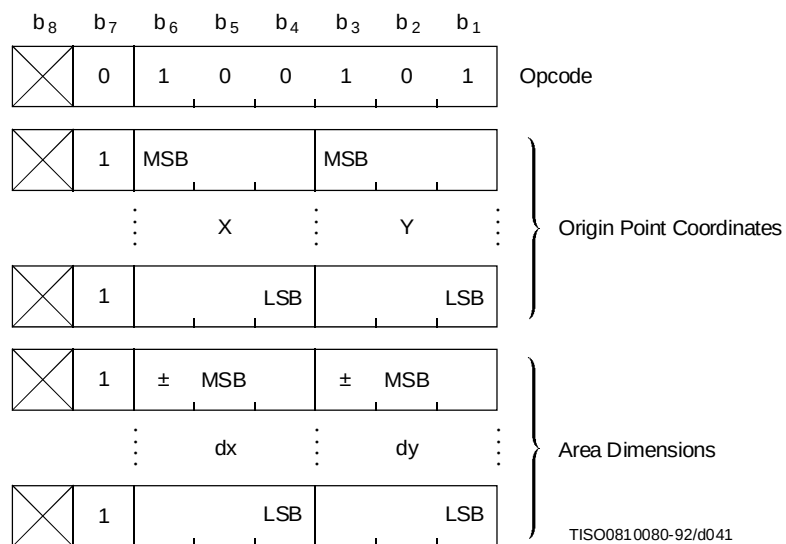


Figure B.3-16/T.101 – AREA

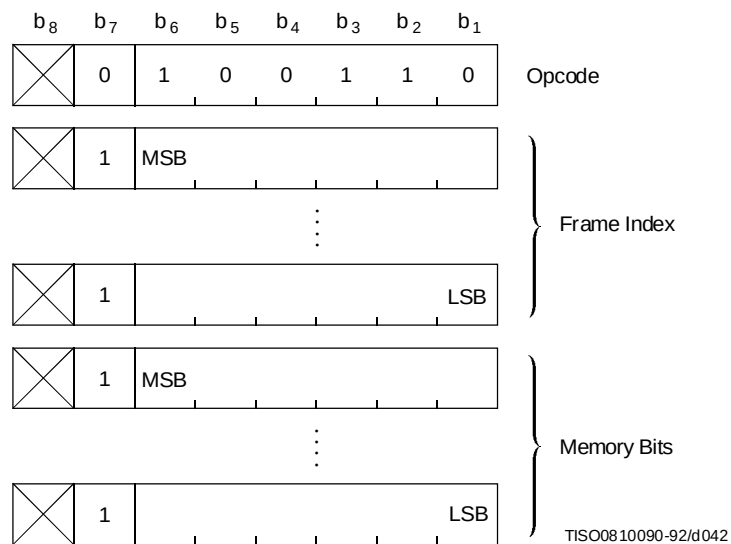


Figure B.3-17/T.101 – SET FRAME

B.3.3.3.5 ASSIGN FRAME

The ASSIGN FRAME command selects the frame on which succeeding drawings are made. The frame is assigned by the frame index stipulated by the SET FRAME command. The command format is:

[Opcode ai]

as shown in Figure B.3-18. Here, 'ai' is the frame index.

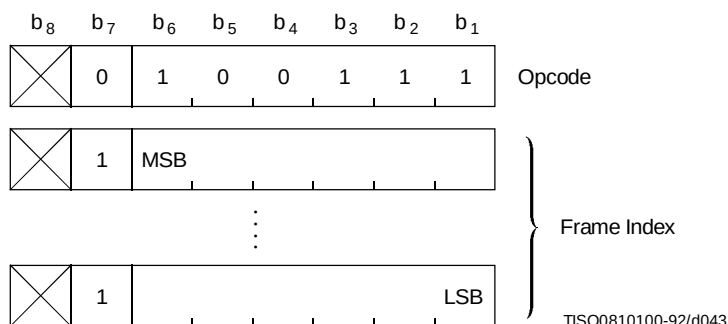


Figure B.3-18/T.101 – ASSIGN FRAME

Table B.3-6 shows commands controlled by the ASSIGN FRAME command: those commands have their effects only on the assigned frame.

Table B.3-6/T.101 – Commands Controlled by ASSIGN FRAME

Table set	Commands controlled by ASSIGN FRAME
C0 control set	None
C1 control set	None
Display control command set	P-RESET, SET LUT, P-BLINK and P-WAIT ^{a)}
Character sets	All
Mosaic sets	All
Dynamically Redefinable Character Sets	All
PDI set	All accept DOMAIN, TEXTURE, WAIT, SELECT COLOUR and BLINK
MVI set	MOVE
Photographic commands	All except P-DRCS 1, P-DRCS 2 and Telesoftware
^{a)} Only the case at when this command is used between the MVI START and the MVI END.	

B.3.4 Character Sets

B.3.4.1 Japanese-Kanji Character Set

The Japanese-Kanji character set defines Japanese-Kanji characters, Japanese-phonetic signs (Katakana and Hiragana), Roman characters, numerics and additional characters.

The Japanese-Kanji character set is basically defined in Recommendation T.52.

However, the Japanese-Kanji character set used in this Recommendation has the following differences:

- i) the two characters in 84/05 and 84/06 are not used;
- ii) additional 169 characters in lines 92-94 are used.

Table B.3-7 shows the Japanese-Kanji character set. Two byte codes are used for the Japanese-Kanji character set, where the first byte represents the 'Ku' number and the second byte represents the 'Ten' number. Characters 1 'Ku' 13 'Ten' through 18 'Ten' and 2 'Ku' 94 'Ten' are defined as non-spacing characters. Exact interpretation of 'Ku' or 'Ten' is the number specified by the byte plus 32.

The Japanese-Kanji characters can be displayed in normal size, double width, double height and double size.

B.3.4.2 Primary Character Set

The primary character set defines 52 Roman characters, 10 numerics and 32 marks. Coded representation of Latin based characters is specified in Recommendation T.51. Table B.3-8 shows the primary character set. A primary character is represented by a single byte code. Any character size including medium size and small size are valid for the primary characters.

B.3.4.3 Katakana Character Set

The Katakana character set defines 63 Japanese phonetic signs. Table B.3-9 shows the Katakana character set. A Katakana character is represented by a single byte code. Any character sizes are valid.

B.3.5 Mosaic Sets

Two mosaic sets (Mosaic 1 set and Mosaic 2 set) are defined. Single byte codes are used for both Mosaic 1 and Mosaic 2 sets.

Both contiguous and separated shapes are provided for. In the underline mode activated by STL C1 control character, all characters from the mosaic sets are displayed in separated graphics representation.

B.3.5.1 Mosaic 1 Set

The Mosaic 1 set defines 59 graphic characters shown in Table B.3-10.

B.3.5.2 Mosaic 2 Set

The Mosaic 2 set defines 94 graphic characters which are compatible with the second supplementary set of mosaic characters of Data Syntax II (see Table B.3-11).

B.3.6 Dynamically Redefinable Character Sets (DRCSs)

The Dynamically Redefinable Character Set (DRCS) provides a functionality whereby custom defined patterns can be downloaded and utilized as a G-set. Downloading is executed in the transparent mode through the Photo-DRCS 1 or Photo-DRCS 2 commands.

Table B.3-7/T.101 – Japanese-Kanji Set (1)

								Second byte								First byte																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																											
																b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																						
0	1	0	0	0	0	1	1	(SP)	,	.	:	;	?	!	"	'	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	]	{	}	~	^	&	%	\$	#	@	A	B	C	D	E	F

Table B.3-7/T.101 – Japanese-Kanji Set (2)

[illegible]

T0821560-95/d045

**Table B.3-7/T.101 –
Japanese-Kanji Set (3)**

First byte								Second byte																																																							
b7	b6	b5	b4	b3	b2	b1		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47									
0	1	1	0	1	0	1	21	機	帰	穀	気	汽	畿	折	季	稀	紀	微	規	記	貴	起	軌	輝	飢	騎	鬼	亀	偽	儀	妓	宜	戲	技	擬	欺	犧	疑	祇	義	蟻	誼	議	掬	菊	鞠	吉	吃	喫	枯	橘	詰	砧	杵									
0	1	1	0	1	1	0	22	供	俠	僑	兇	競	共	凶	協	匡	卿	叫	喬	境	峽	強	疆	怯	恐	恭	挾	教	橋	況	狂	狹	矯	胸	脅	興	蕎	鄉	鏡	響	蟻	誼	議	掬	菊	鞠	吉	吃	喫	枯	橘	詰	砧	杵									
0	1	1	0	1	1	1	23	掘	窟	査	靴	轡	窪	熊	限	象	栗	縁	桑	歛	煎	君	薰	訓	群	軍	挾	卦	袈	祁	係	傾	刑	兄	啓	圭	珪	型	契	響	蟻	誼	議	掬	菊	鞠	吉	吃	喫	枯	橘	詰	砧	杵									
0	1	1	1	0	0	0	24	検	権	牽	犬	研	硯	綱	肩	工	巧	巷	謙	賢	軒	遣	鍵	険	頭	驗	餓	元	蔽	幻	弦	更	杭	校	梗	構	絃	絃	型	契	響	蟻	誼	議	掬	菊	鞠	吉	吃	喫	枯	橘	詰	砧	杵								
0	1	1	1	0	0	1	25	后	喉	坑	垢	好	孔	孝	宏	工	巧	巷	謙	賢	軒	遣	鍵	険	頭	驗	餓	元	蔽	幻	弦	更	杭	校	梗	構	絃	絃	型	契	響	蟻	誼	議	掬	菊	鞠	吉	吃	喫	枯	橘	詰	砧	杵								
0	1	1	1	0	1	0	26	此	頃	今	困	坤	壘	婚	恨	懇	昏	昆	根	梱	混	痕	紺	良	魂	些	佐	叉	唆	散	左	差	珊	查	沙	瑳	砂	詐	鎖	酸	坐	座	挫	暫	殘	仔	柴	屢	蕊	司	史	裁	始	赦	熟	章	刀						
0	1	1	1	0	1	1	27	察	拶	撮	擦	札	殺	薩	雜	皐	鯖	捌	鏑	絞	血	晒	三	傘	參	山	慘	撒	穴	叟	燦	叱	集	醜	什	住	充	悉	濕	漆	疾	質	實	斬	暫	殘	仕	仔	柴	屢	蕊	司	史	裁	始	赦	熟	章	刀				
0	1	1	1	1	0	0	28	次	滋	治	爾	璽	痔	磁	示	而	耳	自	蒔	辭	沙	鹿	式	衆	襲	瞞	輜	週	酉	酬	集	醜	什	住	充	悉	濕	漆	疾	質	實	斬	暫	殘	仕	仔	柴	屢	蕊	司	史	裁	始	赦	熟	章	刀						
0	1	1	1	1	0	1	29	宗	就	州	修	愁	拾	洲	秀	秋	終	繡	習	臭	舟	蒐	衆	襲	瞞	輜	週	酉	酬	集	醜	什	住	充	悉	濕	漆	疾	質	實	斬	暫	殘	仕	仔	柴	屢	蕊	司	史	裁	始	赦	熟	章	刀							
0	1	1	1	1	1	0	30	勝	匠	升	召	哨	商	唱	嘗	舐	食	舐	辱	尻	伸	信	侵	唇	寢	審	心	慎	振	招	新	晉	生	羶	存	淡	湛	長	頂	烏	勅	端	筆	綻	彈	斷	椎	塗	妬	屠	通	塚	杜	渡	屆	馬	範	貧	篋	模	溶	力	蔭
0	1	1	1	1	1	1	31	拭	植	殖	燭	織																																																			

**Table B.3-7/T.101 –
Japanese-Kanji Set (4)**

[illegible]

Table B.3-7/T.101 – Japanese-Kanji Set (5)

[illegible]

Table B.3-7/T.101 – Japanese-Kanji Set (6)

[illegible]

Table B.3-7/T.101 – Japanese-Kanji Set (7)

[illegible]

Table B.3-7/T.101 – Japanese-Kanji Set (8)

[illegible]

T0821620-95/d051

Table B.3-8/T.101 – Primary Character Set

					10	11	12	13	14	15	
					b ₇	0	0	1	1	1	1
					b ₆	1	1	0	0	1	1
					b ₅	0	1	0	1	0	1
b ₄	b ₃	b ₂	b ₁		2	3	4	5	6	7	
0	0	0	0	0	⊗	0	@	P	'	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	#	3	C	S	c	s	
0	1	0	0	4	\$	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	[	k	{	
1	1	0	0	12	,	<	L	¥	l		
1	1	0	1	13	–	=	M	]	m	}	
1	1	1	0	14	.	>	N	^	n	–	
1	1	1	1	15	/	?	O	–	o	⊗	

TISO0821630-95/d052

Table B.3-9/T.101 – Katakana Character Set

					10	11	12	13	14	15	
					b ₇	0	0	1	1	1	1
					b ₆	1	1	0	0	1	1
					b ₅	0	1	0	1	0	1
					2	3	4	5	6	7	
b ₄	b ₃	b ₂	b ₁								
0	0	0	0	0	×	ー	タ	ミ			
0	0	0	1	1	。	ア	チ	ム			
0	0	1	0	2	「	イ	ツ	メ			
0	0	1	1	3	」	ウ	テ	モ			
0	1	0	0	4	・	エ	ト	ヤ			
0	1	0	1	5	。	オ	ナ	ユ	Reserved	Reserved	
0	1	1	0	6	ヲ	カ	ニ	ヨ			
0	1	1	1	7	ア	キ	ヌ	ラ			
1	0	0	0	8	イ	ク	ネ	リ			
1	0	0	1	9	ウ	ケ	ノ	ル			
1	0	1	0	10	エ	コ	ハ	レ			
1	0	1	1	11	オ	サ	ヒ	ロ			
1	1	0	0	12	ヤ	シ	フ	ワ			
1	1	0	1	13	ユ	ス	ヘ	ン			
1	1	1	0	14	ヨ	セ	ホ	ベ			
1	1	1	1	15	ツ	ソ	マ	。		×	

NOTE – Characters in 2(10)/7 ~ 2(10)/15 are small size one.

T0821640-95/d053

Table B.3-10/T.101 – Mosaic 1 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
						0	1	2	3	4	5	6	7
b ₄	b ₃	b ₂	b ₁		0								
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									

TISO0810110-95/d054

Table B.3-11/T.101 – Mosaic 2 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
						0	1	2	3	4	5	6	7
b ₄	b ₃	b ₂	b ₁										
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									

TISO0810120-92/d055

B.3.6.1 DRCS 1

DRCS 1 patterns are identified by single byte codes. A DRCS code has a value in the range of 2/1 to 7/14. Pattern downloading is executed through the Photo-DRCS 1 command.

B.3.6.2 DRCS 2

DRCS 2 patterns are identified by two bytes codes. A DRCS code has a value in the range 2/1 2/1 to 7/14 7/14. Pattern downloading is executed through the Photo-DRCS 2 command.

B.3.7 Macro Set

A macro command consists of an arbitrary string of locally stored code that is identified by a code position (macro name) from the macro G-set. This name (from 2/0 to 7/15) thereafter acts as a substitute for the entire string of codes which is downloaded by the Photo-MACRO command.

B.3.8 Picture Description Instruction (PDI) Set

The PDI set is identical to that of Data Syntax III except the TEXT command (see Table B.3-12). However, the programmable texture masks are not supported.

Table B.3-12/T.101 – PDI Set

					10	11	12	13	14	15
					b ₇					
					b ₆					
					b ₅					
b ₄	b ₃	b ₂	b ₁		2	3	4	5	6	7
0	0	0	0	0	RESET	RECTANGLE (OUTLINED)	NUMERIC DATA			
0	0	0	1	1	DOMAIN	RECTANGLE (FILLED)				
0	0	1	0	2		SET and RECT. (OUTLINED)				
0	0	1	1	3	TEXTURE	SET and RECT. (FILLED)				
0	1	0	0	4	POINT SET (ABS)	POLYGON (OUTLINED)				
0	1	0	1	5	POINT SET (REL)	POLYGON (FILLED)				
0	1	1	0	6	POINT (ABS)	SET and POLY. (OUTLINED)				
0	1	1	1	7	POINT (REL)	SET and POLY. (FILLED)				
1	0	0	0	8	LINE (ABS)	FIELD				
1	0	0	1	9	LINE (REL)	INCREMENTAL POINT				
1	0	1	0	10	SET and LINE (ABS)	INCREMENTAL LINE				
1	0	1	1	11	SET and LINE (REL)	INCR. POLY. (FILLED)				
1	1	0	0	12	ARC (OUTLINED)	SET COLOUR				
1	1	0	1	13	ARC (FILLED)	CONTROL STATUS (WAIT)				
1	1	1	0	14	SET and ARC (OUTLINED)	SELECT COLOUR				
1	1	1	1	15	SET and ARC (FILLED)	BLINK				

The following commands in the PDI set and the display control command set have identical effects on display attributes.

