
**Information technology — JPEG 2000
image coding system: Core coding
system**

*Technologies de l'information — Système de codage d'image JPEG
2000: Système de codage de noyau*



Reference number
ISO/IEC 15444-1:2004(E)

AVA Documents
Document Sharing Hub
© ISO/IEC 2004

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

© ISO/IEC 2004

All rights reserved. Unless otherwise specified, 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 either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

CONTENTS

	<i>Page</i>
1 Scope	1
2 References	1
2.1 Identical Recommendations International Standards	1
2.2 Additional references	1
3 Definitions	2
4 Abbreviations and symbols	6
4.1 Abbreviations	6
4.2 Symbols	7
5 General description	8
5.1 Purpose	8
5.2 Codestream	8
5.3 Coding principles	8
6 Encoder requirements	10
7 Decoder requirements	10
7.1 Codestream syntax requirements	10
7.2 Optional file format requirements	11
8 Implementation requirements	11
Annex A – Codestream syntax	12
A.1 Markers, marker segments, and headers	12
A.2 Information in the marker segments	14
A.3 Construction of the codestream	15
A.4 Delimiting markers and marker segments	19
A.5 Fixed information marker segment	20
A.6 Functional marker segments	22
A.7 Pointer marker segments	32
A.8 In-bit-stream marker and marker segments	38
A.9 Informational marker segments	39
A.10 Codestream restrictions conforming to this Recommendation International Standard	40
Annex B – Image and compressed image data ordering	42
B.1 Introduction to image data structure concepts	42
B.2 Component mapping to the reference grid	42
B.3 Image area division into tiles and tile-components	44
B.4 Example of the mapping of components to the reference grid (informative)	45
B.5 Transformed tile-component division into resolution levels and sub-bands	48
B.6 Division of resolution levels into precincts	49
B.7 Division of the sub-bands into code-blocks	50
B.8 Layers	51
B.9 Packets	52
B.10 Packet header information coding	54
B.11 Tile and tile-parts	59
B.12 Progression order	59
Annex C – Arithmetic entropy coding	64
C.1 Binary encoding (informative)	64
C.2 Description of the arithmetic encoder (informative)	65
C.3 Arithmetic decoding procedure	76

	<i>Page</i>
Annex D – Coefficient bit modeling.....	84
D.1 Code-block scan pattern within code-blocks.....	84
D.2 Coefficient bits and significance.....	84
D.3 Decoding passes over the bit-planes.....	85
D.4 Initializing and terminating.....	89
D.5 Error resilience segmentation symbol.....	90
D.6 Selective arithmetic coding bypass.....	90
D.7 Vertically causal context formation.....	92
D.8 Flow diagram of the code-block coding.....	92
Annex E – Quantization.....	95
E.1 Inverse quantization procedure.....	95
E.2 Scalar coefficient quantization (informative).....	97
Annex F – Discrete wavelet transformation of tile-components.....	98
F.1 Tile-component parameters.....	98
F.2 Discrete wavelet transformations.....	98
F.3 Inverse discrete wavelet transformation.....	98
F.4 Forward transformation (informative).....	110
Annex G – DC level shifting and multiple component transformations.....	120
G.1 DC level shifting of tile-components.....	120
G.2 Reversible multiple component transformation (RCT).....	121
G.3 Irreversible multiple component transformation (ICT).....	121
G.4 Chrominance component sub-sampling and the reference grid.....	122
Annex H – Coding of images with regions of interest.....	123
H.1 Decoding of ROI.....	123
H.2 Description of the Maxshift method.....	123
H.3 Remarks on region of interest coding (informative).....	124
Annex I – JP2 file format syntax.....	127
I.1 File format scope.....	127
I.2 Introduction to the JP2 file format.....	127
I.3 Greyscale/Colour/Palettized/multi-component specification architecture.....	129
I.4 Box definition.....	131
I.5 Defined boxes.....	133
I.6 Adding intellectual property rights information in JP2.....	148
I.7 Adding vendor-specific information to the JP2 file format.....	148
I.8 Dealing with unknown boxes.....	150
Annex J – Examples and guidelines.....	151
J.1 Software conventions adaptive entropy decoder.....	151
J.2 Selection of quantization step sizes for irreversible transformations.....	153
J.3 Filter impulse responses corresponding to lifting-based irreversible filtering procedures.....	153
J.4 Example of discrete wavelet transformation.....	154
J.5 Row-based wavelet transform.....	158
J.6 Scan-based coding.....	167
J.7 Error resilience.....	167
J.8 Compatibility requirement with JFIF/SPIFF files.....	168
J.9 Implementing the Restricted ICC method outside of a full ICC colour management engine.....	168
J.10 An example of the interpretation of multiple components.....	173
J.11 An example of decoding showing intermediate steps.....	173
J.12 Visual frequency weighting.....	177
J.13 Encoder sub-sampling of components.....	179
J.14 Rate control.....	180
J.15 Guidelines on handling YCC codestream.....	184

	<i>Page</i>
Annex K – Bibliography	186
K.1 General	186
K.2 Quantization and entropy coding	186
K.3 Wavelet transformation	186
K.4 Region of interest coding	187
K.5 Visual frequency weighting	187
K.6 Error resilience	187
K.7 Scan-based coding	188
K.8 Colour	188
Annex L – Patent statement	189
Index	190

LIST OF FIGURES

	<i>Page</i>
Figure 5-1 – Specification block diagram.....	9
Figure A.1 – Example of the marker segment description figures.....	13
Figure A.2 – Construction of the codestream	16
Figure A.3 – Construction of the main header.....	17
Figure A.4 – Construction of the first tile-part header of a given tile.....	18
Figure A.5 – Construction of a non-first tile-part header.....	18
Figure A.6 – Start of tile-part syntax	19
Figure A.7 – Image and tile size syntax.....	21
Figure A.8 – Coding style default syntax	23
Figure A.9 – Coding style parameter diagram of the SGcod and SPcod parameters.....	24
Figure A.10 – Coding style component syntax.....	26
Figure A.11 – Coding style parameter diagram of the SPcoc parameters	27
Figure A.12 – Region-of-interest syntax	27
Figure A.13 – Quantization default syntax.....	28
Figure A.14 – Quantization component syntax	30
Figure A.15 – Progression order change tile syntax	31
Figure A.16 – Tile-part lengths.....	32
Figure A.17 – Tile part length syntax	33
Figure A.18 – Packets length, main header syntax	34
Figure A.19 – Packet length, tile-part header syntax	35
Figure A.20 – Packed packet headers, main header syntax	36
Figure A.21 – Packed packet headers, tile-part header syntax.....	37
Figure A.22 – Start of packet syntax	38
Figure A.23 – Component registration syntax.....	39
Figure A.24 – Comment syntax.....	40
Figure B.1 – Reference grid diagram.....	43
Figure B.2 – Component sample locations on the reference grid for different XRsiz and YRsiz values	43
Figure B.3 – Example of upper left component sample locations	44
Figure B.4 – Tiling of the reference grid diagram	44
Figure B.5 – Reference grid example	46
Figure B.6 – Example tile sizes and locations for component 0	47
Figure B.7 – Example tile sizes and locations for component 1	48
Figure B.8 – Precincts of one reduced resolution	49

	<i>Page</i>
Figure B.9 – Code-blocks and precincts in sub-band b from four different tiles.....	51
Figure B.10 – Diagram of precincts of one resolution level of one component	52
Figure B.11 – Diagram of code-blocks within precincts at one resolution level	53
Figure B.12 – Example of a tag tree representation.....	54
Figure B.13 – Example of the information known to the encoder.....	57
Figure B.14 – Example of progression order volume in two dimensions.....	62
Figure B.15 – Example of the placement of POC marker segments.....	63
Figure C.1 – Arithmetic encoder inputs and outputs	64
Figure C.2 – Encoder for the MQ-coder.....	66
Figure C.3 – ENCODE procedure	67
Figure C.4 – CODE1 procedure	67
Figure C.5 – CODE0 procedure	68
Figure C.6 – CODELPS procedure with conditional MPS/LPS exchange.....	69
Figure C.7 – CODEMPS procedure with conditional MPS/LPS exchange.....	71
Figure C.8 – Encoder renormalization procedure.....	72
Figure C.9 – BYTEOUT procedure for encoder.....	73
Figure C.10 – Initialization of the encoder	74
Figure C.11 – FLUSH procedure.....	75
Figure C.12 – Setting the final bits in the C register.....	76
Figure C.13 – Arithmetic decoder inputs and outputs	76
Figure C.14 – Decoder for the MQ-coder.....	77
Figure C.15 – Decoding an MPS or an LPS	78
Figure C.16 – Decoder MPS path conditional exchange procedure	79
Figure C.17 – Decoder LPS path conditional exchange procedure	80
Figure C.18 – Decoder renormalization procedure.....	81
Figure C.19 – BYTEIN procedure for decoder.....	82
Figure C.20 – Initialization of the decoder	83
Figure D.1 – Example scan pattern of a code-block bit-plane	84
Figure D.2 – Neighbors states used to form the context.....	85
Figure D.3 – Flow chart for all coding passes on a code-block bit-plane.....	93
Figure F.1 – Inputs and outputs of the IDWT procedure.....	98
Figure F.2 – The IDWT ($N_L = 2$).....	99
Figure F.3 – The IDWT procedure	100
Figure F.4 – Inputs and outputs of the 2D_SR procedure.....	100
Figure F.5 – One level of reconstruction from four sub-bands (2D_SR procedure) into sub-bands	100
Figure F.6 – The 2D_SR procedure.....	101
Figure F.7 – Parameters of 2D_INTERLEAVE procedure	101
Figure F.8 – The 2D_INTERLEAVE procedure	102

	<i>Page</i>
Figure F.9 – Inputs and outputs of the HOR_SR procedure	103
Figure F.10 – The HOR_SR procedure	104
Figure F.11 – Inputs and outputs of the VER_SR procedure	105
Figure F.12 – The VER_SR procedure	105
Figure F.13 – Parameters of the 1D_SR procedure	106
Figure F.14 – The 1D_SR procedure	106
Figure F.15 – Periodic symmetric extension of signal	106
Figure F.16 – Parameters of the ID_FILTR procedure	107
Figure F.17 – Inputs and outputs of the FDWT procedure	110
Figure F.18 – The FDWT ($N_L = 2$)	110
Figure F.19 – The FDWT procedure	111
Figure F.20 – Inputs and outputs of the 2D_SD procedure	111
Figure F.21 – One-level decomposition into four sub-bands (2D_SD procedure)	112
Figure F.22 – The 2D_SD procedure	112
Figure F.23 – Inputs and outputs of the VER_SD procedure	112
Figure F.24 – The VER_SD procedure	113
Figure F.25 – Inputs and outputs of the HOR_SD procedure	114
Figure F.26 – The HOR_SD procedure	114
Figure F.27 – Parameters of 2D_DEINTERLEAVE procedure	115
Figure F.28 – The 2D_DEINTERLEAVE procedure	116
Figure F.29 – Parameters of the 1D_SD procedure	117
Figure F.30 – The 1D_SD procedure	117
Figure F.31 – Parameters of the 1D_FILTD procedure	118
Figure G.1 – Placement of the DC level shifting with component transformation	120
Figure G.2 – Placement of the DC level shifting without component transformation	120
Figure H.1 – The inverse wavelet transformation with the 5-3 reversible filter	125
Figure H.2 – The inverse wavelet transformation with the 9-7 irreversible filter	125
Figure I.1 – Conceptual structure of a JP2 file	128
Figure I.2 – Example of the box description figures	131
Figure I.3 – Example of the superbox description figures	131
Figure I.4 – Organization of a box	131
Figure I.5 – Illustration of box lengths	132
Figure I.6 – Organization of the contents of a File Type box	134
Figure I.7 – Organization of the contents of a JP2 Header box	135
Figure I.8 – Organization of the contents of an Image Header box	136
Figure I.9 – Organization of the contents of a Bits Per Component box	137
Figure I.10 – Organization of the contents of a Colour Specification box	138
Figure I.11 – Organization of the contents of the Palette box	140