<i>PDI set</i>	<i>Display control command set</i>
RESET	P-RESET
DOMAIN	P-DOMAIN, LOGICAL PEL
SET COLOUR	SET LUT
WAIT	P-WAIT
BLINK	P-BLINK

The SELECT COLOUR command has the same effect as the C1 colour control character has.

If there are conflicting commands, the last received command has control over display attributes. In other words, display attributes are common for both PDI commands and photographic commands.

B.3.9 Moving Instruction (MVI) Set

MOVE command (see B.3.9.5) causes the frame specified by ASSIGN FRAME to be moved virtually with reference to the other frames (see Figure B.3-19).

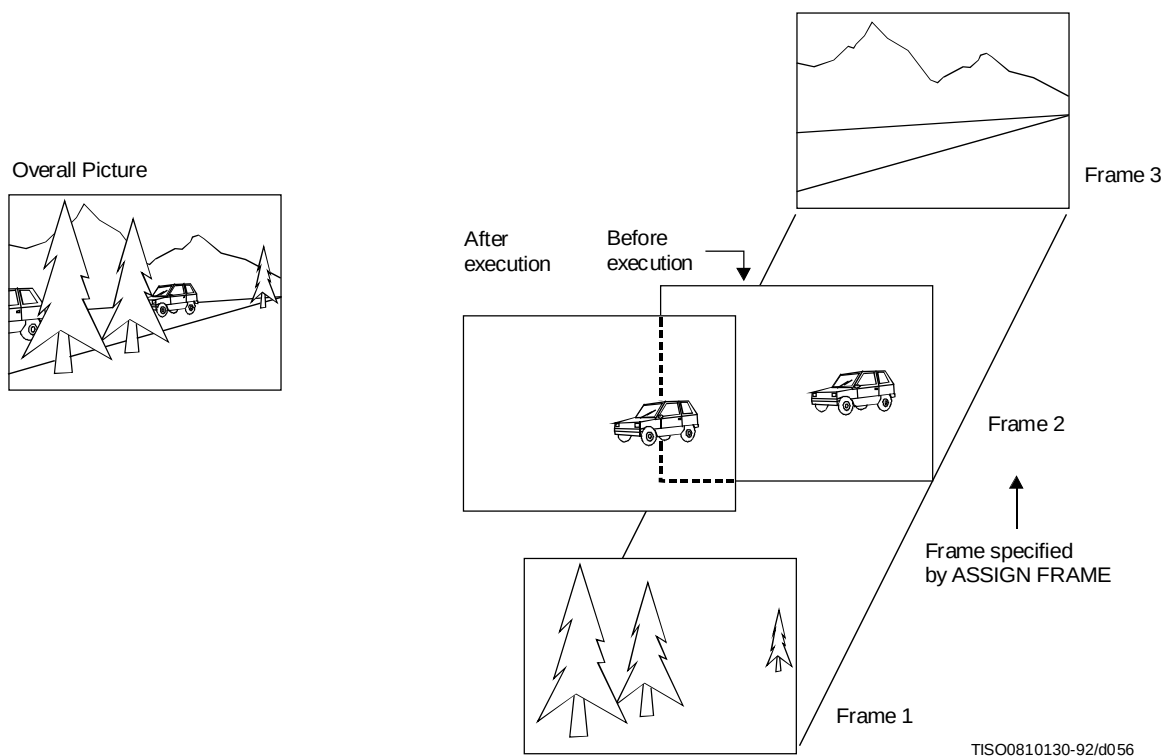


Figure B.3-19/T.101 – Move Instruction Examples

By the use of this command and the multi-frame structure, animated picture presentation is available.

A moving instruction (MVI) is composed of an opcode and operands, being similar to PDIs. (Some instructions have no operands.)

There are single-value operands, multi-value operands and fixed format operands. The operand byte length of a single value operand and a multi-value operand are specified by the P-DOMAIN command in the display control command set.

The MVI set is shown in Table B.3-13.

Table B.3-13/T.101 – MVI Set

					10	11	12	13	14	15
					b ₇					
					b ₆					
					b ₅					
b ₄	b ₃	b ₂	b ₁		2	3	4	5	6	7
0	0	0	0	0	MVI START		NUMERIC DATA			
0	0	0	1	1	MVI END					
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		MOVE				
1	0	1	1	11						
1	1	0	0	12		REPEAT START				
1	1	0	1	13		REPEAT END				
1	1	1	0	14						
1	1	1	1	15						

B.3.9.1 MVI START

MVI START indicates that all characters between MVI START and MVI END should be stored in the MVI buffer memory (see Figure B.3-20). MVI START has no operand.

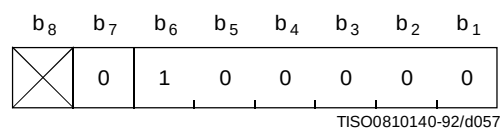


Figure B.3-20/T.101 – MVI START

B.3.9.2 MVI END

MVI END terminates storing commands, then activates the execution of the stored commands (see Figure B.3-21). MVI-END has no operand. When all stored characters are executed, the MVI buffer is cleared.

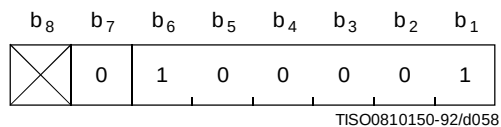


Figure B.3-21/T.101 – MVI END

B.3.9.3 REPEAT START

REPEAT START indicates that all characters between REPEAT START and REPEAT END are repeatedly executed. The number of repetition is specified by the REPEAT START operand (see Figure B.3-22).

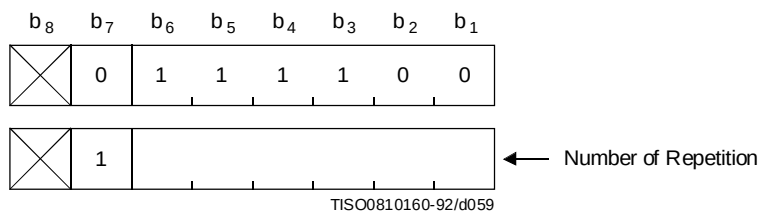


Figure B.3-22/T.101 – REPEAT START

B.3.9.4 REPEAT END

REPEAT END terminates the character strings to be repeated. REPEAT END has no operand (see Figure B.3-23).

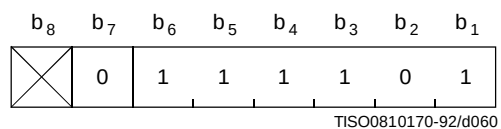


Figure B.3-23/T.101 – REPEAT END

B.3.9.5 MOVE

MOVE takes two multi-value operands and a following single-value operand (see Figure B.3-24). The first multi-value operand specifies the originating point coordinates on the frame assigned by ASSIGN FRAME. The second multi-value operand specifies the terminating point coordinates. The specified frame moves from the originating point to the terminating point. The single-value operand immediately following the second multi-value operand indicates time period in the unit of one tenth second for which the frame moves the originating point to the terminating point.

The originating point coordinates represent the lower left corner coordinates of the frame specified by ASSIGN FRAME with respect to the unit screen. The terminating point coordinates are specified in the relative coordinates with reference to the originating point coordinates. For movement description, the unit screen boundary is expanded to -1.0 for both the X direction and Y direction.

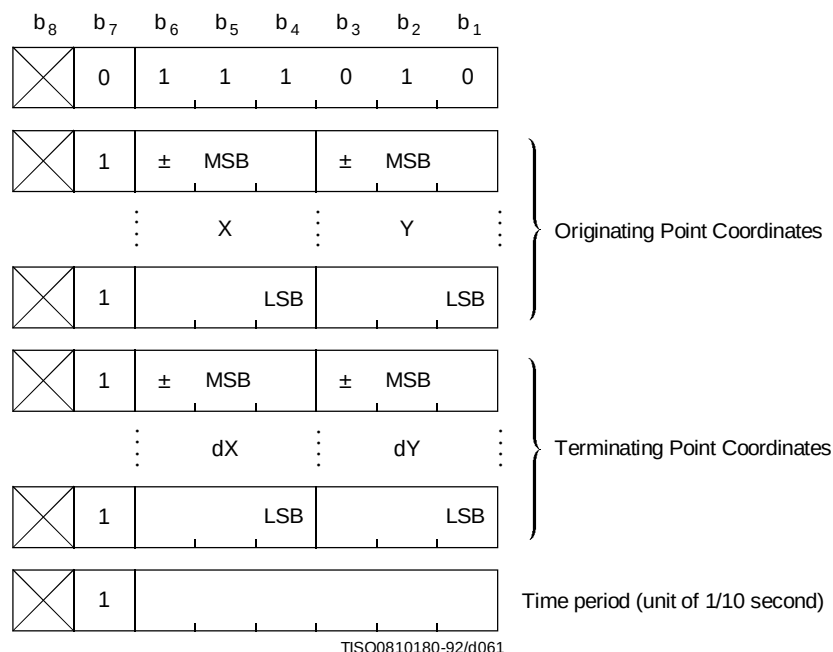


Figure B.3-24/T.101 – MOVE

B.4 Coding in the transparent mode

B.4.1 Photographic Data Unit (PDU)

A photographic data unit (PDU) is composed of an opcode with a length indicator (LI), followed by zero, one or more operands, each of which consists of one or more octets of bit sequences.

The use of all octet patterns is allowed in the operands bit sequences, which results in efficient expression of arbitrary data. The PDU format is shown in Figure B.4-1.

The opcode consists of a single octet which indicates the meaning of photographic data contained in the PDU.

The LI consists of one or more octet. The value of an LI is a binary number that represents the total length of operands following the LI field in octet. If the decimal expression of the second octet is 255, then the next two octets (the third and the fourth) indicate the real length. Since the LI simply indicates the data length, it is not shown in the figure of the PDU format throughout this clause.

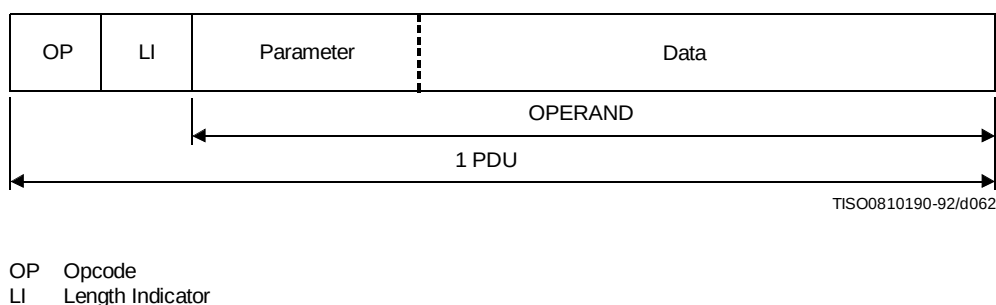


Figure B.4-1/T.101 – PDU Format

One or more octets of parameters are located at the leading part of an operand field. Parameters include the drawing point coordinates where the photographic data should be displayed, and/or the packing format which indicates the way in which photographic data are arranged. Photographic data expressed on a dot-by-dot basis are contained in the remaining part of the operand field.

Here, the dot corresponds to the logical pel or the colouring block in the unit screen. The pel position or the pel coordinate is defined as one of the four corner positions of the logical pel, according to the signs of the width (dx_0) and height (dy_0) of the logical pel. The colouring block position is defined in the same manner as the pel position provided that the word “logical pel” is replaced by the “colouring block”.

There are three types of parameter operands: fixed format, single-value and multi-value. The length of single-value or multi-value parameter operands are determined by the P-DOMAIN command.

PDU opcodes are shown in Table B.4-1. Blanks in the table are reserved for opcodes for dot-by-dot colouring (e.g. PCM, DPCM and transform coding), audio, telesoftware, etc.

B.4.2 LINE DOT PATTERN

The LINE DOT PATTERN PDU format is shown in Figure B.4-2. Photographic data express the values of logical pels, 1s and 0s, which should be written along a horizontal line on the pattern plane.

A single-value parameter operand indicates the absolute coordinate where the data should be written.

Correspondence between the photographic data and logical pel positions in the pattern plane is shown in Figure B.4-3.

The first bit P_0 is deposited at the leftmost logical pel position, next bit P_1 to the right next pel position. When the next bit is received, the pel position moves to the right by one pel’s size. If the bit P_n corresponds to the rightmost logical pel, the pel position corresponding to the next bit P_{n+1} should be moved to the leftmost pel position a logical pel height below.

If a bit corresponds to the lower right corner pel position, and if there are still remaining bits, they are discarded.

B.4.3 LINE DOT PATTERN COMPRESSED

This PDU format is shown in Figure B.4-4. Photographic data express the run length coded patterns of the logical pel values which should be written in the same manner as the LINE DOT PATTERN.

The single-value parameter operand indicates the absolute coordinate Y where the data is written.

Data compression is achieved by using the Modified Huffman (MH) run length coding technique. The coding process begins with 0’s run, then alternately continues for 1’s and 0’s runs. The last run in a line is coded as an end of line code (EOL) regardless of its run length, since the last run continues to the end of line.

Bit sequence following EOL should be decoded for logical pels one logical pel height below. The coding process starts with 0’s run and continues in the same manner as the first line.

The MH code table is shown in Tables B.4-2 and B.4-3. A coding example is shown in Figure B.4-5.

If more data are received after the pixel position reaches the lower screen boundary, they are discarded.

Table B.4-1/T.101 – PDU OP Codes

					b _x	0	0	0	0	0	0	8 ~ 15
					b ₇	0	0	1	1	1	1	
					b ₆	1	1	0	0	1	1	
					b ₅	0	1	0	1	0	1	
b ₄	b ₃	b ₂	b ₁			2	3	4	5	6	7	Reserved
0	0	0	0	0	Line Dot Pattern (comp.)	Line Dot Pattern	Coloring Block (comp.)	Coloring Block	Tele-software	Pulse code Modulation (PMC)		
0	0	0	1	1								
0	0	1	0	2	Field Dot Pattern (comp.)	Field Dot Pattern	Field Coloring Block (comp.)	Field Coloring Block		Adaptive Block Truncation (ABTC)		
0	0	1	1	3								
0	1	0	0	4			Free Format Coloring Block			Adaptive Discrete Cosine Transform (ADCT)		
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	Photo DRCS 1							
1	1	0	1	13	Photo DRCS 2							
1	1	1	0	14								
1	1	1	1	15								

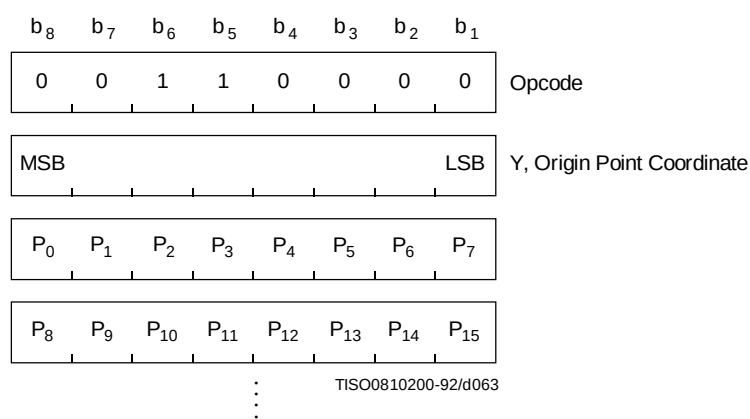


Figure B.4-2/T.101 – LINE DOT PATTERN

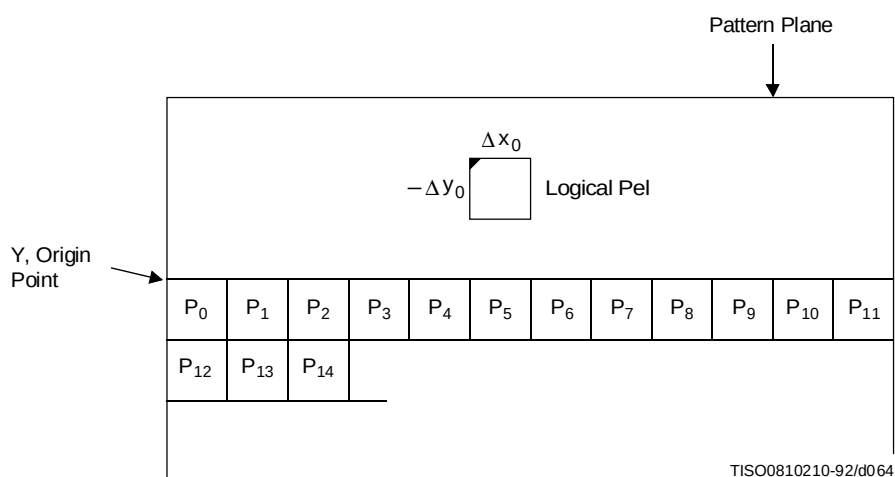


Figure B.4-3/T.101 – Correspondence between the Photographic Data and Logical Pel Positions in the Pattern Plane

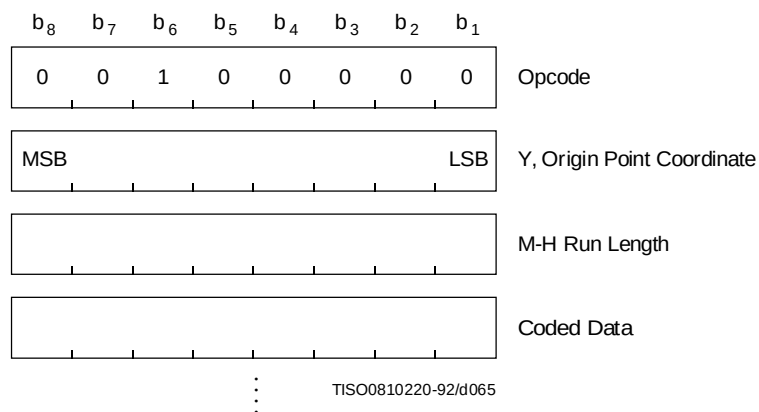


Figure B.4-4/T.101 – LINE DOT PATTERN COMPRESSED

**Table B.4-2/T.101 – MH Run Length Code Table
(Part 1: 0's run)**

	Terminating Code		Make up Code
0	00001	32	1110
1	0100	64	11011
2	0111	96	111100
3	1010	128	111110
4	1100	160	1111110
5	00100	192	00111110
6	00110	224	00111111
7	01100	256	01010110
8	01101	288	01010111
9	000100	320	01011110
10	000110	352	01011111
11	000111	384	10111110
12	001010	416	10111111
13	001110	448	11111110
14	010100	480	11111111
15	10110		
16	11010		
17	0000010		
18	0000011		
19	0001011		
20	0010110		
21	0010111		
22	0011110		
23	010110		
24	101110		
25	0101010		
26	0101110		
27	00010100		
28	1011110		
29	00010101		
30	1111010	EOL	100
31	1111011	Reserved	000000xx - - -

Table B.4-3/T.101 – MH Run Length Code Table
(Part 2: 1's run)

	Terminating Code		Make Up Code
0	00000100	32	101100
1	01	64	10101010
2	11	96	10101011
3	100	128	10101100
4	0001	160	10101101
5	00100	192	10101110
6	10100	224	10101111
7	001010	256	10110100
8	001100	288	10110101
9	0000100	320	10110110
10	0011111	352	10110111
11	1010100	384	10111100
12	00000101	416	10111101
13	00000110	448	10111110
14	00000111	480	10111111
15	00001010		
16	001110		
17	00001011		
18	00001100		
19	00001101		
20	00001110		
21	00001111		
22	00101100		
23	00101101		
24	00101110		
25	00101111		
26	00110100		
27	00110101		
28	00110110		
29	00110111		
30	00111100	EOL	101110
31	00111101	Reserved	000000xx - - -

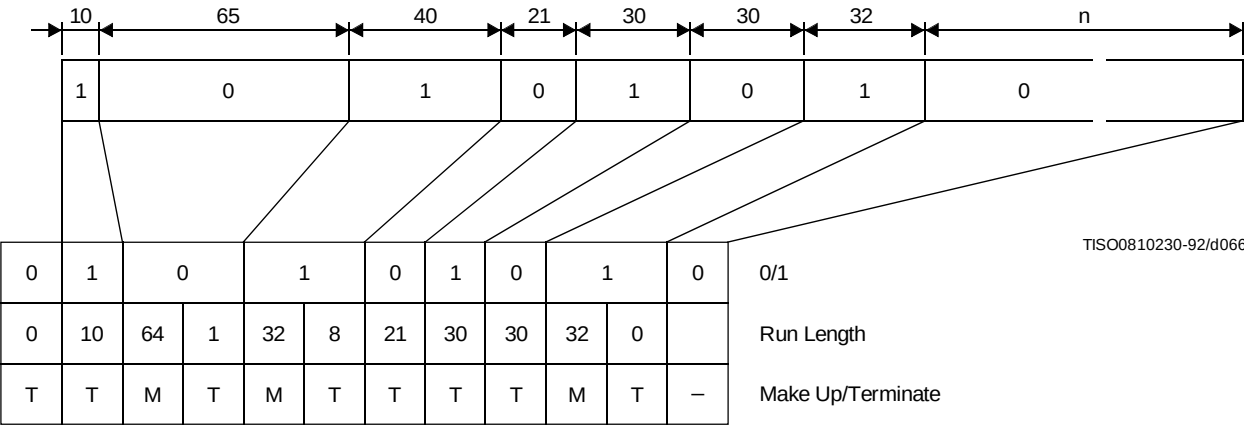


Figure B.4-5/T.101 – A Coding Example of LINE DOT PATTERN COMPRESSED

B.4.4 FIELD DOT PATTERN

The FIELD DOT PATTERN PDU format is shown in Figure B.4-6. Photographic data express the pel values, in a rectangular field on the pattern plane.

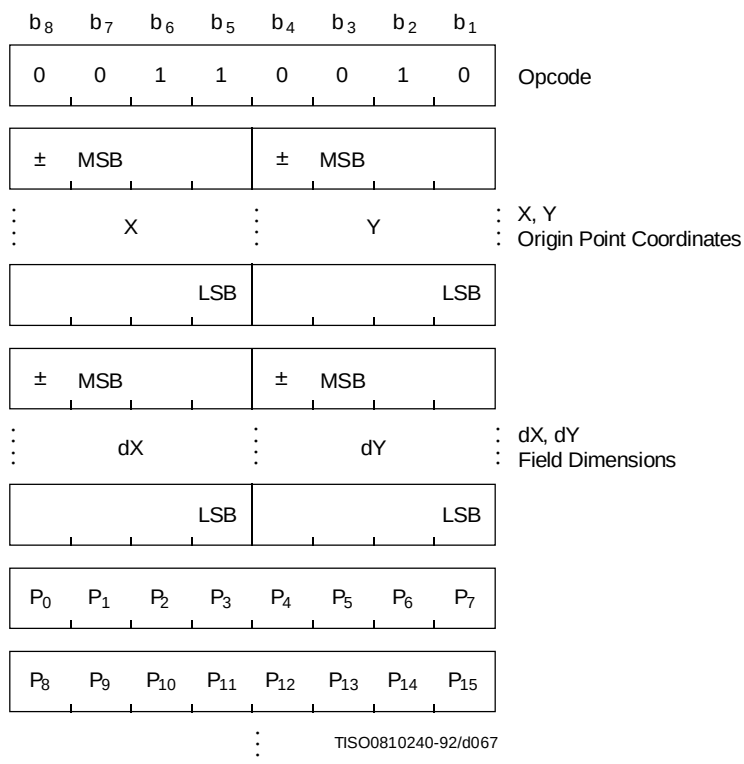


Figure B.4-6/T.101 – FIELD DOT PATTERN

There are two multi-value parameters operands which define the rectangular field location and dimensions in which the photographic data should be deposited.

The first multi-value operand specifies the initial logical pel position in absolute coordinates. The width and height (dx, dy) are given by the second multi-value operand. The initial pel position may be one of four vertices of the field since the width and height may be positive or negative.

Correspondence between the data and the logical pels in the pattern plane is shown in Figure B.4-7. After a bit in the photographic data sequence is deposited at the current logical pel, the pel position is automatically moved in the x direction a distance equal to the width (dx_0) of the logical pel. The next bit is deposited at this position. If the field dimension dx is positive, position moves to the right and if dx is negative, position moves to the left.

If the pel position reaches or exceeds a vertical side in the field, x coordinate of the pel is moved to the other side of the field and the y coordinate is moved by the height (dy_0) of the logical pel. If the field dimension dy is positive, the pel position moves up, and if dy is negative, the position moves down.

If the pel position reaches or exceeds a horizontal side of the rectangle, and if there are still remaining bits, they should be interpreted as if there proceeds the same opcode with field parameter operands ($X + dx$, Y), (dx, dy).

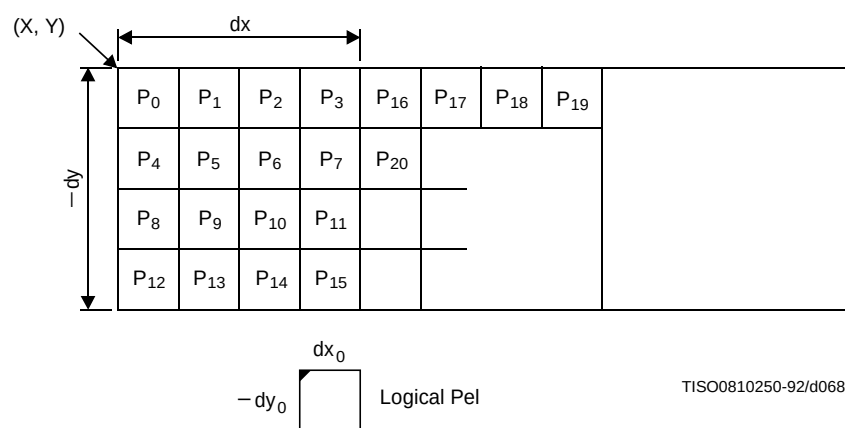


Figure B.4-7/T.101 – Correspondence between the Photographic Data and Logical Pel Positions

B.4.5 FIELD DOT PATTERN COMPRESSED

The FIELD DOT PATTERN COMPRESSED PDU format is shown in Figure B.4-8. Photographic data express the run length coded patterns of the logical pel values which should be written in the same manner as the FIELD DOT PATTERN.

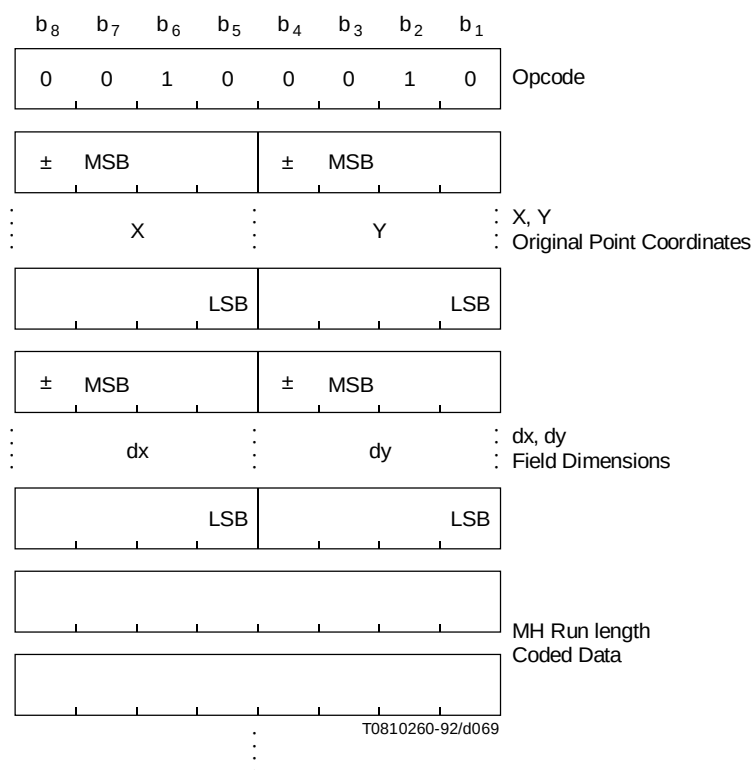


Figure B.4-8/T.101 – Field dot pattern compressed

Data compression achieved by using the Modified Haffman (MH) run length coding technique. The MH code table is shown in Tables B.4-2 and B.4-3.

There are two multi-value parameters operands which define the rectangular field location and dimensions in which the photographic data should be deposited.

The first multi-value operand specifies the initial logical pel position in absolute coordinates. The width and height (dx, dy) are given by the second multi-value operand. The initial pel position may be one of four vertices of the field since the width and height may be positive or negative.

In a field, the coding process begins with 0's run, then alternately continues for 1's runs and 0' runs. The last run in a field is coded as an end of line code (EOL) regardless of its runlength. If the run length is more than 511 (= 480 + 31), the dummy code of which length is 0 is inserted, then the coding process alternately continues for 1's runs and 0's runs.

If the field dimension dx is positive, position moves to the right and if the dx is negative, position moves to the left.

If the pel position reaches or exceeds a vertical side in the field, x coordinate of the pel is moved to the other side of the field and the y coordinate is moved by the height (dx₀) of the local pel. If the field dimension dy is positive, the pel position moves up, and if dy is negative, the position moves down.

When EOL is recognized, the following data should be interpreted as if it proceeds the same opcode with field parameter operands (X + dx, Y), (dx, dy). If the position exceeds the pattern plane, then the data should be discarded.

B.4.6 COLOURING BLOCK

The COLOURING BLOCK PDU format is shown in Figure B.4-9. Photographic data express the colour and attributes of the colouring blocks in a horizontal line on the FG, BG and display attribute planes.

The first fixed format parameter operand indicates the colour and attribute flags. If the flag of the FG colour, BG colour or display attribute is "1", the PDU contains the corresponding data.

The second parameter, a single-value operand, indicates the absolute coordinate Y where the attribute data decoding is initiated.

Photographic data are composed of three successive groupings of bit codes which represent FG data, BG data and display attributes. A four-bit for FG data or BG data is an entry address of the LUT. Four-bit display attributes are used for flashing controls, and their meaning is shown in Table B.4-4. When the first operand indicates all flags are 1s, FG colour data for the 1st line come first, BG colour data for the same line come next and then display attribute data for the same line follow. If there remain more data, they are interpreted as the data for colouring blocks in a line which is one colouring block height below. The remaining data are interpreted in the same manner as the first line.

Any data boundary in operand bytes (between FG colour data and BG colour data, between BG colour and display attributes, and between a line and a line below) keeps byte boundary, i.e. if last data occupy only a portion of a byte, remaining bits in the byte are stuffed with 0 bits.

If operand data overrun the lowest block boundary, they are ignored.

B.4.7 COLOURING BLOCK COMPRESSED

The COLOURING BLOCK COMPRESSED PDU format is shown in Figure B.4-10. Photographic data express the colour and attribute of the colouring blocks in a horizontal line in the FG, BG and display attribute plans.