	<i>Page</i>
Figure I.12 – Organization of the contents of a Component Mapping box.....	141
Figure I.13 – Organization of the contents of a Channel Definition box.....	142
Figure I.14 – Organization of the contents of the Resolution box	145
Figure I.15 – Organization of the contents of the Capture Resolution box.....	145
Figure I.16 – Organization of the contents of the Default Display Resolution box.....	146
Figure I.17 – Organization of the contents of the Contiguous Codestream box	147
Figure I.18 – Organization of the contents of the XML box	148
Figure I.19 – Organization of the contents of the UUID box	148
Figure I.20 – Organization of the contents of a UUID Info box.....	149
Figure I.21 – Organization of the contents of a UUID List box	149
Figure I.22 – Organization of the contents of a Data Entry URL box	150
Figure J.1 – Initialization of the software-conventions decoder	151
Figure J.2 – Decoding an MPS or an LPS in the software-conventions decoder.....	152
Figure J.3 – Inserting a new byte into the C register in the software-conventions decoder.....	152
Figure J.4 – The FDWT_ROW procedure.....	159
Figure J.5 – The GET_ROW procedure	160
Figure J.6 – The INIT procedure	161
Figure J.7 – The START_VERT procedure	162
Figure J.8 – The RB_VERT_1 procedure.....	163
Figure J.9 – The RB_VERT_2 procedure.....	164
Figure J.10 – The END_1 procedure	165
Figure J.11 – The END_2 procedure	166
Figure J.12 – Illustration of code-block contributions to bit-stream layers	181
Figure J.13 – 4:2:2 format (co-sited)	184
Figure J.14 – 4:2:2 format (centered)	184
Figure J.15 – 4:2:0 format (co-sited)	185
Figure J.16 – 4:2:0 format (centered)	185

LIST OF TABLES

	<i>Page</i>
Table A.1 – Marker definitions.....	13
Table A.2 – List of markers and marker segments	14
Table A.3 – Information in the marker segments	15
Table A.4 – Start of codestream parameter values	19
Table A.5 – Start of tile-part parameter values	20
Table A.6 – Number of tile-parts, TNsot, parameter value	20
Table A.7 – Start of data parameter values	20
Table A.8 – End of codestream parameter values	20
Table A.9 – Image and tile size parameter values	22
Table A.10 – Capability Rsiz parameter.....	22
Table A.11 – Component Ssiz parameter	22
Table A.12 – Coding style default parameter values	23
Table A.13 – Coding style parameter values for the Scod parameter	24
Table A.14 – Coding style parameter values of the SGcod parameter	24
Table A.15 – Coding style parameter values of the SPcod and SPcoc parameters.....	24
Table A.16 – Progression order for the SGcod, SPcoc, and Ppoc parameters.....	25
Table A.17 – Multiple component transformation for the SGcod parameters.....	25
Table A.18 – Width or height exponent of the code-blocks for the SPcod and SPcoc parameters.....	25
Table A.19 – Code-block style for the SPcod and SPcoc parameters.....	25
Table A.20 – Transformation for the SPcod and SPcoc parameters	26
Table A.21 – Precinct width and height for the SPcod and SPcoc parameters	26
Table A.22 – Coding style component parameter values	27
Table A.23 – Coding style parameter values for the Scoc parameter	27
Table A.24 – Region-of-interest parameter values	28
Table A.25 – Region-of-interest parameter values for the Srgn parameter	28
Table A.26 – Region-of-interest values from SPrn parameter (Srgn = 0)	28
Table A.27 – Quantization default parameter values.....	29
Table A.28 – Quantization default values for the Sqcd and Sqcc parameters	29
Table A.29 – Reversible step size values for the SPqcd and SPqcc parameters (reversible transform only)	29
Table A.30 – Quantization values for the SPqcd and SPqcc parameters (irreversible transformation only).....	30
Table A.31 – Quantization component parameter values	31
Table A.32 – Progression order change, tile parameter values.....	32
Table A.33 – Tile-part length parameter values	33

	<i>Page</i>
Table A.34 – Size parameters for Stlm.....	34
Table A.35 – Packets length, main header parameter values.....	35
Table A.36 – Iplm, Iplt list of packet lengths	35
Table A.37 – Packet length, tile-part headers parameter values	36
Table A.38 – Packed packet headers, main header parameter values.....	37
Table A.39 – Packet header, tile-part headers parameter values	37
Table A.40 – Start of packet parameter values	38
Table A.41 – End of packet header parameter values.....	39
Table A.42 – Component registration parameter values.....	39
Table A.43 – Comment parameter values.....	40
Table A.44 – Registration values for the Rcom parameter	40
Table A.45 – Codestream restrictions.....	41
Table B.1 – Quantities (xo_b, yo_b) for sub-band b	49
Table B.2 – Example of layer formation (only one component shown).....	52
Table B.3 – Example of packet formation.....	53
Table B.4 – Codewords for the number of coding passes for each code-block.....	56
Table B.5 – Example packet header bit stream.....	58
Table C.1 – Encoder register structures.....	66
Table C.2 – Qe values and probability estimation	69
Table C.2 – Qe values and probability estimation (<i>concluded</i>)	70
Table C.3 – Decoder register structures.....	77
Table D.1 – Contexts for the significance propagation and cleanup coding passes.....	86
Table D.2 – Contributions of the vertical (and the horizontal) neighbors to the sign context	86
Table D.3 – Sign contexts from the vertical and horizontal contributions	87
Table D.4 – Contexts for the magnitude refinement coding passes.....	87
Table D.5 – Run-length decoder for cleanup passes.....	88
Table D.6 – Example of sub-bit-plane coding order and significance propagation.....	88
Table D.7 – Initial states for all contexts	89
Table D.8 – Arithmetic coder termination patterns	89
Table D.9 – Selective arithmetic coding bypass.....	91
Table D.10 – Decisions in the context model flow chart.....	94
Table D.11 – Decoding in the context model flow chart	94
Table E.1 – Sub-band gains	96
Table F.1 – Decomposition level n_b for sub-band b	99
Table F.2 – Extension to the left.....	107
Table F.3 – Extension to the right.....	107
Table F.4 – Definition of lifting parameters for the 9-7 irreversible filter.....	109
Table F.5 – Definition of coefficients g_n	109

	<i>Page</i>
Table F.6 – Intermediate expressions (r_0, r_1, s_0, t_0).....	109
Table F.7 – Intermediate expressions	110
Table F.8 – Extension to the left.....	117
Table F.9 – Extension to the right.....	118
Table I.1 – Binary structure of a box	132
Table I.2 – Defined boxes.....	133
Table I.3 – Legal Brand values	134
Table I.4 – Format of the contents of the File Type box.....	135
Table I.5 – Format of the contents of the Image Header box	137
Table I.6 – BPC values	137
Table I.7 – Format of the contents of the Bits Per Component box.....	138
Table I.8 – BPC ⁱ values	138
Table I.9 – Legal METH values	139
Table I.10 – Legal EnumCS values	139
Table I.11 – Format of the contents of the Colour Specification box.....	140
Table I.12 – Format of the contents of the Palette box	141
Table I.13 – B ⁱ values	141
Table I.14 – MTyp ⁱ field values.....	142
Table I.15 – Format of the contents of the Component Mapping box	142
Table I.16 – Typ ⁱ field values	143
Table I.17 – Assoc ⁱ field values	143
Table I.18 – Colours indicated by the Assoc ⁱ field.....	144
Table I.19 – Format of the Channel Definition box.....	145
Table I.20 – Format of the contents of the Capture Resolution box	146
Table I.21 – Format of the contents of the Default Display Resolution box	147
Table I.22 – Format of the contents of the Contiguous Codestream box.....	147
Table I.23 – Format of the contents of a UUID box	149
Table I.24 – UUID List box contents data structure values.....	149
Table I.25 – Data Entry URL box contents data structure values.....	150
Table J.1 – Definition of impulse responses for the 9-7 irreversible analysis filter bank.....	153
Table J.2 – Definition of impulse responses for the 9-7 irreversible synthesis filter band	154
Table J.3 – Source tile component samples	154
Table J.4 – 2LL sub-band coefficients (9-7 irreversible wavelet transformation).....	155
Table J.5 – 2HL sub-band coefficients (9-7 irreversible wavelet transformation).....	155
Table J.6 – 2LH sub-band coefficients (9-7 irreversible wavelet transformation).....	155
Table J.7 – 2HH sub-band coefficients (9-7 irreversible wavelet transformation)	155
Table J.8 – 1HL sub-band coefficients (9-7 irreversible wavelet transformation).....	155
Table J.9 – 1LH sub-band coefficients (9-7 irreversible wavelet transformation).....	156

	<i>Page</i>
Table J.10 – 1HH sub-band coefficients (9-7 irreversible wavelet transformation)	156
Table J.11 – 2LL sub-band coefficients (5-3 reversible wavelet transformation)	156
Table J.12 – 2HL sub-band coefficients (5-3 reversible wavelet transformation)	156
Table J.13 – 2LH sub-band coefficient (5-3 reversible wavelet transformation)	157
Table J.14 – 2HH sub-band coefficients (5-3 reversible wavelet transformation)	157
Table J.15 – 1HL sub-band coefficients (5-3 reversible wavelet transformation)	157
Table J.16 – 1LH sub-band coefficients (5-3 reversible wavelet transformation)	157
Table J.17 – 1HH sub-band coefficients (5-3 reversible wavelet transformation)	158
Table J.18 – Error resilience tools	167
Table J.19 – Processing tags used by a Restricted ICC profile	169
Table J.20 – Decoding first packet header	175
Table J.21 – Decoding second packet header	175
Table J.22 – Arithmetic decode of first code-block	176
Table J.23 – Arithmetic decode of second code-block	177
Table J.24 – Recommended frequency weighting	179
Table J.25 – Recommended frequency weighting for multiple component (colour) images	179
Table J.26 – CRG (Component registration) values	185
Table L.1 – Received intellectual property rights statements	189

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents, as indicated in Table L.1.

This part of ISO/IEC 15444 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information* in collaboration with ITU-T. The identical text is published as ITU-T Rec. T.800.

This second edition cancels and replaces the first edition (ISO/IEC 15444-1:2000), of which it constitutes a minor revision. It also incorporates the Amendment ISO/IEC 15444-1:2000/Amd.1:2002 and the Technical Corrigenda ISO/IEC 15444-1:2000/Cor.1:2002 and ISO/IEC 15444-1:2000/Cor.2:2002.

ISO/IEC 15444 consists of the following parts, under the general title *Information technology — JPEG 2000 image coding system*:

- *Part 1: Core coding system*
- *Part 2: Extensions*
- *Part 3: Motion JPEG 2000*
- *Part 4: Conformance testing*
- *Part 5: Reference software*
- *Part 6: Compound image file format*
- *Part 9: Interactivity tools, APIs and protocols*

The following part is under preparation:

- *Part 8: Secure JPEG 2000*

**INTERNATIONAL STANDARD
ITU-T RECOMMENDATION**

Information technology – JPEG 2000 image coding system: Core coding system

1 Scope

This Recommendation | International Standard defines a set of lossless (bit-preserving) and lossy compression methods for coding bi-level, continuous-tone grey-scale, palletized color, or continuous-tone colour digital still images.

This Recommendation | International Standard:

- specifies decoding processes for converting compressed image data to reconstructed image data;
- specifies a codestream syntax containing information for interpreting the compressed image data;
- specifies a file format;
- provides guidance on encoding processes for converting source image data to compressed image data;
- provides guidance on how to implement these processes in practice.

2 References

The following Recommendations and International Standards contain provisions which, through reference in this text, constitute provisions of this Recommendation | International Standard. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation | International Standard are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of the ITU maintains a list of currently valid ITU-T Recommendations.