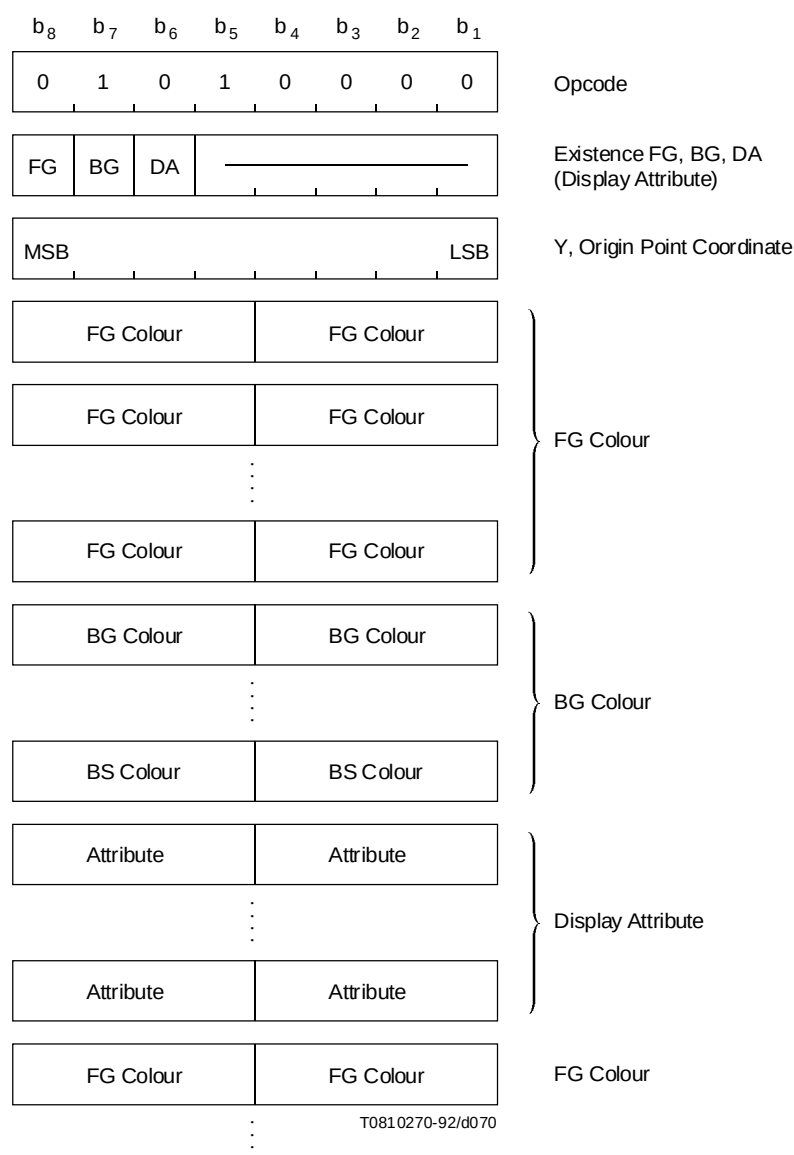


Figure B.4-9/T.101 – COLOURING BLOCK

The first fixed format parameter operand indicates the attribute flags. If the flag of the FG colour, BG colour or display attribute is “1”, the PDU contains the corresponding data.

The second parameter, a single-value operand, indicates the absolute coordinate Y where the attribute data decoding is initiated.

Photographic data are composed of attribute-run code pairs.

In an attribute-run pair, four bits are assigned for both attribute and run. A four-bit for FG or BG planes is an entry address of the LUT. Four-bit attributes for display attribute planes are used for flashing controls, and their meaning is shown in Table B.4-4. The four-bit run length code table is shown in Table B.4-5. The last run in a horizontal line is coded as ECL (End of Colouring Line) regardless of its length.

Table B.4-4/T.101 – Display attribute code

Code	Flashing	Phase	Pattern
0000	ON	2	 BG FG
0001	ON	3	
0010	ON	3	
0011	ON	3	
0100	ON	3	
0101	ON	3	
0110	ON	3	
0111	ON	2	
1111	OFF	—	—

T0810280-92/d071

If the flags of the fixed format operand are all 1s, FG colour-run pairs for the first line come first, BG colour-run pairs for the same line come next and display attribute-run pairs for the same line follow. If there remain more data, they are interpreted as the data for colouring blocks in a line a colouring block height below. The remaining data are interpreted in the same manner as the first line.

If there is flag '0' in the fixed format operand, the data corresponding to it is omitted from the expression.

A coding example is shown in Figure B.4-11.

B.4.8 FIELD COLOURING BLOCK

The PDU format is shown in Figure B.4-12. Photographic data express colours and display attributes of the colouring blocks in a field on the FG, BG and display attribute planes.

The first operand (fixed format) indicates the attribute flag in the same manner as the COLOURING BLOCK command. The second and the third are multi-value operands, which specify the location and the dimensions of a rectangular field to which attribute data should be deposited. The initial colouring block position is specified in absolute coordinates by the first multi-value operand, and the width and height (dx, dy) are given by the second multi-value operand. The initial colouring block position may be one of four vertices of the field, since the width and height may be positive or negative.

Photographic data are composed of successive three-four bit codes which represent FG data, BG data and display attributes. A four-bit for FG data or BG data is an entry address of the LUT. Four-bit display attributes are used for flashing controls, and their meaning is shown in Table B.4-14.

When the first operand indicates all flags are 1s, FG colour data for the field come first, BG colour data for the same field come next and display attribute data for the same field follow.

When the width (dx) of a field is positive, the drawing position moves to the right next pixel position after data is deposited to a pixel. When dx is negative, the drawing position moves to the left.

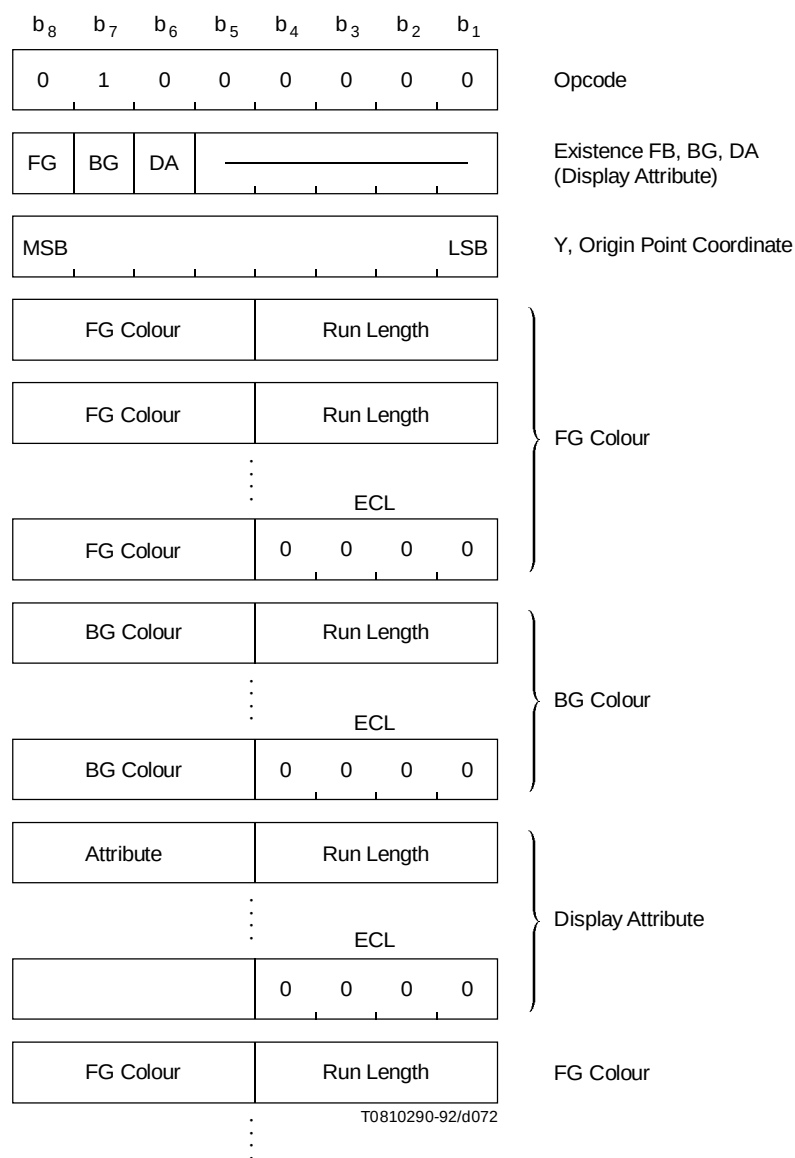
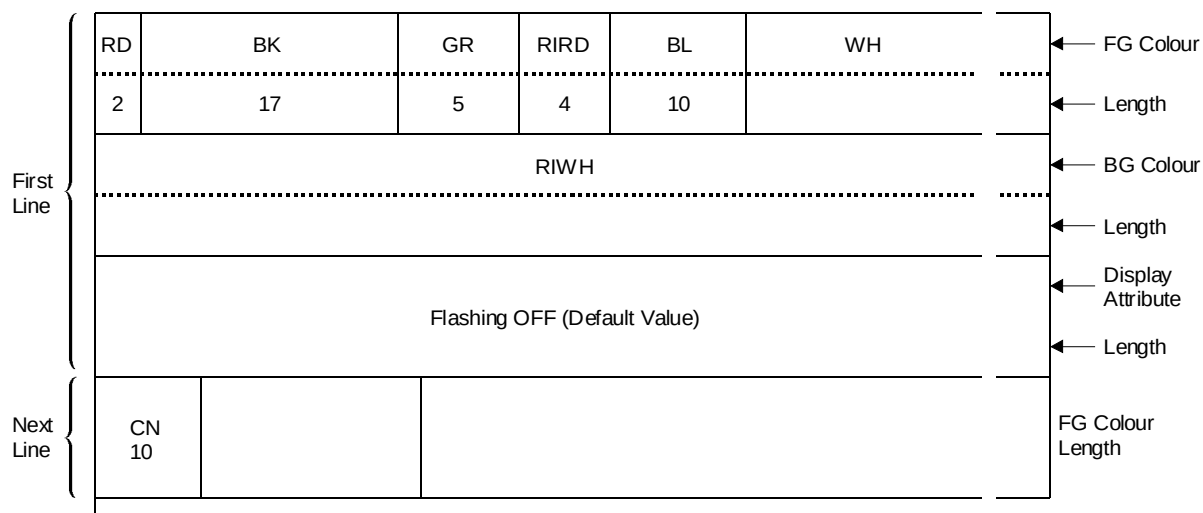


Figure B.4-10/T.101 – COLOURING BLOCK COMPRESSED

Table B.4-5/T.101 – 4-bit Run Length Code

Run Length	Code	Run Length	Code
1	0001	9	1001
2	0010	10	1010
3	0011	11	1011
4	0100	12	1100
5	0101	13	1101
6	0110	14	1110
7	0111	15	1111
8	1000	ECL	0000



1	1	0					FG/BG	
RD			0	0	1	0	2	FG Colour
BK			1	1	1	1	15	
BK			0	0	1	0	2	
GR			0	1	1	0	5	
RIRD			0	1	0	0	4	
BL			1	0	1	0	10	
WH			0	0	0	0	ECL	
RIWH			0	0	0	0	ECL	BG Colour
CN			1	0	1	0	10	FG Colour

T0810300-92/d073

Figure B.4-11/T.101 – A coding example of COLOURING BLOCK COMPRESSED

When the height (dy) of a field is positive, the drawing position moves to the upper next line, after an entire line within a field is filled. When dy is negative, the drawing position moves to the lower next line. If data overrun the defined field, following data are filled into the field whose parameters are (X + dx, Y) and (dx, dy).

If the field exceeds the block colouring plane boundary, the remaining data are discarded.

Operand data should keep byte boundary in the operand field.

B.4.9 FIELD COLOURING BLOCK COMPRESSED

The PDU format is shown in Figure B.4-13. Photographic data express colours and attributes of the colouring blocks in a field on the FG, BG and display attribute planes.