2.1 Identical Recommendations | International Standards

- ITU-T Recommendation T.81 (1992) | ISO/IEC 10918-1:1994, *Information technology – Digital compression and coding of continuous-tone still images: Requirements and guidelines*.
- ITU-T Recommendation T.88 (2000) | ISO/IEC 14492:2001, *Information technology – Lossy/lossless coding of bi-level images*.
- ISO/IEC 646:1991, *Information technology – ISO 7-bit coded character set for information interchange*.
- ISO 8859-15:1999, *Information technology – 8-bit single-byte coded graphic character sets – Part 15: Latin alphabet No. 9*.
- ITU-T Recommendation T.84 (1996) | ISO/IEC 10918-3:1997, *Information technology – Digital compression and coding of continuous-tone still images: Extensions*.
- ITU-T Recommendation T.84 (1996)/Amd.1 (1999) | ISO/IEC 10918-3:1997/Amd.1:1999, *Information technology – Digital compression and coding of continuous-tone still images: Extensions – Amendment 1: Provisions to allow registration of new compression types and versions in the SPIFF header*.
- ITU-T Recommendation T.86 (1998) | ISO/IEC 10918-4:1999, *Information technology – Digital compression and coding of continuous-tone still images: Registration of JPEG Profiles, SPIFF Profiles, SPIFF Tags, SPIFF colour Spaces, APPn Markers, SPIFF, Compression types and Registration Authorities (REGAUT)*.
- ITU-T Recommendation T.87 (1998) | ISO/IEC 14495-1:1999, *Lossless and near-lossless compression of continuous-tone still images – Baseline*.

2.2 Additional references

- Specification ICC.1:1998-09, *File format for Color Profiles*.
- IEC 61966-2-1:1999, *Multimedia systems and equipment – Colour measurement and management – Part 2-1: Colour management – Default RGB colour space – sRGB*.
- W3C REC-xml-19980210, *Extensible Markup Language (XML 1.0)*.
- IETF RFC 2279 (1998), UTF-8, *a transformation format of ISO 10646*.

- ISO/IEC 11578:1996, *Information technology – Open Systems Interconnection – Remote Procedure Call*.
- IEC 61966-2-1:1999/Amd.1:2003, *Multimedia systems and equipment – Colour measurement and management – Part 2-1: Colour management – Default RGB colour space – sRGB*.

3 Definitions

For the purposes of this Recommendation | International Standard, the following definitions apply.

- 3.1** $\lfloor x \rfloor$, **floor function**: This indicates the largest integer not exceeding x .
- 3.2** $\lceil x \rceil$, **ceiling function**: This indicates the smallest integer not exceeded by x .
- 3.3** **5-3 reversible filter**: A particular filter pair used in the wavelet transformation. This reversible filter pair has 5 taps in the low-pass and 3 taps in the high-pass.
- 3.4** **9-7 irreversible filter**: A particular filter pair used in the wavelet transformation. This irreversible filter pair has 9 taps in the low-pass and 7 taps in the high-pass.
- 3.5** **AND**: Bit wise AND logical operator.
- 3.6** **arithmetic coder**: An entropy coder that converts variable length strings to variable length codes (encoding) and visa versa (decoding).
- 3.7** **auxiliary channel**: A channel that is used by the application outside the scope of colourspace conversion. For example, an opacity channel or a depth channel would be an auxiliary channel.
- 3.8** **bit**: A contraction of the term "binary digit"; a unit of information represented by a zero or a one.
- 3.9** **bit-plane**: A two dimensional array of bits. In this Recommendation | International Standard a bit-plane refers to all the bits of the same magnitude in all coefficients or samples. This could refer to a bit-plane in a component, tile-component, code-block, region of interest, or other.
- 3.10** **bit stream**: The actual sequence of bits resulting from the coding of a sequence of symbols. It does not include the markers or marker segments in the main and tile-part headers or the EOC marker. It does include any packet headers and in stream markers and marker segments not found within the main or tile-part headers.
- 3.11** **big endian**: The bits of a value representation occur in order from most significant to least significant.
- 3.12** **box**: A portion of the file format defined by a length and unique box type. Boxes of some types may contain other boxes.
- 3.13** **box contents**: Refers to the data wrapped within the box structure. The contents of a particular box are stored within the DBox field within the box data structure.
- 3.14** **box type**: Specifies the kind of information that shall be stored with the box. The type of a particular box is stored within the TBox field within the box data structure.
- 3.15** **byte**: Eight bits.
- 3.16** **channel**: One logical component of the image. A channel may be a direct representation of one component from the codestream, or may be generated by the application of a palette to a component from the codestream.
- 3.17** **cleanup pass**: A coding pass performed on a single bit-plane of a code-block of coefficients. The first pass and only coding pass for the first significant bit-plane is a cleanup pass; the third and the last pass of every remaining bit-plane is a cleanup pass.
- 3.18** **codestream**: A collection of one or more bit streams and the main header, tile-part headers, and the EOC required for their decoding and expansion into image data. This is the image data in a compressed form with all of the signalling needed to decode.
- 3.19** **code-block**: A rectangular grouping of coefficients from the same sub-band of a tile-component.
- 3.20** **code-block scan**: The order in which the coefficients within a code-block are visited during a coding pass. The code-block is processed in stripes, each consisting of four rows (or all remaining rows if less than four) and spanning the width of the code-block. Each stripe is processed column by column from top to bottom and from left to right.
- 3.21** **coder**: An embodiment of either an encoding or decoding process.

- 3.22 coding pass:** A complete pass through a code-block where the appropriate coefficient values and context are applied. There are three types of coding passes: significance propagation pass, magnitude refinement pass and cleanup pass. The result of each pass (after arithmetic coding, if selective arithmetic coding bypass is not used) is a stream of compressed image data.
- 3.23 coefficient:** The values that are result of a transformation.
- 3.24 colour channel:** A channel that functions as an input to a colour transformation system. For example, a red channel or a greyscale channel would be a colour channel.
- 3.25 component:** A two-dimensional array of samples. A image typically consists of several components, for instance representing red, green, and blue.
- 3.26 compressed image data:** Part or all of a bit stream. Can also refer to a collection of bit streams in part or all of a codestream.
- 3.27 conforming reader:** An application that reads and interprets a JP2 file correctly.
- 3.28 context:** Function of coefficients previously decoded and used to condition the decoding of the present coefficient.
- 3.29 context label:** The arbitrary index used to distinguish different context values. The labels are used as a convenience of notation rather than being normative.
- 3.30 context vector:** The binary vector consisting of the significance states of the coefficients included in a context.
- 3.31 decoder:** An embodiment of a decoding process, and optionally a colour transformation process.
- 3.32 decoding process:** A process which takes as its input all or part of a codestream and outputs all or part of a reconstructed image.
- 3.33 decomposition level:** A collection of wavelet sub-bands where each coefficient has the same spatial impact or span with respect to the source component samples. These include the HL, LH, and HH sub-bands of the same two dimensional sub-band decomposition. For the last decomposition level the LL sub-band is also included.
- 3.34 delimiting markers and marker segments:** Markers and marker segments that give information about beginning and ending points of structures in the codestream.
- 3.35 discrete wavelet transformation (DWT):** A transformation that iteratively transforms one signal into two or more filtered and decimated signals corresponding to different frequency bands. This transformation operates on spatially discrete samples.
- 3.36 encoder:** An embodiment of an encoding process.
- 3.37 encoding process:** A process that takes as its input all or part of a source image data and outputs a codestream.
- 3.38 file format:** A codestream and additional support data and information not explicitly required for the decoding of codestream. Examples of such support data include text fields providing titling, security and historical information, data to support placement of multiple codestreams within a given data file, and data to support exchange between platforms or conversion to other file formats.
- 3.39 fixed information markers and fixed information marker segments:** Markers and marker segments that offer information about the original image.
- 3.40 functional markers and functional marker segments:** Markers and marker segments that offer information about the coding procedures.
- 3.41 grid resolution:** The spatial resolution of the reference grid, specifying the distance between neighboring points on the reference grid.
- 3.42 guard bits:** Additional most significant bits that have been added to sample data.
- 3.43 header:** Either a part of the codestream that contains only markers and marker segments (main header and tile-part header) or the signalling part of a packet (packet header).
- 3.44 HH sub-band:** The sub-band obtained by forward horizontal high-pass filtering and vertical high-pass filtering. This subband contributes to reconstruction with inverse vertical high-pass filtering and horizontal high-pass filtering.

- 3.45 HL sub-band:** The sub-band obtained by forward horizontal high-pass filtering and vertical low-pass filtering. This sub-band contributes to reconstruction with inverse vertical low-pass filtering and horizontal high-pass filtering.
- 3.46 image:** The set of all components.
- 3.47 image area:** A rectangular part of the reference grid, registered by offsets from the origin and the extent of the reference grid.
- 3.48 image area offset:** The number of reference grid points down and to the right of the reference grid origin where the origin of the image area can be found.
- 3.49 image data:** The components and component samples making up an image. Image data can refer to either the source image data or the reconstructed image data.
- 3.50 in-bit-stream markers and in-bit-stream marker segments:** Markers and marker segments that provide error resilience functionality.
- 3.51 informational markers and informational marker segments:** Markers and marker segments that offer ancillary information.
- 3.52 irreversible:** A transformation, progression, system, quantization, or other process that, due to systemic or quantization error, disallows lossless recovery. An irreversible process can only lead to lossy compression.
- 3.53 JP2 file:** The name of a file in the file format described in this Recommendation | International Standard. Structurally, a JP2 file is a contiguous sequence of boxes.
- 3.54 JPEG:** Used to refer globally to the encoding and decoding process of the following Recommendations | International Standards:
- ITU-T Rec. T.81 | ISO/IEC 10918-1, Information technology – Digital compression and coding of continuous-tone still images: Requirements and guidelines;
 - ITU-T Rec. T.83 | ISO/IEC 10918-2, Information technology – Digital compression and coding of continuous-tone still images: Compliance testing;
 - ITU-T Rec. T.84 | ISO/IEC 10918-3, Information technology – Digital compression and coding of continuous-tone still images: Extensions;
 - ITU-T Rec. T.84 | ISO/IEC 10918-3/Amd.1, Information technology – Digital compression and coding of continuous-tone still images: Extensions – Amendment 1: Provisions to allow registration of new compression types and versions in the SPIFF header;
 - ITU-T Rec. T.86 | ISO/IEC 10918-4, Information technology – Digital compression and coding of continuous-tone still images: Registration of JPEG Profiles, SPIFF Profiles, SPIFF Tags, SPIFF colour Spaces, APPn Markers, SPIFF, Compression types and Registration Authorities (REGAUT).
- 3.55 JPEG 2000:** Used to refer globally to the encoding and decoding processes in this Recommendation | International Standard and their embodiment in applications.
- 3.56 LH sub-band:** The sub-band obtained by forward horizontal low-pass filtering and vertical high-pass filtering. This sub-band contributes to reconstruction with inverse vertical high-pass filtering and horizontal low-pass filtering.
- 3.57 LL sub-band:** The sub-band obtained by forward horizontal low-pass filtering and vertical low-pass filtering. This sub-band contributes to reconstruction with inverse vertical low-pass filtering and horizontal low-pass filtering.
- 3.58 layer:** A collection of compressed image data from coding passes of one, or more, code-blocks of a tile-component. Layers have an order for encoding and decoding that must be preserved.
- 3.59 lossless:** A descriptive term for the effect of the overall encoding and decoding processes in which the output of the decoding process is identical to the input to the encoding process. Distortion-free restoration can be assured. All of the coding processes or steps used for encoding and decoding are reversible.
- 3.60 lossy:** A descriptive term for the effect of the overall encoding and decoding processes in which the output of the decoding process is not identical to the input to the encoding process. There is distortion (measured mathematically). At least one of the coding processes or steps used for encoding and decoding is irreversible.
- 3.61 magnitude refinement pass:** A type of coding pass.
- 3.62 main header:** A group of markers and marker segments at the beginning of the codestream that describe the image parameters and coding parameters that can apply to every tile and tile-component.

- 3.63 marker:** A two-byte code in which the first byte is hexadecimal FF (0xFF) and the second byte is a value between 1 (0x01) and hexadecimal FE (0xFE).
- 3.64 marker segment:** A marker and associated (not empty) set of parameters.
- 3.65 mod:** $\text{mod}(y, x) = z$, where z is such that $0 \leq z < x$, and such that $y - z$ is a multiple of x .
- 3.66 packet:** A part of the bit stream comprising a packet header and the compressed image data from one layer of one precinct of one resolution level of one tile-component.
- 3.67 packet header:** Portion of the packet that contains signalling necessary for decoding that packet.
- 3.68 pointer markers and pointer marker segments:** Markers and marker segments that offer information about the location of structures in the codestream.
- 3.69 precinct:** A one rectangular region of a transformed tile-component, within each resolution level, used for limiting the size of packets.
- 3.70 precision:** Number of bits allocated to a particular sample, coefficient, or other binary numerical representation.
- 3.71 progression:** The order of a codestream where the decoding of each successive bit contributes to a "better" reconstruction of the image. What metrics make the reconstruction "better" is a function of the application. Some examples of progression are increasing resolution or improved sample fidelity.
- 3.72 quantization:** A method of reducing the precision of the individual coefficients to reduce the number of bits used to entropy-code them. This is equivalent to division while compressing and multiplying while decompressing. Quantization can be achieved by an explicit operation with a given quantization value or by dropping (truncating) coding passes from the codestream.
- 3.73 raster order:** A particular sequential order of data of any type within an array. The raster order starts with the top left data point and moves to the immediate right data point, and so on, to the end of the row. After the end of the row is reached the next data point in the sequence is the left-most data point immediately below the current row. This order is continued to the end of the array.
- 3.74 reconstructed image:** An image that is the output of a decoder.
- 3.75 reconstructed sample:** A sample reconstructed by the decoder. This always equals the original sample value in lossless coding but may differ from the original sample value in lossy coding.
- 3.76 reference grid:** A regular rectangular array of points used as a reference for other rectangular arrays of data. Examples include components and tiles.
- 3.77 reference tile:** A rectangular sub-grid of any size associated with the reference grid.
- 3.78 region of interest (ROI):** A collections of coefficients that are considered of particular relevance by some user-defined measure.
- 3.79 resolution level:** Equivalent to decomposition level with one exception: the LL sub-band is also a separate resolution level.
- 3.80 reversible:** A transformation, progression, system, or other process that does not suffer systemic or quantization error and, therefore, allows lossless signal recovery.
- 3.81 sample:** One element in the two-dimensional array that comprises a component.
- 3.82 segmentation symbol:** A special symbol coded with a uniform context at the end of each coding pass for error resilience.
- 3.83 selective arithmetic coding bypass:** A coding style where some of the code-block passes are not coded by the arithmetic coder. Instead the bits to be coded are appended directly to the bit stream without coding.
- 3.84 shift:** Multiplication or division of a number by powers of two.
- 3.85 sign bit:** A bit that indicates whether a number is positive (zero value) or negative (one value).
- 3.86 sign-magnitude notation:** A binary representation of an integer where the distance from the origin is expressed with a positive number and the direction from the origin (positive or negative) is expressed with a separate single sign bit.
- 3.87 significance propagation pass:** A coding pass performed on a single bit-plane of a code-block of coefficients.

- 3.88 significance state:** State of a coefficient at a particular bit-plane. If a coefficient, in sign-magnitude notation, has the first magnitude 1 bit at, or before, the given bit-plane it is considered "significant". If not, it is considered "insignificant".
- 3.89 source image:** An image used as input to an encoder.
- 3.90 sub-band:** A group of transform coefficients resulting from the same sequence of low-pass and high-pass filtering operations, both vertically and horizontally.
- 3.91 sub-band coefficient:** A transform coefficient within a given sub-band.
- 3.92 sub-band decomposition:** A transformation of an image tile-component into sub-bands.
- 3.93 superbox:** A box that itself contains a contiguous sequence of boxes (and only a contiguous sequence of boxes). As the JP2 file contains only a contiguous sequence of boxes, the JP2 file is itself considered a superbox. When used as part of a relationship between two boxes, the term "superbox" refers to the box which directly contains the other box.
- 3.94 tile:** A rectangular array of points on the reference grid, registered with and offset from the reference grid origin and defined by a width and height. The tiles which overlap are used to define tile-components.
- 3.95 tile-component:** All the samples of a given component in a tile.
- 3.96 tile index:** The index of the current tile ranging from zero to the number of tiles minus one.
- 3.97 tile-part:** A portion of the codestream with compressed image data for some, or all, of a tile. The tile-part includes at least one, and up to all, of the packets that make up the coded tile.
- 3.98 tile-part header:** A group of markers and marker segments at the beginning of each tile-part in the codestream that describe the tile-part coding parameters.
- 3.99 tile-part index:** The index of the current tile-part ranging from zero to the number of tile-parts minus one in a given tile.
- 3.100 transformation:** A mathematical mapping from one signal space to another.
- 3.101 transform coefficient:** A value that is the result of a transformation.
- 3.102 XOR:** Exclusive OR logical operator.

4 Abbreviations and symbols

4.1 Abbreviations

For the purposes of this Recommendation | International Standard, the following abbreviations apply.

CCITT	International Telegraph and Telephone Consultative Committee, now ITU-T
ICC	International Colour Consortium
ICT	Irreversible Component Transform
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
ITTF	Information Technology Task Force
ITU	International Telecommunication Union
ITU-T	International Telecommunication Union – Telecommunication Standardization Sector (formerly the CCITT)
JPEG	Joint Photographic Experts Group – The joint ISO/ITU committee responsible for developing standards for continuous-tone still picture coding. It also refers to the standards produced by this committee: ITU-T Rec. T.81 ISO/IEC 10918-1, ITU-T Rec. T.83 ISO/IEC 10918-2, ITU-T Rec. T.84 ISO/IEC 10918-3 and ITU-T Rec. T.87 ISO/IEC 14495.
JURA	JPEG Utilities Registration Authority
1D-DWT	One-dimensional Discrete Wavelet Transformation
FDWT	Forward Discrete Wavelet Transformation
IDWT	Inverse Discrete Wavelet Transformation

LSB	Least Significant Bit
MSB	Most Significant Bit
PCS	Profile Connection Space
RCT	Reversible Component Transform
ROI	Region Of Interest
SNR	Signal-to-Noise Ratio
UCS	Universal Character Set
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTF-8	UCS Transformation Format 8
UUID	Universal Unique Identifier
XML	Extensible Markup Language
W3C	World-Wide Web Consortium

4.2 Symbols

For the purposes of this Recommendation | International Standard, the following symbols apply.

0x----	Denotes a hexadecimal number
\nnn	A three-digit number preceded by a backslash indicates the value of a single byte within a character string, where the three digits specify the octal value of that byte
ϵ_b	Exponent of the quantization value for a sub-band defined in QCD and QCC
μ_b	Mantissa of the quantization value for a sub-band defined in QCD and QCC
M_b	Maximum number of bit-planes coded in a given code-block
N_L	Number of decomposition levels as defined in COD and COC
R_b	Dynamic range of a component sample as defined in SIZ
COC	Coding style component marker
COD	Coding style default marker
COM	Comment marker
CRG	Component registration marker
EPH	End of packet header marker
EOC	End of codestream marker
PLM	Packet length, main header marker
PLT	Packet length, tile-part header marker
POC	Progression order change marker
PPM	Packed packet headers, main header marker
PPT	Packed packet headers, tile-part header marker
QCC	Quantization component marker
QCD	Quantization default marker
RGN	Region-of-interest marker
SIZ	Image and tile size marker
SOC	Start of codestream marker
SOP	Start of packet marker
SOD	Start of data marker
SOT	Start of tile-part marker
TLM	Tile-part lengths marker

5 General description

This Recommendation | International Standard describes an image compression system that allows great flexibility, not only for the compression of images, but also for the access into the codestream. The codestream provides a number of mechanisms for locating and extracting portions of the compressed image data for the purpose of retransmission, storage, display, or editing. This access allows storage and retrieval of compressed image data appropriate for a given application, without decoding.

The division of both the original image data and the compressed image data in a number of ways leads to the ability to extract image data from the compressed image data to form a reconstructed image with lower resolution or lower precision, or regions of the original image. This allows the matching of a codestream to the transmission channel, storage device, or display device, regardless of the size, number of components, and sample precision of the original image. The codestream can be manipulated without decoding to achieve a more efficient arrangement for a given application.

Thus, the sophisticated features of this Recommendation | International Standard allow a single codestream to be used efficiently by a number of applications. The largest image source devices can provide a codestream that is easily processed for the smallest image display device, for example.

In general, this Recommendation | International Standard deals with three domains: spatial (samples), transformed (coefficients), and compressed image data. Some entities (e.g., tile-component) have meaning in all three domains. Other entities (e.g., code-block or packet) have meaning in only one domain (e.g., transformed or compressed image data, respectively). The splitting of an entity into other entities in the same domain (e.g., component to tile-components) is described separately for each of the domains.

5.1 Purpose

There are four main elements described in this Recommendation | International Standard:

- Encoder: An embodiment of an encoding process. An encoder takes as input digital source image data and parameter specifications, and by means of a set of procedures generates as output a codestream.
- Decoder: An embodiment of a decoding process. A decoder takes as input compressed image data and parameter specifications, and by means of a specified set of procedures generates as output digital reconstructed image data.
- Codestream syntax: A representation of the compressed image data that includes all parameter specifications required by the decoding process.
- Optional file format: The optional file format is for exchange between application environments. The codestream can be used by other file formats or stand-alone without this file format.

5.2 Codestream

The codestream is a linear stream of bits from the first bit to the last bit. For convenience, it can be divided into (8-bit) bytes, starting with the first bit of the codestream, with the "earlier" bit in a byte viewed as the most significant bit of the byte when given e.g., a hexadecimal representation. This byte stream may be divided into groups of consecutive bytes. The hexadecimal value representation is sometimes implicitly assumed in the text when describing bytes or group of bytes that do not have a "natural" numeric value representation.

5.3 Coding principles

The main procedures for this Recommendation | International Standard are shown in Figure 5-1. This shows the decoding order only. The compressed image data is already conceptually assigned to portions of the image data. Procedures are presented in the Annexes in the order of the decoding process. The coding process is summarized below.

NOTE 1 – Annexes A through I are considered normative to this Recommendation | International Standard. Certain denoted sub-clauses and notes and all examples are informative, however.

Many images have multiple components. This Recommendation | International Standard has a multiple component transformation to decorrelate three components. This is the only function in this Recommendation | International Standard that relates components to each other. (See Annex G.)

The image components may be divided into tiles. These tile-components are rectangular arrays that relate to the same portion of each of the components that make up the image. Thus, tiling of the image actually creates tile-components that can be extracted or decoded independently of each other. This tile independence provides one of the methods for extracting a region of the image. (See Annex B.)

The tile-components are decomposed into different decomposition levels using a wavelet transformation. These decomposition levels contain a number of sub-bands populated with coefficients that describe the horizontal and vertical spatial frequency characteristics of the original tile-components. The coefficients provide frequency information about a local area, rather than across the entire image like the Fourier transformation. That is, a small number of coefficients completely describe a single sample. A decomposition level is related to the next decomposition level by a spatial factor of two. That is, each successive decomposition level of the sub-bands has approximately half the horizontal and half the vertical resolution of the previous. Images of lower resolution than the original are generated by decoding a selected subset of these sub-bands. (See Annex F.)

Although there are as many coefficients as there are samples, the information content tends to be concentrated in just a few coefficients. Through quantization, the information content of a large number of small-magnitude coefficients is further reduced (Annex E). Additional processing by the entropy coder reduces the number of bits required to represent these quantized coefficients, sometimes significantly compared to the original image. (See Annexes C, D, and B.)