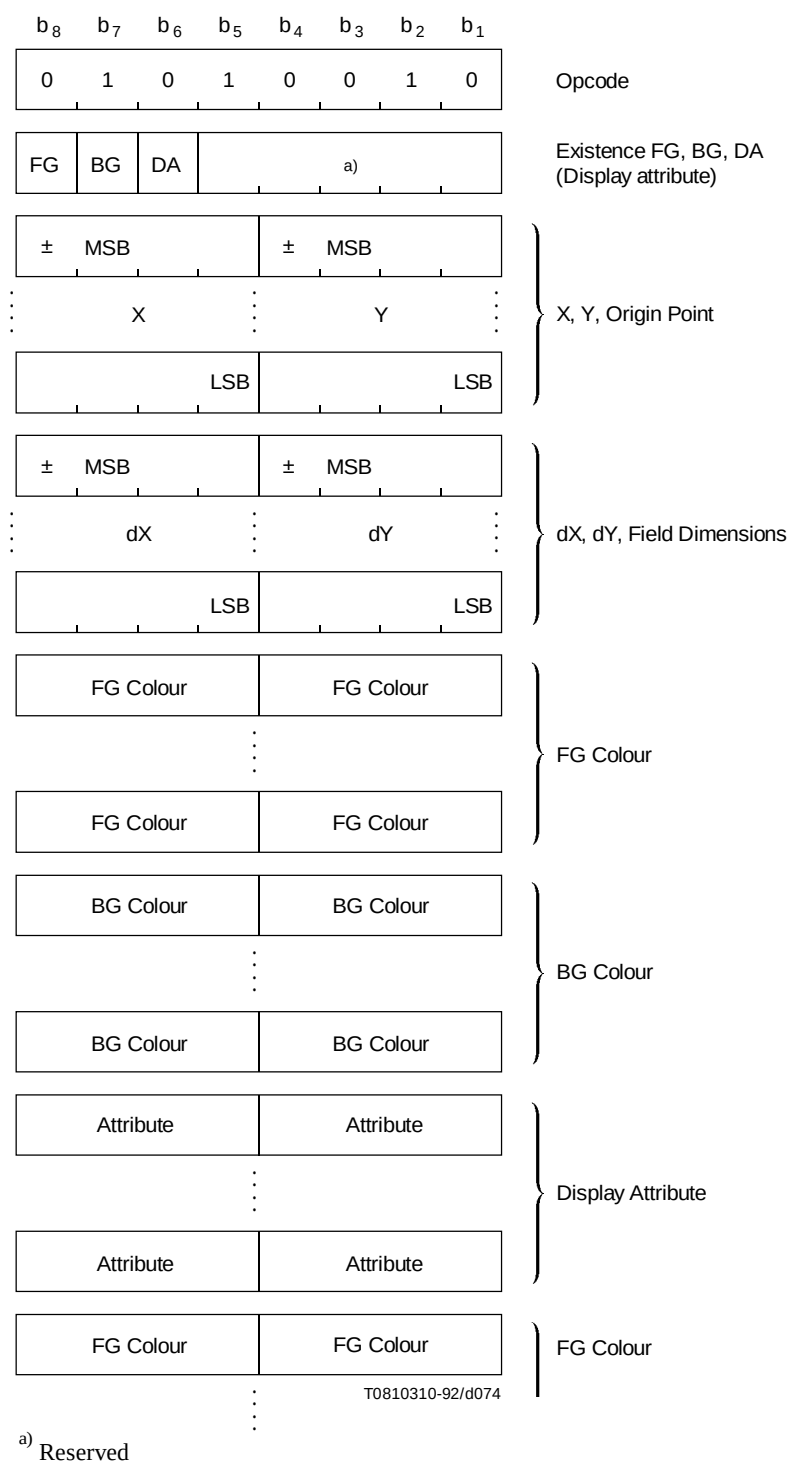


Figure B.4-12/T.101 – FIELD COLOURING BLOCK

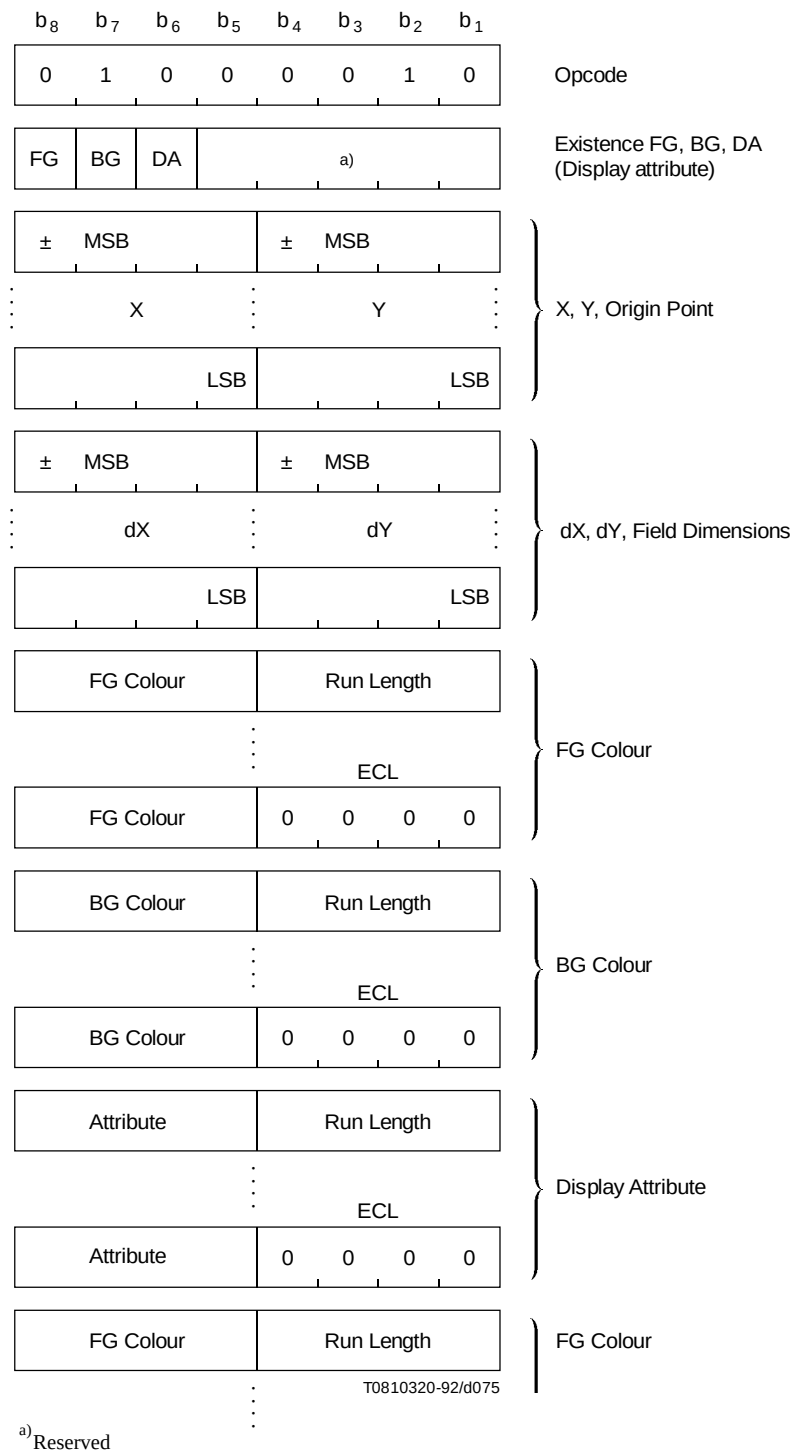


Figure B.4-13/T.101 – FIELD COLOURING BLOCK COMPRESSED

The first operand (fixes format) indicates the attribute flag in the same manner as the COLOURING BLOCK.

The second and the third are multi-value operand, which specify a rectangular field location and dimensions of the colouring blocks to which attribute data should be deposited. The initial colouring block position is specified in absolute coordinates by the first multi-value operand, and the width and height (dx, dy) are given by the second multi-value operand. The initial colouring block position may be one of four vertices of the field, since the width and height may be positive or negative.

Photographic data following the parameters are composed of attribute-run code pairs, the number of bits of each part is eight (4 bits-4 bits pair), in the same manner as the COLOURING BLOCK COMPRESSED. The last run in the defined field is coded as ECL.

The colouring block position is automatically moved in the X direction a distance equal to the width (dx₀) of the colouring block. If the field dimension dx is positive, position moves to the right and if dx is negative, position moves to the left.

If the block position reaches or exceeds a vertical side of the field, x coordinate of the colouring block is moved to the other side of the field and the y coordinate is moved by the height (dy) of the colouring block. If the field dimension dy is negative, the block position moves down.

If the colouring block position reaches or exceeds a horizontal side of the field, and if there are remaining photographic data, they should be interpreted as if there proceeds the same opcode with field parameter operands (x + dx, y), (dx, dy).

B.4.10 FREE FORMAT COLOURING BLOCK

The PDU format is shown in Figure B.4-14. Photographic data is expressed in the same manner as in the FIELD COLOURING BLOCK COMPRESSED, except the bit assignment of an attribute-run code part.

The first fixed format parameter operand indicates the content of photographic data, FG, BG, DA code length and run length expression. The bits and their meanings are shown in Table B.4-6.

The run length code specified by the first parameter operand is shown in Table B.4-7.

B.4.11 PHOTO DRCS 1 AND PHOTO DRCS 2

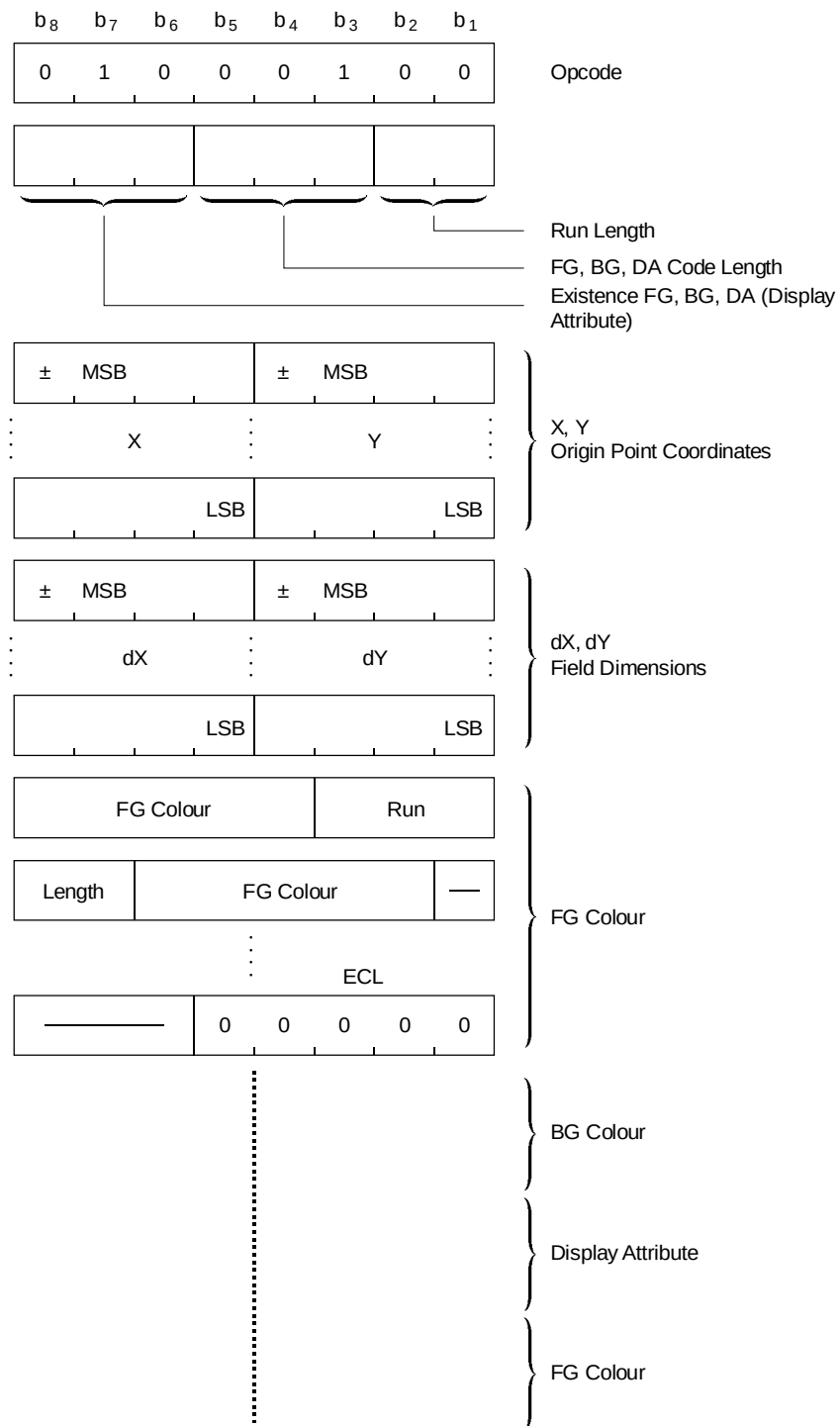
These PDU formats are shown in Figure B.4-15. Photographic data express the pixel pattern of the character to be defined. The operand is 3 bytes fixed format. The first fixed format parameter operand indicates the number of pixels in a character block for both horizontal and vertical directions. The codes are as follows:

7/3: 16 × 24 pixels (Normal size);

7/8: 8 × 24 pixels (Medium size);

7/1: 8 × 12 pixels (Small size).

The second byte is fixed to 4/1 for DRCS 1, or it represents first byte code for DRCS 2. The third byte is DRCS code for DRCS 1 or second byte code for DRCS 2.



T0810330-92/d076

Figure B.4-14/T.101 – FREE FORMAT COLOURING BLOCK

**Table B.4-6/T.101 – Fixed Format Byte Interpretation of
FREE FORMAT COLOURING BLOCK**

b ₈	b ₇	b ₆	Data Contents
0	0	1	DA (Display Attributes)
0	1	0	BG
0	1	1	BG and DA
1	0	0	FG
1	0	1	FG and DA
1	1	0	FG and BG
1	1	1	FG, BG and DA
b ₅	b ₄	b ₃	FG/BG/DA Code Length in Bits
0	0	0	3
0	0	1	4
0	1	0	5
0	1	1	6
1	0	0	7
1	0	1	8
1	1	0	9
1	1	1	10
b ₂	b ₁		Run Length Code's Length in Bits
0	0		5
0	1		6
1	0		7
1	1		8

**Table B.4-7/T.101 – Run Length Code for FREE FORMAT
COLOURING BLOCK**

Code	Run Length
00 ----- 00	ECL
00 ----- 01	1
00 ----- 10	2
00 ----- 11	3
—	—
—	—
—	—

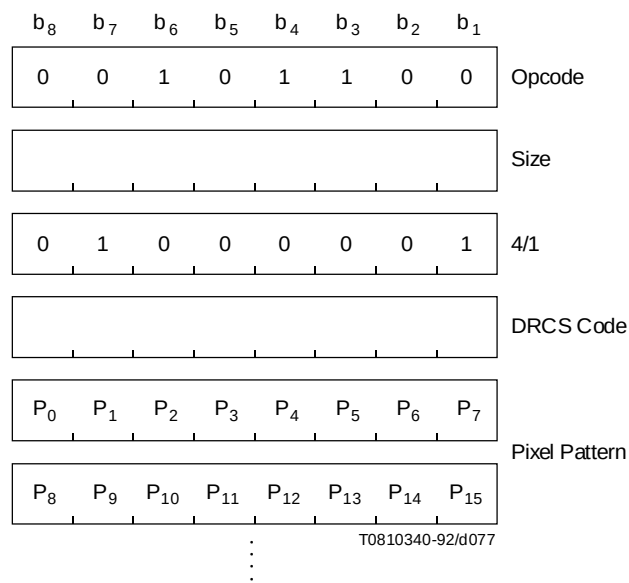


Figure B.4-15(1)/T.101 – PHOTO DRCS 1

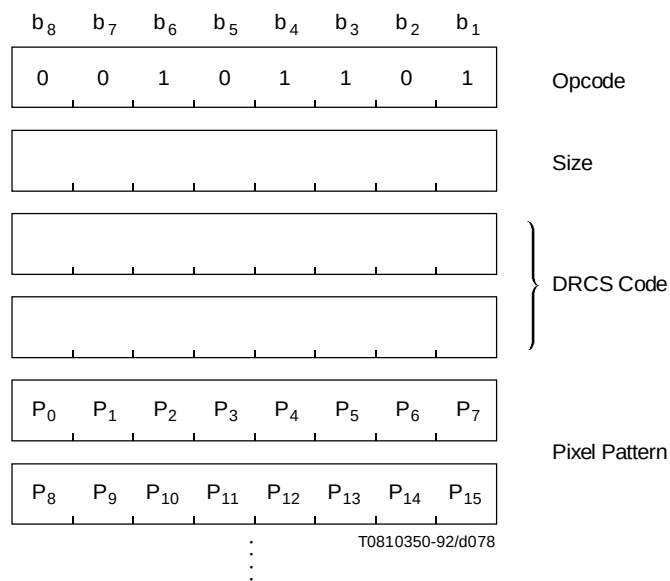


Figure B.4-15(2)/T.101 – PHOTO DRCS 2

B.4.12 Telesoftware

The telesoftware PDU format is shown in Figure B.4-16.

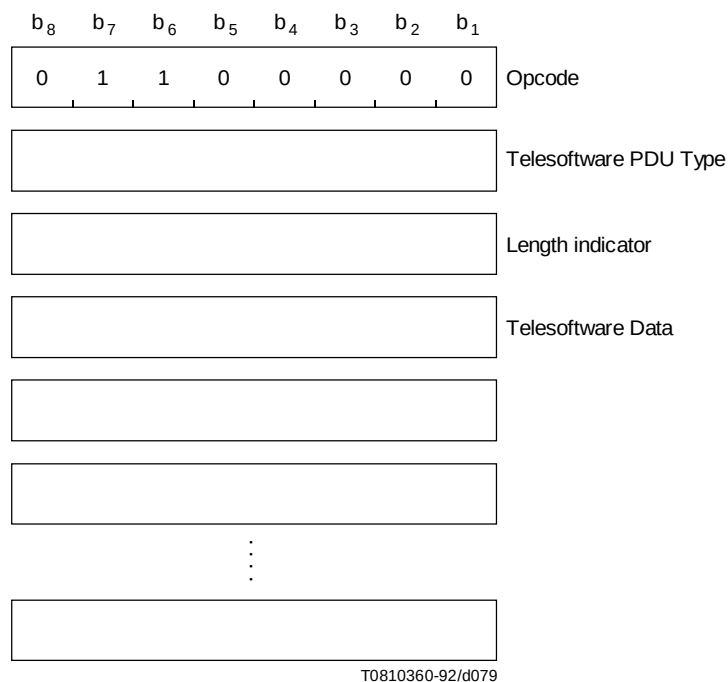


Figure B.4-16/T.101 – Telesoftware PDU Type

The data part is the telesoftware data in itself.

The first fixed format parameter operand shows the telesoftware PDU types explained in Table B.4-8.

Table B.4-8/T.101 – Telesoftware PDU type

b ₈	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	
0	0	0	0	0	0	0	0	First PDU of the telesoftware data
0	0	0	0	0	0	0	1	Intermediate PDU of the telesoftware data
1	1	1	1	1	1	1	1	Last PDU of the telesoftware data
0	0	0	0	0	0	1	0	Reserved
		:		:				
1	1	1	1	1	1	1	0	

Usually, the length of telesoftware data is longer than that of picture data.

The effect of transmission bit error on telesoftware data is severe than that on picture data.

It is danger to transmit the telesoftware data in one PDU, so the telesoftware data is separated to several PDUs.

The second fixed format parameter operand shows the length of the separated telesoftware data.

B.4.13 PCM photography

The PCM (Pulse Code Modulation) PDU format is shown in Figure B.4-17. Photographic data express the pel values in a rectangular field of the frame.

The first fixed format parameter operand which is composed of three bytes indicates the five parameters for PCM coding scheme explained in Table B.4-9.

The first parameter shows the display component, and the second shows the transmitting sequence of each component.

The third, fourth and fifth parameters show the quantizer bit number of component 1, 2, 3 respectively.

Multi-value parameters operands define the rectangular field location and dimensions in which the photographic data should be deposited.

The first multi-value operands specifies the initial logical pel position in absolute coordinates. The width and height (dX, dY) are given by the second multi-value operand.