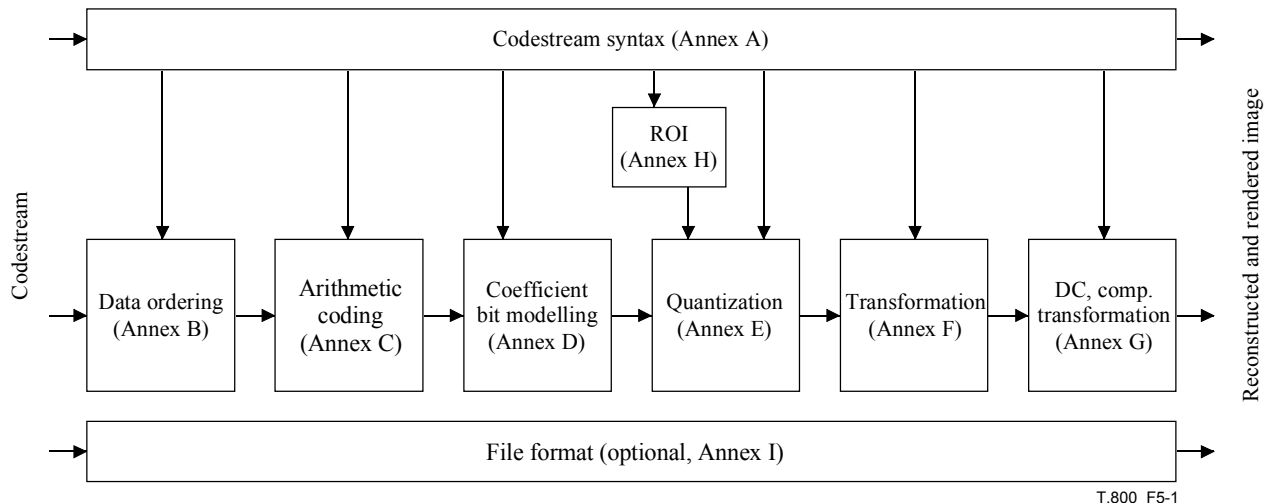


Figure 5-1 – Specification block diagram

The individual sub-bands of a tile-component are further divided into code-blocks. These rectangular arrays of coefficients can be extracted independently. The individual bit-planes of the coefficients in a code-block are coded with three coding passes. Each of these coding passes collects contextual information about the bit-plane compressed image data. (See Annex D.) An arithmetic coder uses this contextual information, and its internal state, to decode a compressed bit stream. (See Annex C.) Different termination mechanisms allow different levels of independent extraction of this coding pass compressed image data.

The bit stream compressed image data created from these coding passes is grouped in layers. Layers are arbitrary groupings of coding passes from code-blocks. (See Annex B.)

NOTE 2 – Although there is great flexibility in layering, the premise is that each successive layer contributes to a higher quality image.

Sub-band coefficients at each resolution level are partitioned into rectangular areas called precincts. (See Annex B.)

Packets are a fundamental unit of the compressed codestream. A packet contains compressed image data from one layer of a precinct of one resolution level of one tile-component. Packets provide another method for extracting a spatial region independently from the codestream. These packets are interleaved in the codestream using a few different methods. (See Annex B.)

A mechanism is provided that allows the compressed image data corresponding to regions of interest in the original tile-components to be coded and placed earlier in the bit stream. (See Annex H.)

Several mechanisms are provided to allow the detection and concealment of bit errors that might occur over a noisy transmission channel. (See D.5 and J.7.)

The codestream relating to a tile, organized in packets, are arranged in one, or more, tile-parts. A tile-part header, comprised of a series of markers and marker segments, contains information about the various mechanisms and coding styles that are needed to locate, extract, decode, and reconstruct every tile-component. At the beginning of the entire codestream is a main header, comprised of markers and marker segments, that offers similar information as well as information about the original image. (See Annex A.)

The codestream is optionally wrapped in a file format that allows applications to interpret the meaning of, and other information about, the image. The file format may contain data besides the codestream. (See Annex I.)

In review, procedures that divide the original image are the following:

- The components of the image are divided into rectangular tiles. The tile-component is the basic unit of the original or reconstructed image.
- Performing the wavelet transformation on a tile-component creates decomposition levels.
- These decomposition levels are made up of sub-bands of coefficients that describe the frequency characteristics of local areas (rather than across the entire tile-component) of the tile-component.
- The sub-bands of coefficients are quantized and collected into rectangular arrays of code-blocks.
- Each bit-plane of the coefficients in a code-block are entropy coded in three types of coding passes.
- Some of the coefficients can be coded first to provide a region of interest.

At this point the image data is fully converted to compressed image data. The procedures that reassemble these bit stream units into the codestream are the following:

- The compressed image data from the coding passes are collected in layers.
- Packets are composed compressed image data from one precinct of a single layer of a single resolution level of a single tile-component. The packets are the basic unit of the compressed image data.
- All the packets from a tile are interleaved in one of several orders and placed in one, or more, tile-parts.
- The tile-parts have a descriptive tile-part header and can be interleaved in some orders.
- The codestream has a main header at the beginning that describes the original image and the various decomposition and coding styles.
- The optional file format describes the meaning of the image and its components in the context of the application.

6 Encoder requirements

An encoding process converts source image data to compressed image data. Annexes A, B, C, D, E, F, G, and H describe the encoding process. All encoding processes are specified informatively.

An encoder is an embodiment of the encoding process. In order to conform to this Recommendation | International Standard, an encoder shall convert source image data to compressed image data, that conform to the codestream syntax specified in Annex A.

7 Decoder requirements

A decoding process converts compressed image data to reconstructed image data. Annexes A through H describe and specify the decoding process. All decoding processes are normative.

A decoder is an embodiment of the decoding process. In order to conform to this Recommendation | International Standard, a decoder shall convert all, or specific parts of, any compressed image data that conform to the codestream syntax specified in Annex A to a reconstructed image.

There is no normative or required implementation for the encoder or decoder. In some cases, the descriptions use particular implementation techniques for illustrative purposes only.

7.1 Codestream syntax requirements

Annex A describes the codestream syntax that defines the coded representation of compressed image data for exchange between application environments. Any compressed image data shall comply with the syntax and code assignments appropriate for the coding processes defined in the Recommendation | International Standard.

This Recommendation | International Standard does not include a definition of compliance or conformance. The parameter values of the syntax described in Annex A are not intended to portray the capabilities required to be compliant.

7.2 Optional file format requirements

Annex I describes the optional file format containing metadata about the image in addition to the codestream. This data allows, for example, screen display or printing at a specific resolution. The optional file format, when used, shall comply with the file format syntax and code assignments appropriate for the coding processes defined in the Recommendation | International Standard.

8 Implementation requirements

There is no normative or required implementation for this Recommendation | International Standard. In some cases, the descriptions use particular implementation techniques for illustrative purposes only.

Annex A

Codestream syntax

(This annex forms an integral part of this Recommendation | International Standard)

In this annex and all of its subclauses, the flow charts and tables are normative only in the sense that they are defining an output that alternative implementations shall duplicate.

This annex specifies the marker and marker segment syntax and semantics defined by this Recommendation | International Standard. These markers and marker segments provide codestream information for this Recommendation | International Standard. Further, this annex provides a marker and marker segment syntax that is designed to be used in future specifications that include this Recommendation | International Standard as a normative reference.

This Recommendation | International Standard does not include a definition of compliance or conformance. The parameter values of the syntax described in this annex are not intended to portray the capabilities required to be compliant.

A.1 Markers, marker segments, and headers

This Recommendation | International Standard uses markers and marker segments to delimit and signal the characteristics of the source image and codestream. This set of markers and marker segments is the minimal information needed to achieve the features of this Recommendation | International Standard and is not a file format. A minimal file format is offered in Annex I.

Main and tile-part headers are collections of markers and marker segments. The main header is found at the beginning of the codestream. The tile-part headers are found at the beginning of each tile-part (see below). Some markers and marker segments are restricted to only one of the two types of headers while others can be found in either.

Every marker is two bytes long. The first byte consists of a single 0xFF byte. The second byte denotes the specific marker and can have any value in the range 0x01 to 0xFE. Many of these markers are already used in ITU-T Rec. T.81 | ISO/IEC 10918-1 and ITU-T Rec. T.84 | ISO/IEC 10918-3 and shall be regarded as reserved unless specifically used.

A marker segment includes a marker and associated parameters, called marker segment parameters. In every marker segment the first two bytes after the marker shall be an unsigned value that denotes the length in bytes of the marker segment parameters (including the two bytes of this length parameter but not the two bytes of the marker itself). When a marker segment that is not specified in the Recommendation | International Standard appears in a codestream, the decoder shall use the length parameter to discard the marker segment.

A.1.1 Types of markers and marker segments

Six types of markers and marker segments are used: delimiting, fixed information, functional, in-bit stream, pointer, and informational. Delimiting markers and marker segments are used to frame the main and tile-part headers and the bit-stream data. Fixed information marker segments give required information about the image. The location of these marker segments, like delimiting marker and marker segments, is specified. Functional marker segments are used to describe the coding functions used. In-bit-stream markers and marker segments are used for error resilience. Pointer marker segments provide specific offsets in the bit stream. Informational marker segments provide ancillary information.

A.1.2 Syntax similarity with ITU-T Rec. T.81 | ISO/IEC 10918-1

The marker and marker segment syntax uses the same construction as defined in ITU-T Rec. T.81 | ISO/IEC 10918-1.

The marker range 0xFF30 to 0xFF3F is reserved by this Recommendation | International Standard for markers without marker segment parameters. Table A.1 shows in which specification these markers and marker segments are defined.

Table A.1 – Marker definitions

Marker code range	Standard definition
0xFF00, 0xFF01, 0xFFFE, 0xFFC0 to 0xFFDF	Defined in ITU-T Rec. T.81 ISO/IEC 10918-1
0xFFF0 to 0xFFF6	Defined in ITU-T Rec. T.84 ISO/IEC 10918-3
0xFFF7 to 0xFFF8	Defined in ITU-T Rec. T.87 ISO/IEC 14495-1
0xFF4F to 0xFF6F, 0xFF90 to 0xFF93	Defined in this Recommendation International Standard
0xFF30 to 0xFF3F	Reserved for definition as markers only (no marker segments)
	All other values reserved

A.1.3 Marker and marker segment and codestream rules

- Marker segments, and therefore the main and tile-part headers, are a multiple of 8 bits (one byte). Further, the bit stream data between the headers and before the EOC marker (see A.4.4) are padded to also be aligned to a multiple of 8 bits.
- All marker segments in a tile-part header apply only to the tile to which they belong.
- All marker segments in the main header apply to the whole image unless specifically overridden by markers or marker segments in a tile-part header.
- Delimiting and fixed information marker and marker segments must appear at specific points in the codestream.
- The marker segments shall correctly describe the image as represented by the codestream. If truncation, alteration, or editing of the codestream has been performed, the marker segments shall be updated, if necessary.
- All parameter values in marker segments are big endian.
- Marker segments can appear in any order in a given header. Exceptions are the delimiting markers and marker segments and the fixed information marker segments.
- All markers with the marker code between 0xFF30 and 0xFF3F have no marker segment parameters. They shall be skipped by the decoder.
- Some marker segments have values assigned to groups of bits within a parameter. In some cases there are bits, denoted by "x," that are not assigned a value for any field within a parameter. The codestream shall contain a value of zero for all such bits. The decoder shall ignore these bits.

NOTE – The markers in the range 0xFF30 to 0xFF3F may be used by future extensions. They may or may not be skipped by a decoder without ramification.

A.1.4 Key to graphical descriptions (informative)

Each marker segment is described in terms of its function, usage, and length. The function describes the information contained in the marker segment. The usage describes the logical location and frequency of this marker segment in the codestream. The length describes which parameters determine the length of the marker segment.

These descriptions are followed by a figure that shows the order and relationship of the parameters in the marker segment. Figure A.1 shows an example of this type of figure. The marker segments are designated by the three-letter code of the marker associated with the marker segment. The parameter symbols have capital letter designations followed by the marker's symbol in lower-case letters. A rectangle is used to indicate a parameter's location in the marker segment. The width of the rectangle is proportional to the number of bytes of the parameter. A shaded rectangle (diagonal stripes) indicates that the parameter is of varying size. Two parameters with superscripts and a grey area between indicate a run of several of these parameters.

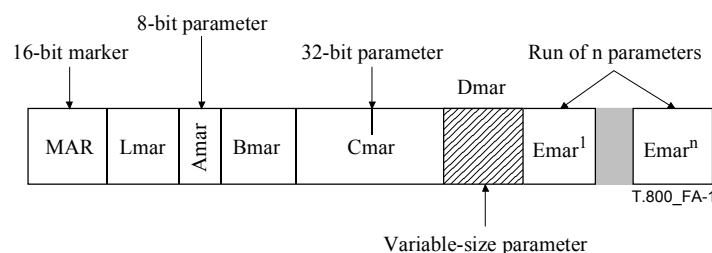


Figure A.1 – Example of the marker segment description figures

The figure is followed by a list that describes the meaning of each parameter in the marker segment. If parameters are repeated, the length and nature of the run of parameters is defined. As an example, in Figure A.1, the first rectangle represents the marker with the symbol MAR. The second rectangle represents the length parameter. Parameters Amar, Bmar, Cmar, and Dmar are 8-, 16-, 32-bit and variable length respectively. The notation Emarⁱ implies that there are n different parameters, Emarⁱ, in a row.

After the list is a table that either describes the allowed parameter values or provides references to other tables that describe these values. Tables for individual parameters are provided to describe any parameter without a simple numerical value. In some cases these parameters are described by a bit value in a bit field. In this case, an "x" is used to denote bits that are not included in the specification of the parameter or sub-parameter in the corresponding row of the table.

Some marker segment parameters are described using the notation "Sxxx" and "SPxxx" (for a marker symbol, XXX). The Sxxx parameter selects between many possible states of the SPxxx parameter. According to this selection, the SPxxx parameter or parameter list is modified.

A.2 Information in the marker segments

Table A.2 lists the markers specified in this Recommendation | International Standard. Table A.3 shows a list of which information is provided by which marker and marker segments.

Table A.2 – List of markers and marker segments

	Symbol	Code	Main header	Tile-part header
Delimiting markers and marker segments				
Start of codestream	SOC	0xFF4F	required ^a	not allowed
Start of tile-part	SOT	0xFF90	not allowed	required
Start of data	SOD	0xFF93	not allowed	last marker
End of codestream	EOC	0xFFD9	not allowed	not allowed
Fixed information marker segments				
Image and tile size	SIZ	0xFF51	required	not allowed
Functional marker segments				
Coding style default	COD	0xFF52	required	optional
Coding style component	COC	0xFF53	optional	optional
Region-of-interest	RGN	0xFF5E	optional	optional
Quantization default	QCD	0xFF5C	required	optional
Quantization component	QCC	0xFF5D	optional	optional
Progression order change ^{b)}	POC	0xFF5F	optional	optional
Pointer marker segments				
Tile-part lengths	PLM	0xFF57	optional	not allowed
Packet length, main header	PLT	0xFF58	not allowed	optional
Packet length, tile-part header	PPM	0xFF60	optional	not allowed
Packed packet headers, main header ^{c)}	PPT	0xFF61	not allowed	optional
Packed packet headers, tile-part header ^{c)}	TLM	0xFF55	optional	not allowed
In-bit-stream markers and marker segments				
Start of packet	SOP	0xFF91	not allowed	not allowed in tile-part header, optional in-bit stream
End of packet header	EPH	0xFF92	optional inside PPM marker segment	optional inside PPT marker segment or in-bit stream
Informational marker segments				
Component registration	CRG	0xFF63	optional	not allowed
Comment	COM	0xFF64	optional	optional
a) "required" means the marker or marker segment shall be in this header; "optional" means it may be used. b) The POC marker segment is required if there are progression order changes. c) Either the PPM or PPT marker segment is required if the packet headers are not distributed in the bit stream. If the PPM marker segment is used then PPT marker segments shall not be used, and vice versa.				

Table A.3 – Information in the marker segments

Information	Marker segment
Capabilities Image area size or reference grid size (height and width) Tile size (height and width) Number of components Component precision Component mapping to the reference grid (sub-sampling)	SIZ
Tile index Tile-part data length	SOT, TLM
Progression order Number of layers Multiple component transformation used	COD
Coding style Number of decomposition levels Code-block size Code-block style Wavelet transformation Precinct size	COD, COC
Region-of-interest shift	RGN
No quantization Quantization derived Quantization expounded	QCD, QCC
Progression starting point Progression ending point Progression order default	POC
Error resilience	SOP
End of packet header	EPH
Packet headers	PPM, PPT
Packet lengths	PLM, PLT
Component registration	CRG
Optional information	COM

A.3 Construction of the codestream

Figure A.2 shows the construction of the codestream. Figure A.3 shows the main header construction. All of the solid lines show required marker segments. The following markers and marker segments are required to be in a specific location: SOC, SIZ, SOT, SOD, and EOC. The dashed lines show optional or possibly not required marker segments. Figure A.4 shows the construction of the first tile-part header in a given tile. Figure A.5 shows the construction of a tile-part header other than the first in a tile.