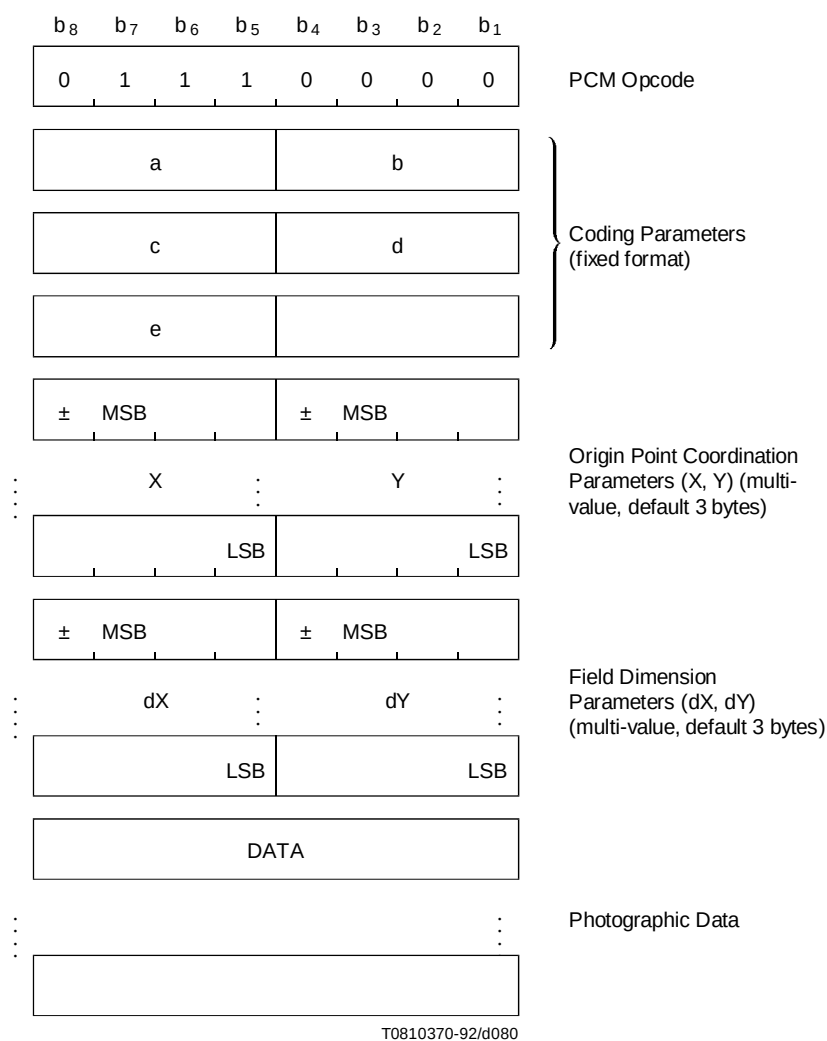


Figure B.4-17/T.101 – PCM PDU

Table B.4-9/T.101 – Pulse Code Modulation Parameters

a:	Component (logical pel size dx_0, dy_0)			
	0 0 0 0	Y(dx_0, dy_0)	(monochrome)	
	0 0 0 1	G(dx_0, dy_0)	R(dx_0, dy_0)	B(dx_0, dy_0)
	0 0 1 0	Y(dx_0, dy_0)	I(dx_0, dy_0)	Q(dx_0, dy_0)
	0 0 1 1	Y(dx_0, dy_0)	U(dx_0, dy_0)	V(dx_0, dy_0)
	0 1 0 0	Y(dx_0, dy_0)	I(2 dx_0, dy_0)	Q(2 dx_0, dy_0)
	0 1 0 1	Y(dx_0, dy_0)	U(2 dx_0, dy_0)	V(2 dx_0, dy_0)
	0 1 1 0	Y(dx_0, dy_0)	I(2 $dx_0, 2dy_0$)	Q(2 $dx_0, 2dy_0$)
	0 1 1 1	Y(dx_0, dy_0)	U(2 $dx_0, 2dy_0$)	V(2 $dx_0, 2dy_0$)
	1 0 0 0	Reserved		
	1 1 1 1			
b:	Transmitting sequence of each component			
	0 0 0 0	pel by pel		
	0 0 0 1	line by line		
	0 0 1 0	field by field		
	0 0 1 1	Reserved		
	1 1 1 1			
c:	Quantizer bit number of component 1			
	0 0 0 0	0 bit quantizer		
	0 0 0 1	1 bit quantizer		
	0 0 1 0	2 bit quantizer		

	1 1 1 1	15 bit quantizer		
d:	Quantizer bit number of component 2 same as component 1			
e:	Quantizer bit number of component 3 same as component 1			

B.4.14 ABTC photography

The ABTC (Adaptive Block Truncation Coding) PDU format is shown in Figure B.4-18. Photographic data express coded output sequences of the predefined size square blocks in a rectangular field of the frame.

The first fixed format parameter operand indicates the two parameters for ABTC scheme shown in Table B.4-10.

The first parameter shows the display component.

The second parameter shows the transmitting sequence of each component.

Two multi-value parameters operands define the rectangular field location and dimensions in which the photographic data should be deposited.

The first multi-value operands specifies the initial logical pel position in absolute coordinates. The width and height (dX, dY) are given by the second multi-value operand.

The coding algorithm of the ABTC is explained in Appendix III.

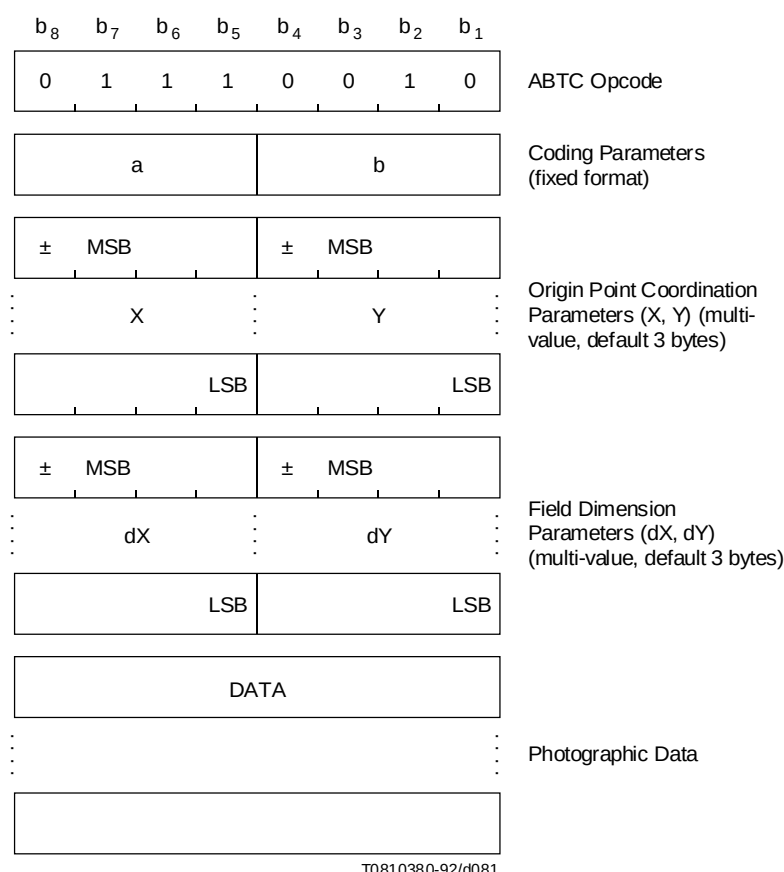


Figure B.4-18/T.101 – Adaptive Block Truncation Coding PDU

B.4.15 ADCT photography

The ADCT (Adaptive Discrete Cosine Transform) PDU format is shown in Figure B.4-19. Photographic data express coded output sequence of predefined size square blocks in a rectangular field of the frame.

The first fixed format parameter operand indicates the parameters for ADCT scheme shown in Table B.4-11.

The first parameter shows the display component, the second ~ fifth shows the sampling ratio, the sixth shows the ADCT type and the seventh shows the transmitting sequence of each component.

The eighth, ninth and tenth parameters show the quantizer bit number of component 1, 2, 3 respectively.

Multi-value operands define the rectangular field location and dimensions in which the photographic data should be deposited.

The first multi-value operands specifies the initial logical pel position in absolute coordinates. The width and height (dX, dY) are given by the second multi-value operand.

The coding of ADCT is defined in ISO DP XXXX.

Table B.4-10/T.101 – Adaptive Block Truncation Coding Parameters

a:	Component (logocal pel size dx_0, dy_0)			
	0 0 0 0	Y(dx_0, dy_0)	(monochrome)	
	0 0 0 1	G(dx_0, dy_0)	R(dx_0, dy_0)	B(dx_0, dy_0)
	0 0 1 0	Y(dx_0, dy_0)	I(dx_0, dy_0)	Q(dx_0, dy_0)
	0 0 1 1	Y(dx_0, dy_0)	U(dx_0, dy_0)	V(dx_0, dy_0)
	0 1 0 0	Y(dx_0, dy_0)	I(2 dx_0, dy_0)	Q(2 dx_0, dy_0)
	0 1 0 1	Y(dx_0, dy_0)	U(2 dx_0, dy_0)	V(2 dx_0, dy_0)
	0 1 1 0	Y(dx_0, dy_0)	I(2 $dx_0, 2dy_0$)	Q(2 $dx_0, 2dy_0$)
	0 1 1 1	Y(dx_0, dy_0)	U(2 $dx_0, 2dy_0$)	V(2 $dx_0, 2dy_0$)
	1 0 0 0	Reserved		
	1 1 1 1			
	1 1 1 1			

b:	Transmitting sequence of each component			
	0 0 0 0	block by block		
	0 0 0 1	line by line		
	0 0 1 0	field by field		
	0 0 1 1	Reserved		
	1 1 1 1			

B.5 Coding in the sound mode

B.5.1 General

Musical sound note data is conveyed in the sound mode. Although data is character coded in the sound mode, this annex specifies the mode apart from the character code mode, since information contents are different from others. A C-set and a G-set different from the character code mode are set upon invocation of the sound mode.

Figure B.5-1 illustrates the table setting in the sound mode.

B.5.2 Sound tone set

The Sound tone set is a two-bytes code set which represents pitch and duration of a sound tone. The sound tone set is shown in Table B.5-1.

The first byte of a code specifies the tone pitch, defining eighty-eight pitches from A₀ up to C₈.

The frequency of pitch A₄ is determined as 440 Hz: The frequency of each pitch is expressed as

$$f(I) = 440 \times 2^{(I - 81)/12} \text{ Hz}$$

where I is the decimal expression of the first byte.

IP (Indefined pitch) (in case the first byte is 7/15) is used for a rhythm part, which has a fixed pitch.

RST (in case the first byte is 2/0) is used to cause a rest.

The second byte determines the duration of a sound whose pitch is specified by the first byte. Sound duration is expressed with a number of Tu: Tu is a unit length of duration. The value of Tu is transmitted prior to tone codes via SMC (see B.5.3.1.1).

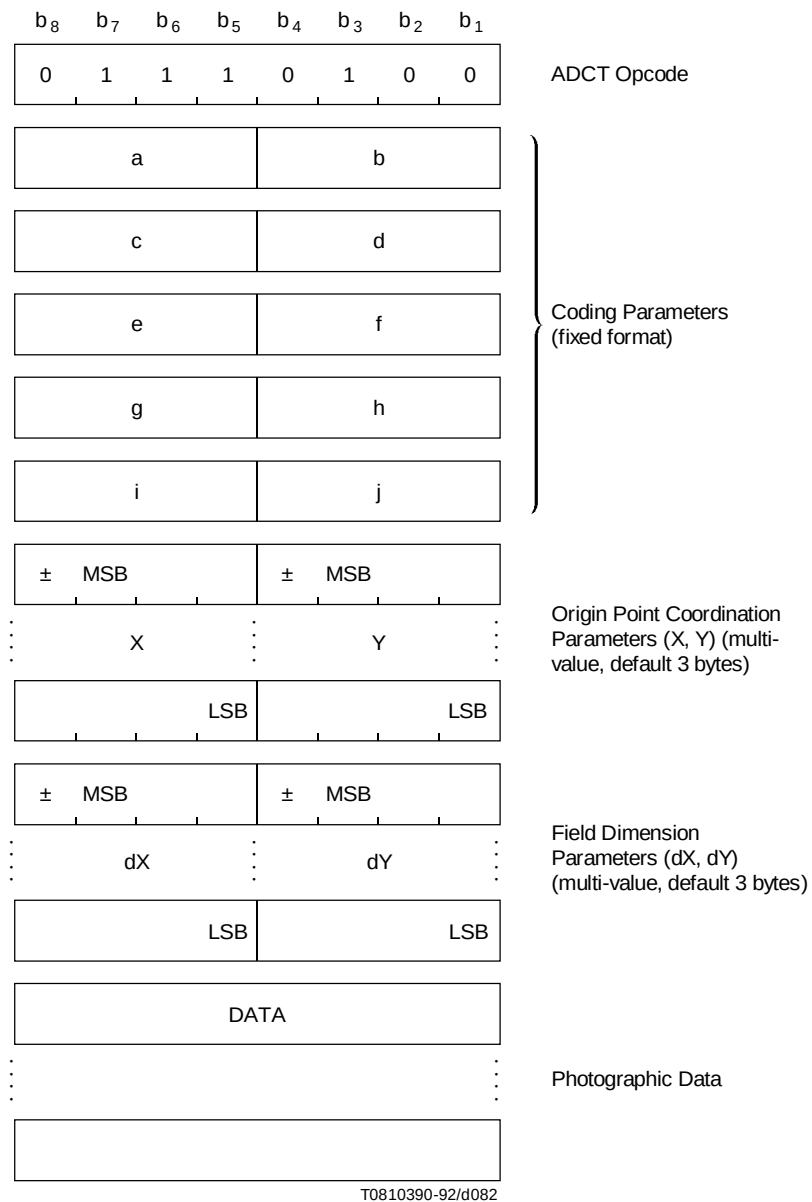
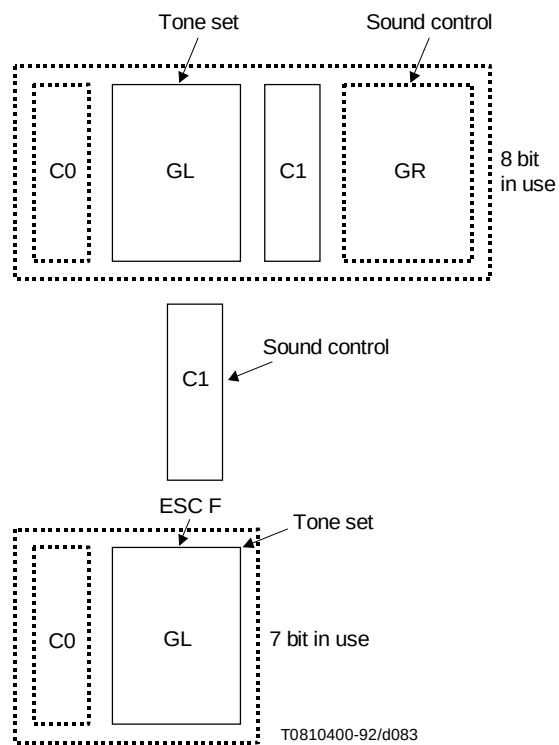


Figure B.4-19/T.101 – Adaptive Discrete Cosine Transform PDU

Table B.4-11/T.101 – Adaptive Discrete Cosine Transform Parameters

a:	Component
	0 0 0 0 Y (monochrome)
	0 0 0 1 G R B
	0 0 1 0 Y I Q
	0 0 1 1 Y U V
	0 1 0 0 } Reserved
	1 1 1 1 }
b:	Horizontal sampling ratio of component 2
	0 0 0 0 1 : 1
	0 0 0 1 1 : 2
	0 0 1 0 1 : 4
	0 0 1 1 } Reserved
	1 1 1 1 }
c:	Vertical sampling ratio of component 2 same as horizontal
d:	Horizontal sampling ratio of component 3 same as component 1
e:	Vertical sampling ratio of component 3 same as component 1
f:	A D C T Type
	0 0 0 0 Baseline
	0 0 0 1 Extended System
	0 0 1 0 Independent System
	0 1 0 0 } Reserved
	1 1 1 1 }
g:	Transmitting sequence of each component
	0 0 0 0 Block by Block
	0 0 0 1 Line by Line
	0 0 1 0 Field by Field
	0 1 0 0 } Reserved
	1 1 1 1 }
h:	Quantizer bit number of component 1
	0 0 0 0 0 bit quantizer
	0 0 0 1 1 bit quantizer
	0 0 1 0 2 bit quantizer

	1 1 1 1 15 bit quantizer
i:	Quantizer bit number of component 2 same as component 1
j:	Quantizer bit number of component 3 same as component 1



**Figure B.5-1/T.101 – Table Setting
in the Sound Mode**

Table B.5-1 (1)/T.101 – Sound Tone Set (Pitch)

					10	11	12	13	14	15
				b ₇	0	0	1	1	1	1
				b ₆	1	1	0	0	1	1
				b ₅	0	1	0	1	0	1
b ₄	b ₃	b ₂	b ₁		2	3	4	5	6	7
0	0	0	0	0	RST	C ₂	E ₃	G ₄ [#]	C ₆	E ₇
0	0	0	1	1	A ₀	C ₂ [#]	F ₃	A ₄	C ₆ [#]	F ₇
0	0	1	0	2	A ₀ [#]	D ₂	F ₃ [#]	A ₄ [#]	D ₆	F ₇ [#]
0	0	1	1	3	B ₀	D ₂ [#]	G ₃	B ₄	D ₆ [#]	G ₇
0	1	0	0	4	C ₁	E ₂	G ₃ [#]	C ₅	E ₆	G ₇ [#]
0	1	0	1	5	C ₁ [#]	F ₂	A ₃	C ₅ [#]	F ₆	A ₇
0	1	1	0	6	D ₁	F ₂ [#]	A ₃ [#]	D ₅	F ₆ [#]	A ₇ [#]
0	1	1	1	7	D ₁ [#]	G ₂	B ₃	D ₅ [#]	G ₆	B ₇
1	0	0	0	8	E ₁	G ₂ [#]	C ₄	E ₅	G ₆ [#]	C ₈
1	0	0	1	9	F ₁	A ₂	C ₄ [#]	F ₅	A ₆	
1	0	1	0	10	F ₁ [#]	A ₂ [#]	D ₄	F ₅ [#]	A ₆ [#]	
1	0	1	1	11	G ₁	B ₂	D ₄ [#]	G ₅	B ₆	
1	1	0	0	12	G ₁ [#]	C ₃	E ₄	G ₅ [#]	C ₇	
1	1	0	1	13	A ₁	C ₃ [#]	F ₄	A ₅	C ₇ [#]	
1	1	1	0	14	A ₁ [#]	D ₃	F ₄ [#]	A ₅ [#]	D ₇	
1	1	1	1	15	B ₁	D ₃ [#]	G ₄	B ₅	D ₇ [#]	IP

Table B.5-1 (2)/T.101 – Sound Tone Set (Duration)

					10	11	12	13	14	15	
					b ₇	0	0	1	1	1	1
					b ₆	1	1	0	0	1	1
					b ₅	0	1	0	1	0	1
b ₄	b ₃	b ₂	b ₁		2	3	4	5	6	7	
0	0	0	0	0	Reserved		64 Tu	16 Tu	32 Tu	48 Tu	
0	0	0	1	1			1 Tu	17 Tu	33 Tu	49 Tu	
0	0	1	0	2			2 Tu	18 Tu	34 Tu	50 Tu	
0	0	1	1	3			3 Tu	19 Tu	35 Tu	51 Tu	
0	1	0	0	4			4 Tu	20 Tu	36 Tu	52 Tu	
0	1	0	1	5			5 Tu	21 Tu	37 Tu	53 Tu	
0	1	1	0	6			6 Tu	22 Tu	38 Tu	54 Tu	
0	1	1	1	7			7 Tu	23 Tu	39 Tu	55 Tu	
1	0	0	0	8			8 Tu	24 Tu	40 Tu	56 Tu	
1	0	0	1	9			9 Tu	25 Tu	41 Tu	57 Tu	
1	0	1	0	10			10 Tu	26 Tu	42 Tu	58 Tu	
1	0	1	1	11			11 Tu	27 Tu	43 Tu	59 Tu	
1	1	0	0	12			12 Tu	28 Tu	44 Tu	60 Tu	
1	1	0	1	13			13 Tu	29 Tu	45 Tu	61 Tu	
1	1	1	0	14			14 Tu	30 Tu	46 Tu	62 Tu	
1	1	1	1	15	15 Tu	31 Tu	47 Tu	63 Tu			
Tu Time unit											
NOTE – Columns 2 to 3 are reserved for future extension.											

B.5.3 Sound control set

Twelve control codes are used for controlling sound generation. The sound control set is shown in Table B.5-2.

Table B.5-2/T.101 – Sound Control Set

					b ₈	1	1
					b ₇	0	0
					b ₆	0	0
					b ₅	0	1
						8	9
					b ₇	1	1
					b ₆	0	0
					b ₅	0	1
						4	5
b ₄	b ₃	b ₂	b ₁			4/8	5/9
0	0	0	0	0	SMC	SLV	
0	0	0	1	1	SMP		
0	0	1	0	2	SRP		
0	0	1	1	3			
0	1	0	0	4	EMC		
0	1	0	1	5	EPT		
0	1	1	0	6			
0	1	1	1	7			
1	0	0	0	8	LBL	LRT	
1	0	0	1	9	JMP		
1	0	1	0	10	RPT		
1	0	1	1	11	BRA		
1	1	0	0	12	CTM		
1	1	0	1	13			
1	1	1	0	14			
1	1	1	1	15			

B.5.3.2.2 SRP (Start of rhythm part)

SRP denotes the start of a rhythm or percussive part and specify the timbre played in this rhythm part. The timbre is defined by the parameter of SRP shown in Table B.5-4.

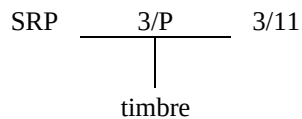


Table B.5-4/T.101 – Timbre in Rythm Part

Timbre (instrument)	Code	
Snare drum	3/1	3/0
Bass drum	3/1	3/1
Tom-tom	3/2	3/0
Suspended cymbal (top cymbal)	3/3	3/0
High-hat cymbals	3/3	3/1

B.5.3.2.3 EPT (End of part)

EPT denotes the end of a melody part or a rhythm part.

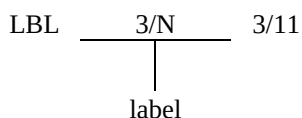
B.5.3.3 Repeat and jump

The following four codes are used to express repeat or jumps. They are followed by one or two numeric parameters, M and/or N.

M indicates a number of repeat or passing time. N does a label number such as a start point of repetition or a destination for a jump.

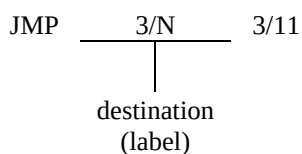
B.5.3.3.1 LBL (label)

Followed by the parameter N, LBL defines a label as the start point of a destination.



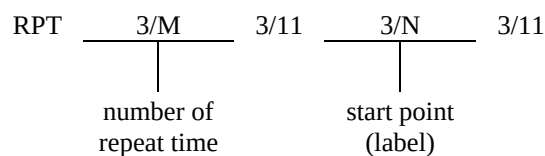
B.5.3.3.2 JMP (jump)

JMP causes jump to the part labeled with the parameter N.



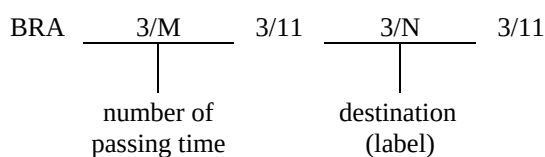
B.5.3.3.3 RPT (repeat)

Followed by two parameters M and N, RPT denotes passing after repeating M times. The start point of repetition is indicated by the second parameter N.



B.5.3.3.4 BRA (branch)

Followed by two parameters M and N, BRA denotes a jump to label N just after the Mth passing.



B.5.3.4 SLV (sound level)

The sound level (intensity) is denoted by SLV, followed by a parameter shown in Table B.5-5. The sound intensity is specified in eight levels, from level 1 to level 8.

Level 1 indicates the minimum intensity and level 8 the maximum.

The sound level differences between adjacent levels is about 3 dB.

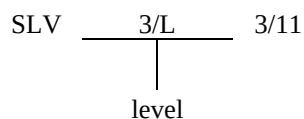


Table B.5-5/T.101 – Sound Level

Level	Code
1	3/1
2	3/2
3	3/3
4	3/4
5	3/5
6	3/6
7	3/7
8	3/8

Level 1: lowest
Level 8: highest
Level 4: default

B.5.3.5 CTM (change of timbre)

The change of timbre in a part is denoted by CTM, followed by a parameter shown in Table B.5-3.

$$\text{CTM} \quad \frac{3/P}{\text{timbre}} \quad 3/11$$

B.5.3.6 LRT (long rest and tone)

LRT is a code for denoting a long tone or a long rest, causing the duration of the next defined tone to be multiplied by N.

$$\text{LRT} \quad \frac{3/N}{\text{multiplier}} \quad 3/11$$

B.5.3.7 Numeric parameter

Numeric parameters following a sound control code are expressed in binary coded decimals. Characters 3/0 through 3/9 are used to indicate a value and 3/11 is used as a decimal delimiter.

For example, the sequence

$$\frac{3/1, 3/2, 3/3}{N_1} \quad 3/11, \quad \frac{3/4, 3/5, 3/6}{N_2} \quad 3/11$$

is interpreted as

$$N_1 = 123 \text{ (Decimal)}$$

and

$$N_2 = 456 \text{ (Decimal)}$$

B.6 Terminal identification

B.6.1 Introduction

A common enquiry function for multi-syntax videotex terminals has been developed. The basic mechanism is described in the main body of this Recommendation. This describes the terminal identification message parameters particular to the Data Syntax I terminals.

B.6.2 General structure

The Terminal Identification Command sent by the host is as follows:

$$1/15 \quad 2/0 \quad 4/0$$

It was agreed in the last meeting that the Terminal Identification Response Message from the terminals conform to the same form:

$$1/15 \quad x/y \text{ } x/y \quad 4/0$$

where x/y takes the following coding structure:

4/1-4/15	
& 6/0-7/15	= reserved for internal use of the data syntaxes (DSI, DSII, and DSIII)
5/0, 5/1	= audio mode
5/2	= modem capability
5/3	= reserved
5/4	= data syntax and profile (rank/facility) identification
data syntax	
3/1	= DS I
3/2	= DS II
3/3	= DS III
:	
:	
3/14	= private
3/15	= termination code
5/5	= photographic mode
5/6	= support of ISO 9281 switching
5/7	= support of VEMMI data
5/8-5/15	= reserved

Therefore, a response message for a terminal supporting Data Syntax I would be the following form:

1/15 2/0 5/4 3/1 x/y x/y x/y 3/15 4/0

where x/y consists of characters chosen from 4/1-4/15 and 6/0-7/15 which identify the capabilities of the Data Syntax I terminal.

B.6.3 Details for Data Syntax I terminals

In order to identify the data syntax I terminal, the following codes are assigned

4/1 = ability of data access function

6th column: rank of terminals

6/2 = rank 2

6/3 = rank 3

6/4 = rank 4

6/5 = rank 5

:

6/15 = reserved for expansion

7th column: optional functions

7/0 = melody

7/1 = simple moving pictures

7/2 = telesoftware

7/3 = color look up table

7/4 = JIS (Japanese Industrial Standard) level 2 Kanji-characters

7/5 = black and white display

:

7/15 = reserved for expansion

4/2 = ability of data input function

4/3 = method of user identification

6/0 = network ID (User IDs are provided by the network)

6/1 = terminal ID (user IDs are provided by the terminal)

Therefore, when a rank 2 terminal of Data Syntax I supports the melody function and JIS level-2 Kanji characters and its ID is provided by the videotex network, it would send the following codes to inform its capabilities to the host.

1/15 2/0 ... 5/4 3/1 4/1 6/2 7/0 7/4 4/3 6/0 3/15 ... 4/0

B.7 Default conditions

The default conditions of the table setting, the display attribute, etc. are shown below:

<i>Items</i>	<i>Default condition</i>
Command mode	Character code mode
Graphic Sets	
G0	Japanese-Kanji set
G1	Primary set
G2	Display control command set
G3	DRCS 1
In-use table	
GL	G0
GR	G2
(In 7 bit environment	G0)
Single-value operand	1 byte
Length	
Multi-value operand	3 bytes (Character code mode) 2 bytes (Transparent mode)
Logical pel sizes	(1/256, -1/256)
Colouring block sizes	(4/256, -4/256)
Character block sizes	(16/256, -24/256)
Foreground colour	White (7th colour)
Background colour	Transparent (8th colour)
LUT contents	As specified in Table B.3-5
Raster colour	Reduced intensity blue
Header Raster colour	Reduced intensity blue
Curso display	Off
Flashing	Off
Conceal	Off
Lining	Off
Protect	Off
Scroll	Off
Blinking	Off
Character size	Normal
Text rotation	0°
Character path	To the right
Inter-character spacing	1
Inter-row spacing	1
Move parameter	Move independently

<i>Items</i>	<i>Default condition</i>
Cursor Style	Underscore
Cursor position	Home position
Drawing point	(0, 0)
Line texture	Solid
Fill pattern	Solid
Hightlight	Off
Active area	The defined display area
Frame index	10
Overwriting mode	Replace
Macros	Null
DRCS 1 and 2	Null

Appendix I

(to Recommendation T.101, Annex B)

Service Reference Model (SRM)

The Service Reference Model defines the recommended features that should be implemented in an ordinary terminal. These sets of requirements also represent the maximum functionalities that the information provider should assume when encoding text and pictorial information. The SRM is intended to give measure for the implementation of a terminal or for page creation, however, it is by no means obligatory.

<i>Functions</i>	<i>Requirements</i>
1. Code extension sequence	All code extension sequences defined in this Recommendation should be executed.
2. C0 control set	The set should be executed as described in this Recommendation.
3. C1 control set	The set should be executed as described in this Recommendation, except that flashing process may be limited to two phase operation. The ON interval and the OFF interval are about 0.5 second each.
4. The graphic character repertoire	All characters of the primary set, the Japanese-Kanji set and the Katakana set should be presented.
5. Mosaic sets	All mosaics of the Mosaic 1 set and the Mosaic 2 set should be presented.
6. DRCSs	Both the DRCS 1 and the DRS2 set should be definable. The number of simultaneously definable characters is subject to memory limitation.
7. Macro set	The number of macros definable should be 96, subject to memory limitation.

<i>Functions</i>	<i>Requirements</i>
8. Memory requirements	The terminal should provide minimum 2 k bytes memory shared for both DRCS and Macros
9. Display control command set	The following commands should be implemented as specified in this Recommendation. P-RESET P-DOMAIN LOGICAL PEL DISPLAY MODE RASTER HEADER RASTER
10. Photographic commands	The following commands should be implemented as specified in this Recommendation. LINE DOT PATTERN LINE DOT PATTERN COMPRESSED FIELD DOT PATTERN COLOURING BLOCK COLOURING BLOCK COMPRESSED FIELD COLOURING BLOCK PHOTO DRCS 1 PHOTO DRCS 2
11. Physical display parameters	Resolution should be 248 pixels horizontal by 204 pixels vertical including "Header area".

Appendix II

(to Recommendation T.101, Annex B)

Operation of CS and NSR

Both CS and NSR re-establish the following attributes:

- Graphic set table setting
- Domain
- Logical pel sizes
- Colouring block sizes
- Foreground colour
- Background colour
- Cursor display
- Flashing
- Conceal
- Lining
- Protect
- Scroll
- Character block sizes
- Text display attributes

- Character size
- Drawing point
- Line texture
- Fill pattern
- Highlight
- Frame index
- Overwriting mode

The following attributes are set to default states only by CS:

- Raster colour
- Header raster colour
- Cursor position (NSR sets the cursor position according to its parameters)
- LUT contents
- Blink process
- Active area
- Raster memory contents
- DRCS definition
- MACRO definition

Appendix III

(to Recommendation T.101, Annex B)

Adaptive block truncation coding

III.1 Introduction

Adaptive block coding or adaptive block truncation coding (ABTC) is an efficient coding technique which utilizes the correlation between adjacent picture element (pixels). It involves segmenting a picture into small blocks and representing every pixel in one block by either of two gray level commonly allotted to the block. To make the coding adaptive, a variable coding algorithm depending on the local characteristics of pictures is adopted. A compression rate, nearly as high as 2.0 bits/pixel for color photography, is achieved by using this scheme.

The characteristic features of the coding scheme can be summarized as the high compression rate and the algorithm simplicity. A frame of ordinary color TV picture can be coded with 400 Kbits, and thus can be transmitted in about 6 seconds through the public digital telephone network (64 Kbits/S). The algorithm simplicity will enhance the micro-processor implementation of this coding.

III.2 Adaptive block truncation coding (ABTC) algorithm

Each component (R, G, B component or Y, I, Q component) of color images independently coded by the block truncation coding, using same algorithm. Figure III.1 shows the principle of basic block truncation coding (BTC) algorithm.