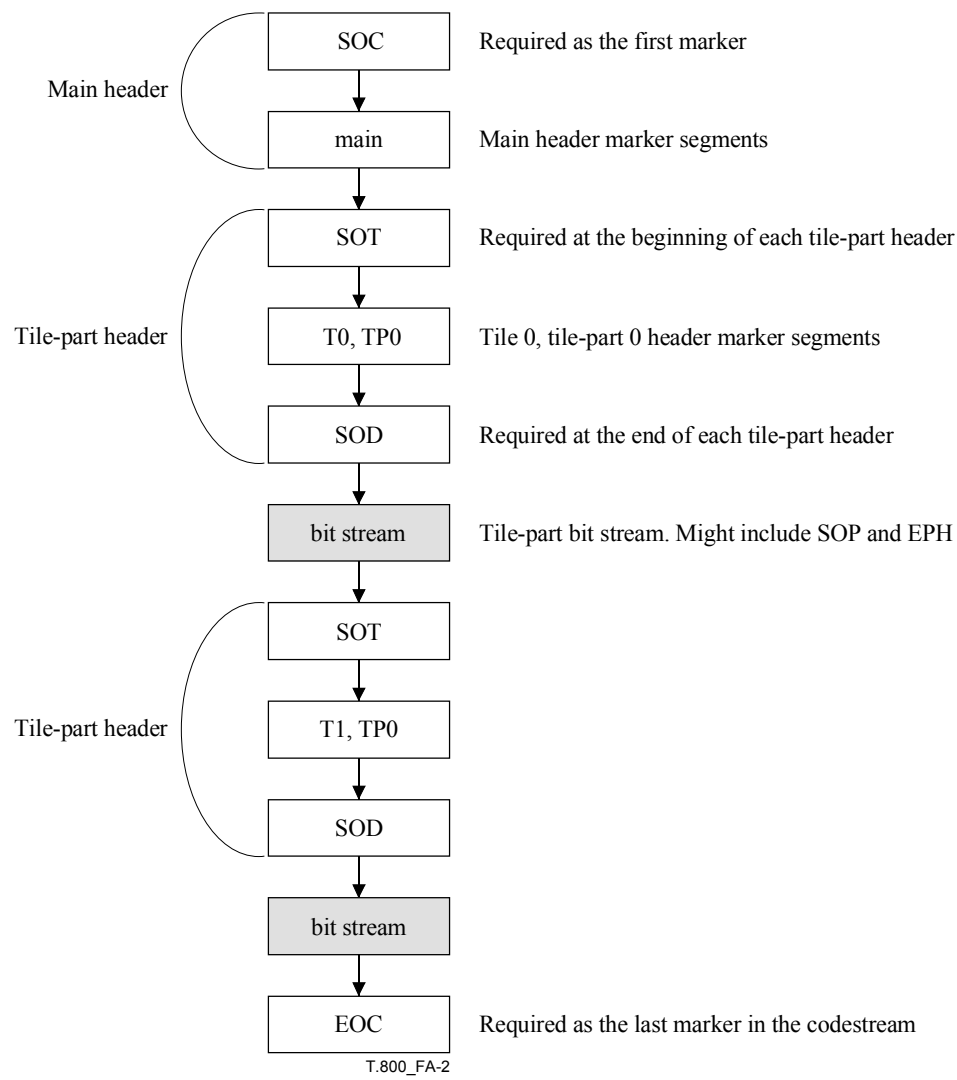


Figure A.2 – Construction of the codestream

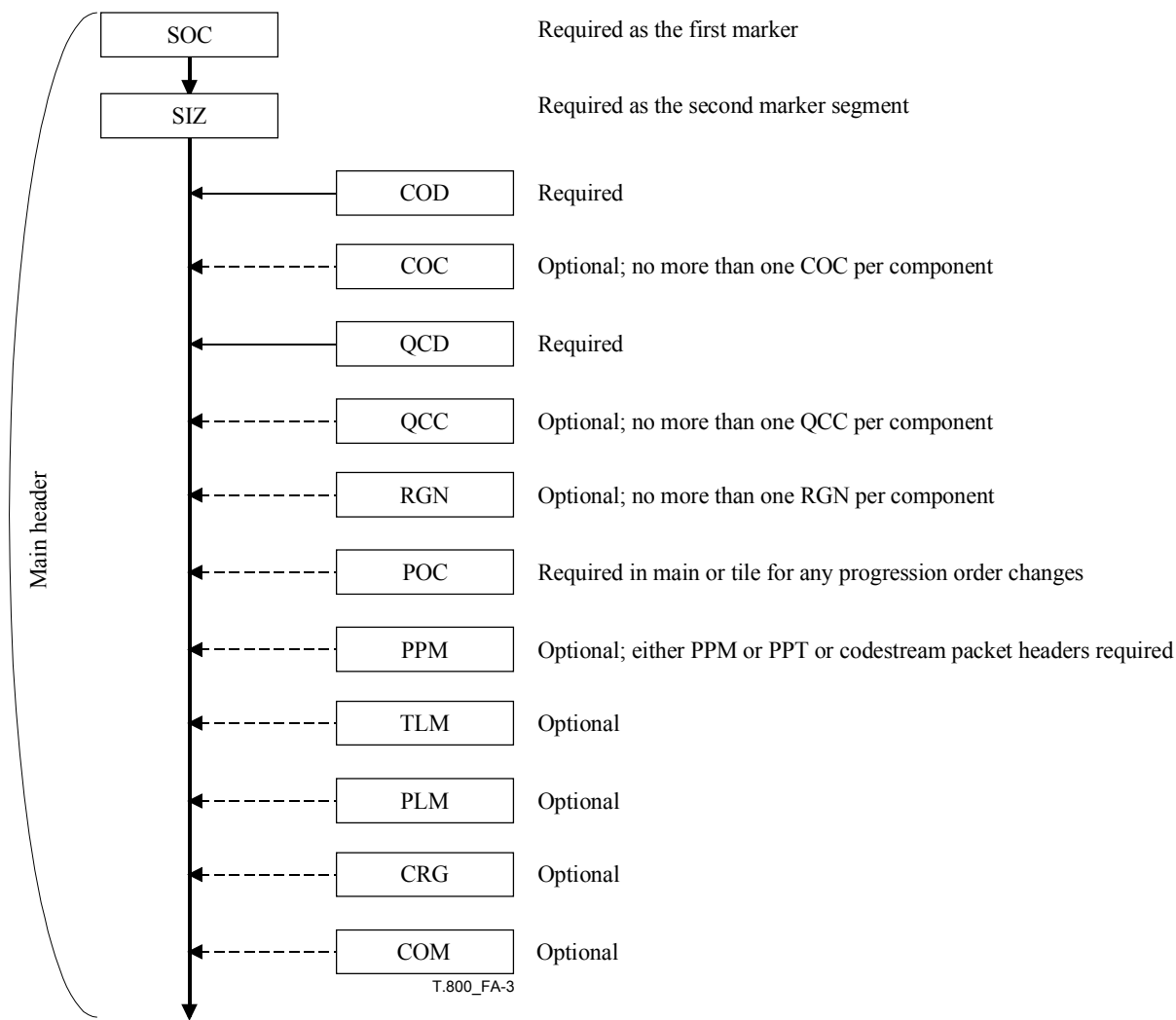


Figure A.3 – Construction of the main header

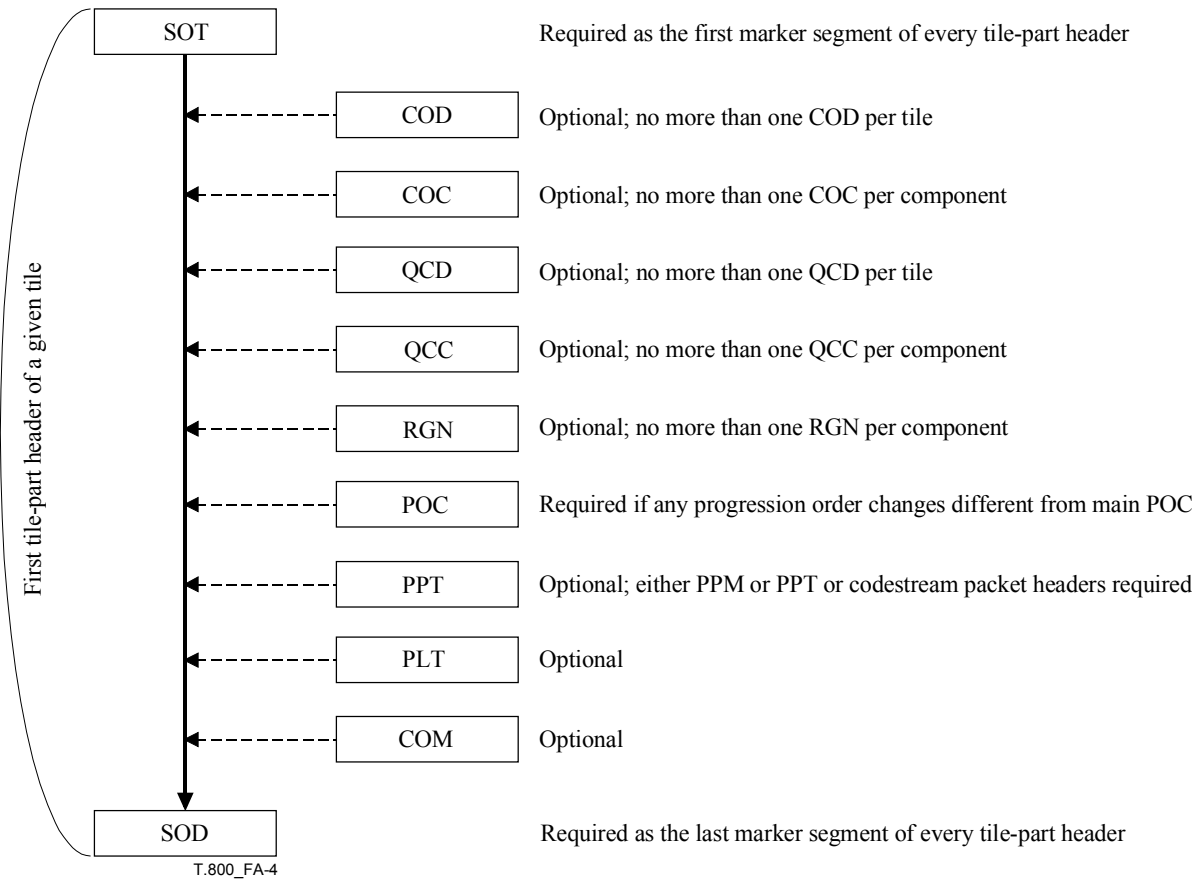


Figure A.4 – Construction of the first tile-part header of a given tile

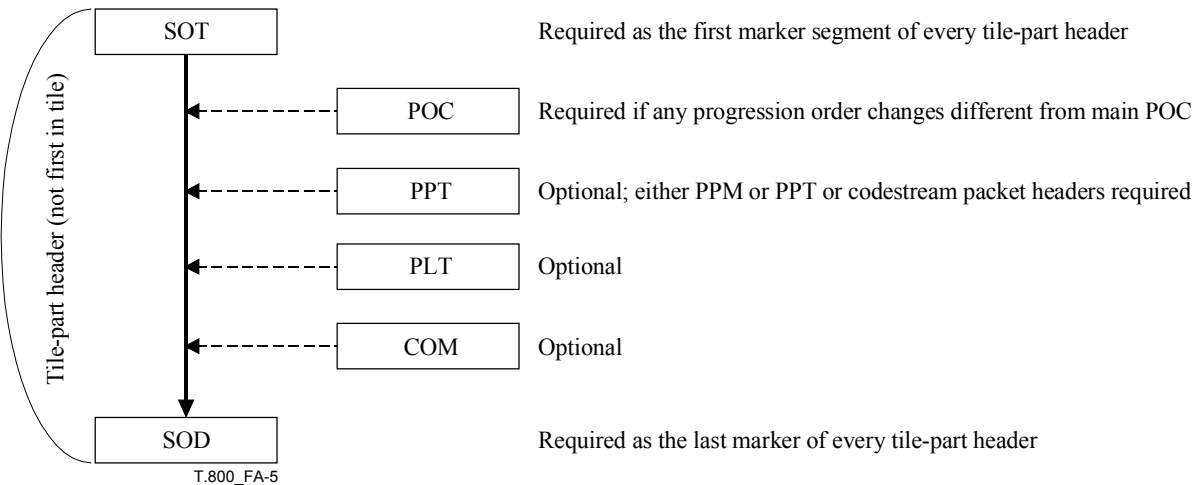


Figure A.5 – Construction of a non-first tile-part header

The COD and COC marker segments and the QCD and QCC marker segments have hierarchy of usage. This is designed to allow tile-components to have dissimilar coding and quantization characteristics with a minimum of signalling.

For example, the COD marker segment is required in the main header. If all components in all the tiles are coded the same way, this is all that is required. If there is one component that is coded differently than the others (for example the luminance component of an image composed of luminance and chrominance components), then the COC can denote that in the main header. If one or more components are coded differently in different tiles, then the COD and COC are used in a similar manner to denote this in the tile-part headers.

The POC marker segment appearing in the main header is used for all tiles unless a different POC appears in the tile-part header.

With the exceptions of the SOC, SOT, SOD, EOC, and SIZ markers and marker segments, the marker segments can appear in any order within the respective headers.

A.4 Delimiting markers and marker segments

The delimiting marker and marker segments shall be present in all codestreams conforming to this Recommendation | International Standard. Each codestream has only one SOC marker, one EOC marker, and at least one tile-part. Each tile-part has one SOT and one SOD marker. The SOC, SOD, and EOC are delimiting markers, not marker segments, and have no explicit length information or other parameters.

A.4.1 Start of codestream (SOC)

Function: Marks the beginning of a codestream specified in this Recommendation | International Standard.

Usage: Main header. This is the first marker in the codestream. There shall be only one SOC per codestream.

Length: Fixed.

SOC: Marker code.

Table A.4 – Start of codestream parameter values

Parameter	Size (bits)	Values
SOC	16	0xFF4F

A.4.2 Start of tile-part (SOT)

Function: Marks the beginning of a tile-part, the index of its tile, and the index of its tile-part. The tile-parts of a given tile shall appear in order (see TP_{sot}) in the codestream. However, tile-parts from other tiles may be interleaved in the codestream. Therefore, the tile-parts from a given tile may not appear contiguously in the codestream.

Usage: Every tile-part header. Shall be the first marker segment in a tile-part header. There shall be at least one SOT in a codestream. There shall be only one SOT per tile-part.

Length: Fixed.

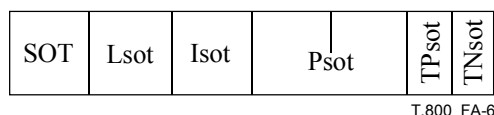


Figure A.6 – Start of tile-part syntax

- SOT:** Marker code. Table A.5 shows the sizes and values of the symbol and parameters for start of tile-part marker segment.
- Lsot:** Length of marker segment in bytes (not including the marker).
- Isot:** Tile index. This number refers to the tiles in raster order starting at the number 0.
- Psot:** Length, in bytes, from the beginning of the first byte of this SOT marker segment of the tile-part to the end of the data of that tile-part. Figure A.16 shows this alignment. Only the last tile-part in the codestream may contain a 0 for Psot. If the Psot is 0, this tile-part is assumed to contain all data until the EOC marker.
- TP_{sot}:** Tile-part index. There is a specific order required for decoding tile-parts; this index denotes the order from 0. If there is only one tile-part for a tile, then this value is zero. The tile-parts of this tile shall appear in the codestream in this order, although not necessarily consecutively.
- TN_{sot}:** Number of tile-parts of a tile in the codestream. Two values are allowed: the correct number of tile-parts for that tile and zero. A zero value indicates that the number of tile-parts of this tile is not specified in this tile-part.

Table A.5 – Start of tile-part parameter values

Parameter	Size (bits)	Values
SOT	16	0xFF90
Lsot	16	10
Isot	16	0 to 65 534
Psot	32	0, or 14 to $(2^{32} - 1)$
TPsot	8	0 to 254
TNsot	8	Table A.6

Table A.6 – Number of tile-parts, TNsot, parameter value

Value	Number of tile-parts
0	Number of tile-parts of this tile in the codestream is not defined in this header
1 to 255	Number of tile-parts of this tile in the codestream

A.4.3 Start of data (SOD)

Function: Indicates the beginning of bit stream data for the current tile-part. The SOD also indicates the end of a tile-part header.

Usage: Every tile-part header. Shall be the last marker in a tile-part header. Bit-stream data between an SOD and the next SOT or EOC (end of image) shall be a multiple of 8 bits – the codestream is padded with bits, as needed. There shall be at least one SOD in a codestream. There shall be one SOD per tile-part.

Length: Fixed.

SOD: Marker code

Table A.7 – Start of data parameter values

Parameter	Size (bits)	Values
SOD	16	0xFF93

A.4.4 End of codestream (EOC)

Function: Indicates the end of the codestream.

NOTE 1 – This marker shares the same code as the EOI marker in ITU-T Rec. T.81 | ISO/IEC 10918-1.

Usage: Shall be the last marker in a codestream. There shall be one EOC per codestream.

NOTE 2 – In the case a file has been corrupted, it is possible that a decoder could extract much useful compressed image data without encountering an EOC marker.

Length: Fixed.

EOC: Marker code

Table A.8 – End of codestream parameter values

Parameter	Size (bits)	Values
EOC	16	0xFFD9

A.5 Fixed information marker segment

This marker segment describes required information about the image. The SIZ marker segment is required in the main header immediately after the SOC marker segment.

A.5.1 Image and tile size (SIZ)

Function: Provides information about the uncompressed image such as the width and height of the reference grid, the width and height of the tiles, the number of components, component bit depth, and the separation of component samples with respect to the reference grid (see B.2).

Usage: Main header. There shall be one and only one in the main header immediately after the SOC marker segment. There shall be only one SIZ per codestream.

Length: Variable depending on the number of components.

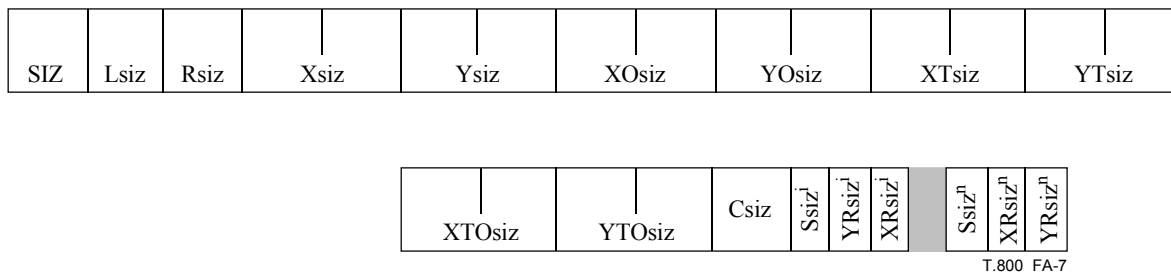


Figure A.7 – Image and tile size syntax

SIZ: Marker code. Table A.9 shows the size and parameter values of the symbol and parameters for image and tile size marker segment.

Lsiz: Length of marker segment in bytes (not including the marker). The value of this parameter is determined by the following equation:

$$Lsiz = 38 + 3 \cdot Csiz \quad (A-1)$$

Rsiz: Denotes capabilities that a decoder needs to properly decode the codestream.

Xsiz: Width of the reference grid.

Ysiz: Height of the reference grid.

XOsiz: Horizontal offset from the origin of the reference grid to the left side of the image area.

YOsiz: Vertical offset from the origin of the reference grid to the top side of the image area.

XTsiz: Width of one reference tile with respect to the reference grid.

YTtiz: Height of one reference tile with respect to the reference grid.

XTOsiz: Horizontal offset from the origin of the reference grid to the left side of the first tile.