BTC is very efficient and has good quality in coding ordinary pictures, it is, however, inefficient in smooth area and has low quality in rapidly changing area. In order to gain high compression ratio and high quality in these areas, new, Adaptive BTC (ABTC) has been proposed. In this ABTC, following three coding modes are prepared and coding mode and block size are changed adaptively according to the locality of pictures.

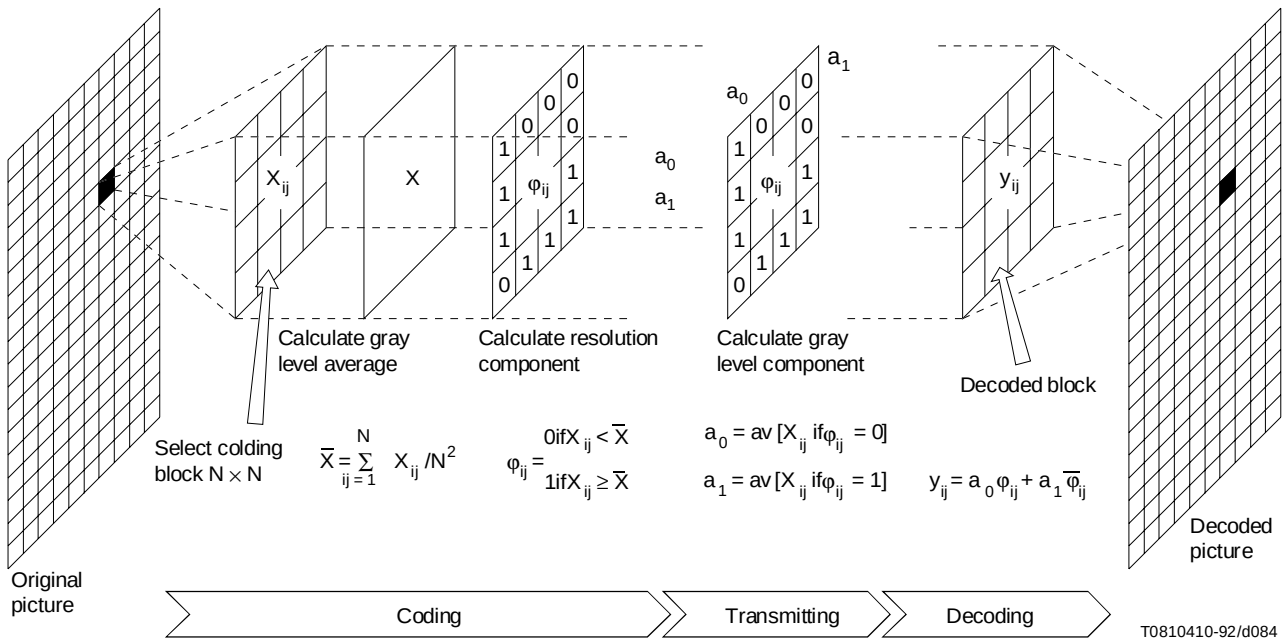


Figure III.1/T.101 – The Principle of the Block Truncation Coding (BTC)

Coding modes

- *Mode A* – A coding mode for smooth area, block size $N \times N$. As the gray level difference within the block is very small, whole $N \times N$ pixel values are approximated by the average value.
- *Mode B* – A basic block coding mode, block size $N \times N$. Both gray level components and the resolution component are calculated and transmitted. To compress further, gray level components are replaced by a_m and a_1 calculated from a_0 and a_1 as follows:

$$a_m = (a_0 + a_1) / 2 \quad (\text{needs 8 bits to transmit})$$

$$a_1 = (a_1 - a_0) / 2 \quad (\text{able to shorten the bit number})$$

- *Mode D* – A coding mode for rapidly changing area, block size $N \times N$. As the pixel values are rapidly changing, the quality becomes very low if this block is coded by coding modes A or B.

It is rather better to transmit all pixels values. To lower data rate, only an upper left pixel value is transmitted and then quantized only differences to this value of other pixels are transmitted.

Switching algorithm

In ABTC coding modes and block size are switched adaptively according to the locality of blocks. The switching algorithm is shown in Figure III.2. Characteristics of selected coding modes are shown in Table III.1.

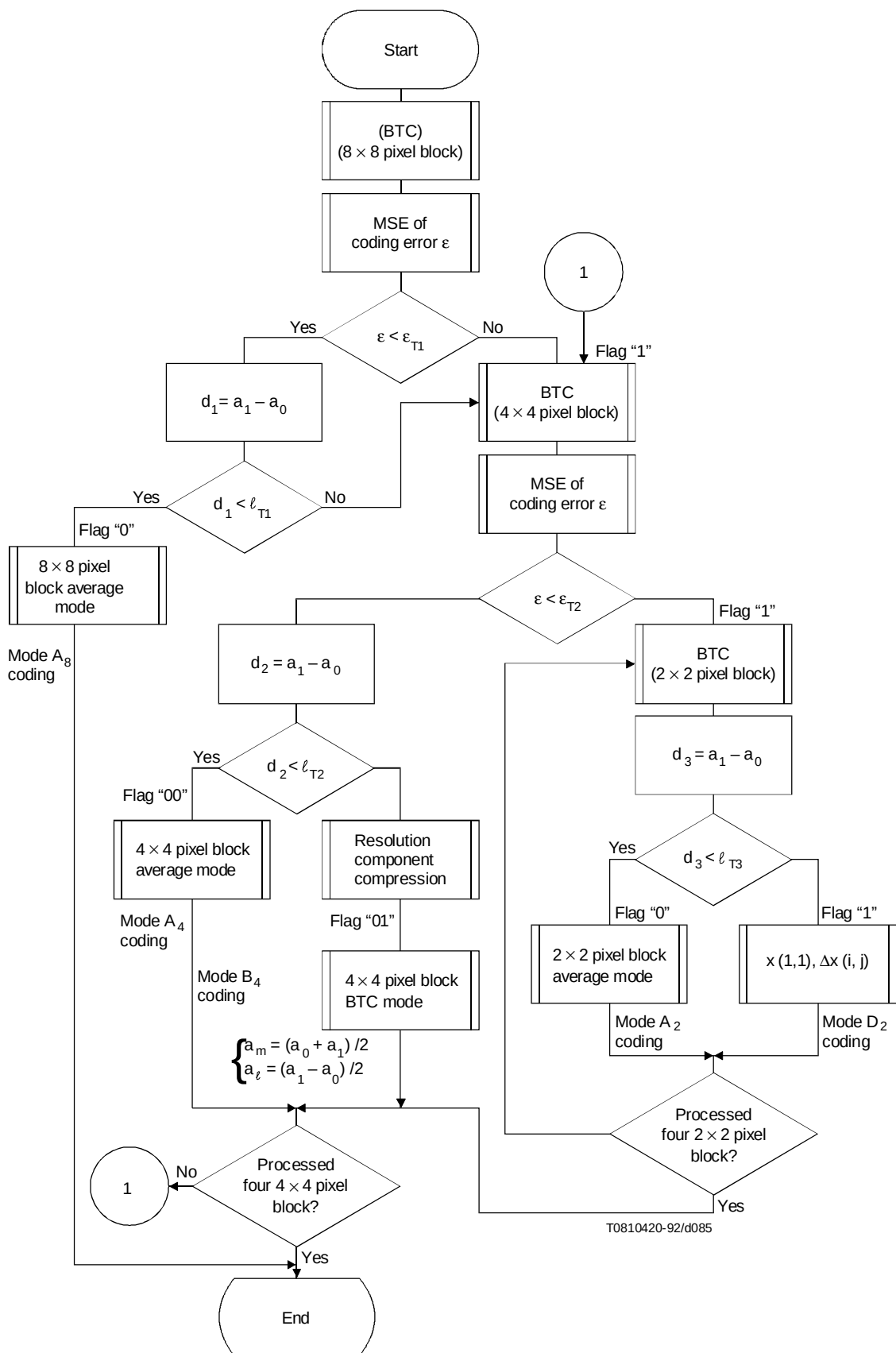


Figure III.2/T.101 – Coding Flow of Adaptive BTC

Table III.1/T.101 – Coded data and its bit rate in each Mode

Coding Mode	Block size		Coded output	Bit rate/pixel
A Average approximation mode	A8	8×8	(flag) + (average value)	$(1/64) + (8/64) = 0.141$
	A4	4×4		$(1/64 + 2/16) + (8/16) = 0.641$
	A2	2×2		$(1/64 + 1/16 + 1/4) + (8/4) = 2.328$
B Block coding mode	B4	4×4	(flag) + (gray level component $a_m a_1$) (resolution component)	$(1/64 + 2/16) + (8/16 + 4/16) +$ $(9 \sim 17/16) = 1.453 \sim 1.953$
D Differential mode	D2	2×2	(flag) + (top left value) + (differential values)	$(1/64 + 1/16 + 1/4) + (8/4) +$ $(4 \sim 5 \times 3/4) = 5.328 \sim 9.078$

Mode switching thresholds

In ABTC, as shown in Figure III.2, 5 mode switching threshold values (ϵt_1 , ϵt_2 , ℓt_1 , ℓt_2 , ℓt_3) are necessary. Switching functions of threshold values are shown in Table III.2.

Table III.2/T.101 – Mode switching threshold

Threshold	Switching modes	Note
ϵt_1	Mode A8 and other Mode	Select smooth 8×8 pixel block. Unselected blocks will be subdivided into 4×4
ϵt_2	Mode A4, B4 and Mode A2, D2	Select smooth 4×4 pixel block. Unselected blocks will be subdivided into 2×2
ℓt_1	Mode A8 and other Mode	Same as ϵt_1
ℓt_2	Mode A4 and Mode B4	Select rather inactive 4×4 pixel blocks
ℓt_3	Mode A2 and Mode D2	Select rather inactive 2×2 pixel blocks. Unselected blocks will be coded in Mode D2

Coding mode distinction flags

Coding mode distinction flags are as follows:

- 1) One bit flag assigned to each 8×8 pixels block data. This flag indicates distinction of “Mode A8” and other coding mode:
 - Mode A8: “0”;
 - Other mode: “1”.

- 2) One bit flag assigned to each 4×4 pixels block data. This flag indicates distinction of “Mode A4 or B4” and “Mode A2 or D2”:
 - Mode A4 or B4: “0”;
 - Mode A2 or D2: “1”.
- 3) Another one bit flag assigned to each 4×4 pixels block data. This flag indicates distinction of “Mode A4” and “Mode B4”:
 - Mode A4: “0”;
 - Mode B4: “1”.
- 4) One bit flag assigned to each 2×2 pixels block data. This flag indicates distinction of “Mode A2” and “Mode D2”:
 - Mode A2: “0”;
 - Mode D2: “1”.

Flag sequence of each coding mode is shown in Table III.3.

Table III.3/T.101 – Flag sequence of each coding mode

	Block size			Total
	8×8	4×4	2×2	
Mode A8	“0”	–	–	“0”
Mode A4	“1”	“00”	–	“100”
Mode B4		“01”	–	“101”
Mode A2		“1”	“0”	“110”
Mode D2			“1”	“111”

Transmission sequence

Output code data of each 8×8 pixels block are transmitted in one burst.

Within one burst, flags of all components are first transmitted and then all component data will follow. Component order is Y-I-Q.

In the example shown in Figure III.3 data stream is as follows:

- *First burst*
 [Y-component flag + I-component flag + Q-component flag] + [Y-component data + I-component data + Q-component data] = [“0” + “I” + “Q”] + [“av (8 bits)” + “I” + “Q”]
- *Second burst*
 [“100010100” + “I” + “Q”] + (av (8 bits) + (a_m, a_l, Φ) + (a_m, a_l, Φ) + av + “I” + “Q”]
- *Third burst*
 [“1110010100” + “I” + “Q”] + [(x11 + D1 + D2 + D3) + av + av + (x11 + D1 + D2 + D3) + (a_m, a_l, Φ) + (a_m, a_l, Φ) av + “I” + “Q”]

A8	A4	B4	D2	A2	B4
			A2	D2	
	B4	A4	B4		A4

FIGURE III.3/T.101

Example of data stream

Restriction component data order is from left to right and from top to bottom. Data value order is from MSB to LSB. If separate transmission of gray level component and resolution component is proceeded, hierarchical decoding will be realized. Because gray level component data transmission realizes fast outline image transmission, and followed resolution component data transmission gives detail of the image.

III.3 ABTC decoding algorithm

Figure III.4 shows the decoding flow of ABTC.

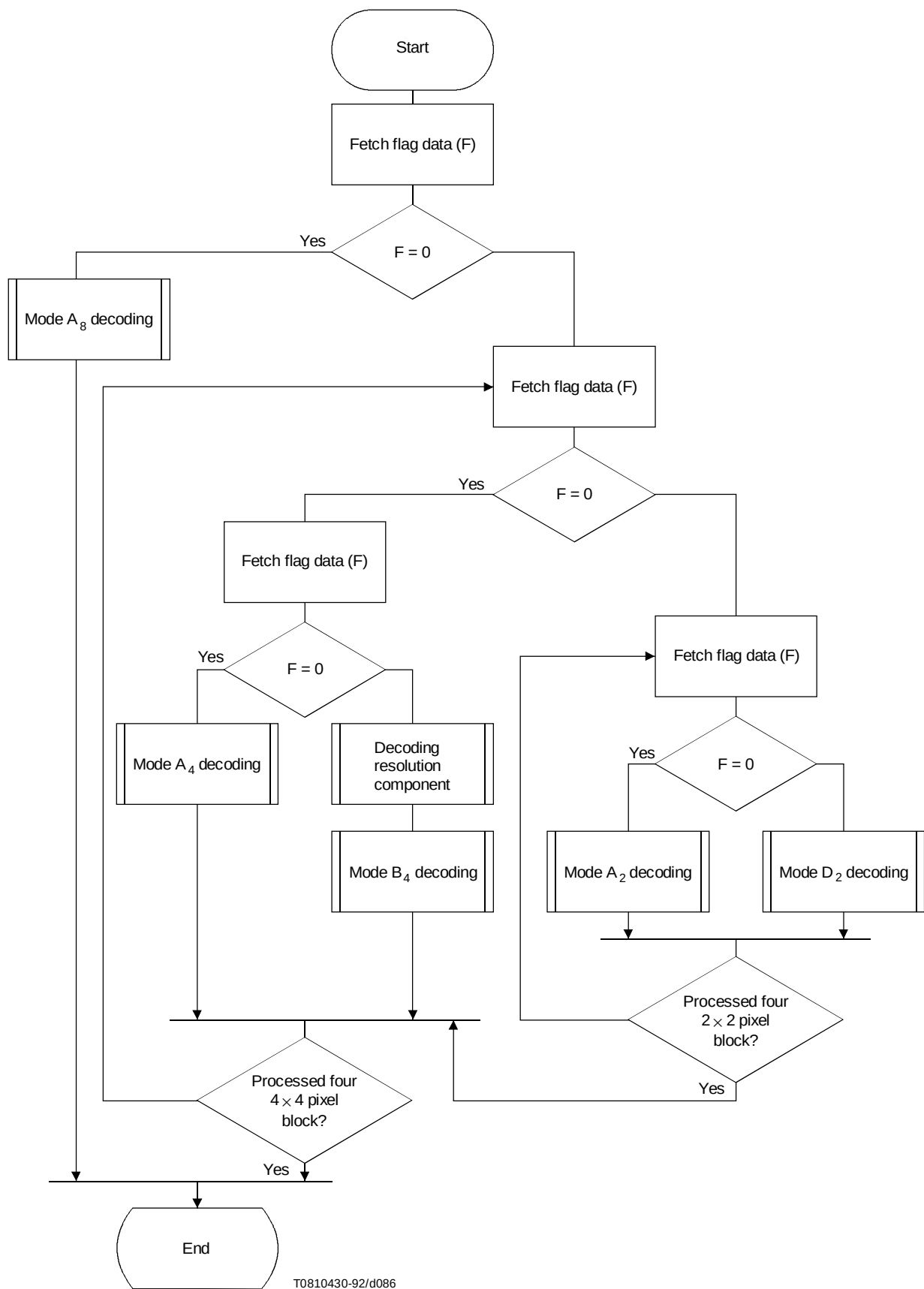


Figure III.4/T.101 – Decoding Flow of Adaptive BTC