YTOsiz: Vertical offset from the origin of the reference grid to the top side of the first tile.

Csiz: Number of components in the image.

Ssizⁱ: Precision (depth) in bits and sign of the *i*th component samples. The precision is the precision of the component samples before DC level shifting is performed (i.e., the precision of the original component samples before any processing is performed). If the component sample values are signed, then the range of component sample values is $-2^{(Ssiz+1 \text{ AND } 0x7F)-1} \leq \text{component sample value} \leq 2^{(Ssiz+1 \text{ AND } 0x7F)-1} - 1$. There is one occurrence of this parameter for each component. The order corresponds to the component's index, starting with zero.

XRsizⁱ: Horizontal separation of a sample of *i*th component with respect to the reference grid. There is one occurrence of this parameter for each component.

YRsizⁱ: Vertical separation of a sample of *i*th component with respect to the reference grid. There is one occurrence of this parameter for each component.

Table A.9 – Image and tile size parameter values

Parameter	Size (bits)	Values
SIZ	16	0xFF51
Lsiz	16	41 to 49 190
Rsiz	16	Table A.10
Xsiz	32	1 to $(2^{32} - 1)$
Ysiz	32	1 to $(2^{32} - 1)$
XOsz	32	0 to $(2^{32} - 2)$
YOsz	32	0 to $(2^{32} - 2)$
XTsiz	32	1 to $(2^{32} - 1)$
YTsiz	32	1 to $(2^{32} - 1)$
XTOsz	32	0 to $(2^{32} - 2)$
YTOsz	32	0 to $(2^{32} - 2)$
Csiz	16	1 to 16 384
Ssiz ⁱ	8	Table A.11
XRsiz ⁱ	8	1 to 255
YRsiz ⁱ	8	1 to 255

Table A.10 – Capability Rsiz parameter

Value (bits)	Capability
MSB	LSB
0000 0000 0000 0000	Capabilities specified in this Recommendation International Standard only
0000 0000 0000 0001	Codestream restricted as described for Profile 0 from Table A.45
0000 0000 0000 0010	Codestream restricted as described for Profile 1 from Table A.45
	All other values reserved

Table A.11 – Component Ssiz parameter

Value (bits)	Component sample precision
MSB	LSB
x000 0000 to x010 0101	Component sample bit depth = value + 1. From 1 bit deep through 38 bits deep respectively (counting the sign bit, if appropriate) ^{a)} , R_I
0xxx xxxx	Component sample values are unsigned values
1xxx xxxx	Component sample values are signed values
	All other values reserved
^{a)} The component sample precision is limited by the number of guard bits, quantization, growth of coefficients at each decomposition level and the number of coding passes that can be signalled. Not all combinations of coding styles will allow the coding of 38-bit samples.	

A.6 Functional marker segments

These marker segments describe the functions used to code the entire tile, if found in the tile-part header, or image, if found in the main header.

A.6.1 Coding style default (COD)

Function: Describes the coding style, number of decomposition levels, and layering that is the default used for compressing all components of an image (if in the main header) or a tile (if in the tile-part header). The parameter values can be overridden for an individual component by a COC marker segment in either the main or tile-part header.

Usage: Main and first tile-part header of a given tile. Shall be one and only one in the main header. Additionally, there may be at most one for each tile. If there are multiple tile-parts in a tile, and this marker segment is present, it shall be found only in the first tile-part (TPsot = 0).

When used in the main header, the COD marker segment parameter values are used for all tile-components that do not have a corresponding COC marker segment in either the main or tile-part header. When used in the tile-part header it overrides the main header COD and COCs and is used for all components in that tile without a corresponding COC marker segment in the tile-part. Thus, the order of precedence is the following:

Tile-part COC > Tile-part COD > Main COC > Main COD

where the "greater than" sign, >, means that the greater overrides the lessor marker segment.

Length: Variable depending on the value of Scod.



Figure A.8 – Coding style default syntax

COD: Marker code. Table A.12 shows the size and values of the symbol and parameters for coding style, default marker segment.

Lcod: Length of marker segment in bytes (not including the marker). The value of this parameter is determined by the following equation:

$$Lcod = \begin{cases} 12 & \text{maximum\_precincts} \\ 13 + \text{number\_decomposition\_levels} & \text{user-defined\_precincts} \end{cases} \quad (\text{A-2})$$

where maximum_precincts and user-defined_precincts are indicated in the Scod parameter and number_decomposition_levels are indicated in the SPcod parameter.

Scod: Coding style for all components. Table A.13 shows the value for the Scod parameter.

SGcod: Parameters for coding style designated in Scod. The parameters are independent of components and are designated, in order from top to bottom, in Table A.14. The coding style parameters within the SGcod field appear in the sequence shown in Figure A.9.

SPcod: Parameters for coding style designated in Scod. The parameters relate to all components and are designated, in order from top to bottom, in Table A.15. The coding style parameters within the SPcod field appear in the sequence shown in Figure A.9.

Table A.12 – Coding style default parameter values

Parameter	Size (bits)	Values
COD	16	0xFF52
Lcod	16	12 to 45
Scod	8	Table A.13
SGcod	32	Table A.14
SPcod	variable	Table A.15

Table A.13 – Coding style parameter values for the Scod parameter

Value (bits)		Coding style
MSB	LSB	
xxxx	xxx0	Entropy coder, precincts with PPx = 15 and PPy = 15
xxxx	xxx1	Entropy coder with precincts defined below
xxxx	xx0x	No SOP marker segments used
xxxx	xx1x	SOP marker segments may be used
xxxx	x0xx	No EPH marker used
xxxx	x1xx	EPH marker shall be used
		All other values reserved

Table A.14 – Coding style parameter values of the SGcod parameter

Parameters (in order)	Size (bits)	Values	Meaning of SGcod values
Progression order	8	Table A.16	Progression order
Number of layers	16	1 to 65 535	Number of layers
Multiple component transformation	8	Table A.17	Multiple component transformation usage

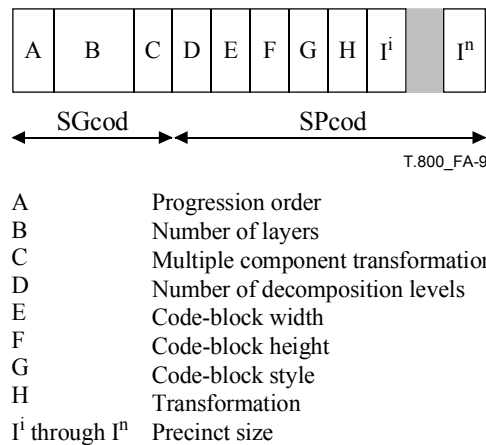


Figure A.9 – Coding style parameter diagram of the SGcod and SPcod parameters

Table A.15 – Coding style parameter values of the SPcod and SPcoc parameters

Parameters (in order)	Size (bits)	Values	Meaning of SPcod values
Number of decomposition levels	8	0 to 32	Number of decomposition levels, N_L . Zero implies no transformation.
Code-block width	8	Table A.18	Code-block width exponent offset value, xcb
Code-block height	8	Table A.18	Code-block height exponent offset value, ycb
Code-block style	8	Table A.19	Style of the code-block coding passes
Transformation	8	Table A.20	Wavelet transformation used
Precinct size	variable	Table A.21	If Scod or Scoc = xxxx xxx0, this parameter is not present; otherwise this indicates precinct width and height. The first parameter (8 bits) corresponds to the N_{LL} sub-band. Each successive parameter corresponds to each successive resolution level in order.

Table A.16 – Progression order for the SGcod, SPcoc, and Ppoc parameters

Value (bits)		Progression order
MSB	LSB	
0000	0000	Layer-resolution level-component-position progression
0000	0001	Resolution level-layer-component-position progression
0000	0010	Resolution level-position-component-layer progression
0000	0011	Position-component-resolution level-layer progression
0000	0100	Component-position-resolution level-layer progression
		All other values reserved

Table A.17 – Multiple component transformation for the SGcod parameters

Value (bits)		Multiple component transformation type
MSB	LSB	
0000	0000	No multiple component transformation specified
0000	0001	Component transformation used on components 0, 1, 2 for coding efficiency (see G.2). Irreversible component transformation used with the 9-7 irreversible filter. Reversible component transformation used with the 5-3 reversible filter
		All other values reserved

Table A.18 – Width or height exponent of the code-blocks for the SPcod and SPcoc parameters

Value (bits)		Code-block width and height
MSB	LSB	
xxxx	0000 to xxxx	Code-block width and height exponent offset value $xcb = value + 2$ or $ycb = value + 2$. The code-block width and height are limited to powers of two with the minimum size being 2^2 and the maximum being 2^{10} . Further, the code-block size is restricted so that $xcb + ycb \leq 12$.
		All other values reserved

Table A.19 – Code-block style for the SPcod and SPcoc parameters

Value (bits)		Code-block style
MSB	LSB	
xxxx	xxx0 xxxx	No selective arithmetic coding bypass Selective arithmetic coding bypass
xxxx	xx0x xxxx	No reset of context probabilities on coding pass boundaries Reset context probabilities on coding pass boundaries
xxxx	x0xx xxxx	No termination on each coding pass Termination on each coding pass
xxxx	0xxx xxxx	No vertically causal context Vertically causal context
xxx0	xxxx xxx1	No predictable termination Predictable termination
xx0x	xxxx xx1x	No segmentation symbols are used Segmentation symbols are used
		All other values reserved

Table A.20 – Transformation for the SPcod and SPcoc parameters

Value (bits)		Transformation type
MSB	LSB	
0000	0000	9-7 irreversible filter
0000	0001	5-3 reversible filter
		All other values reserved

Table A.21 – Precinct width and height for the SPcod and SPcoc parameters

Value (bits)		Precinct size
MSB	LSB	
xxxxx	0000 to xxxxx 1111	4 LSBs are the precinct width exponent, $PPx = value$. This value may only equal zero at the resolution level corresponding to the N_{LL} band.
0000	xxxxx to 1111 xxxxx	4 MSBs are the precinct height exponent $PPy = value$. This value may only equal zero at the resolution level corresponding to the N_{LL} band.

A.6.2 Coding style component (COC)

Function: Describes the coding style and number of decomposition levels used for compressing a particular component.

Usage: Main and first tile-part header of a given tile. Optional in both the main and tile-part headers. No more than one per any given component may be present in either the main or tile-part headers. If there are multiple tile-parts in a tile, and this marker segment is present, it shall be found only in the first tile-part (TPsot = 0).

When used in the main header, it overrides the main COD marker segment for the specific component. When used in the tile-part header, it overrides the main COD, main COC, and tile COD for the specific component. Thus, the order of precedence is the following:

Tile-part COC > Tile-part COD > Main COC > Main COD

where the "greater than" sign, >, means that the greater overrides the lessor marker segment.

Length: Variable depending on the value of Scoc.

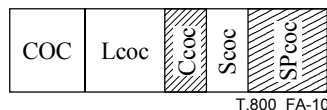


Figure A.10 – Coding style component syntax

COC: Marker code. Table A.22 shows the size and values of the symbol and parameters for coding style component marker segment.

Lcoc: Length of marker segment in bytes (not including the marker). The value of this parameter is determined by the following equation:

$$Lcoc = \begin{cases} 9 & \text{maximum\_precincts AND Csiz} < 257 \\ 10 & \text{maximum\_precincts AND Csiz} \Rightarrow 257 \\ 10 + \text{number\_decomposition\_levels} & \text{user-defined\_precincts AND Csiz} < 257 \\ 11 + \text{number\_decomposition\_levels} & \text{user-defined\_precincts AND Csiz} \Rightarrow 257 \end{cases} \quad (\text{A-3})$$

where maximum_precincts and user-defined_precincts are indicated in the Scoc parameter and number_decomposition_levels is indicated in the SPcoc parameter.

Ccoc: The index of the component to which this marker segment relates. The components are indexed 0, 1, 2, etc.

- Scoc:** Coding style for this component. Table A.23 shows the value for each Scoc parameter.
- SPcoc:** Parameters for coding style designated in Scoc. The parameters are designated, in order from top to bottom, in Table A.15. The coding style parameters within the SPcoc field appear in the sequence shown in Figure A.11.

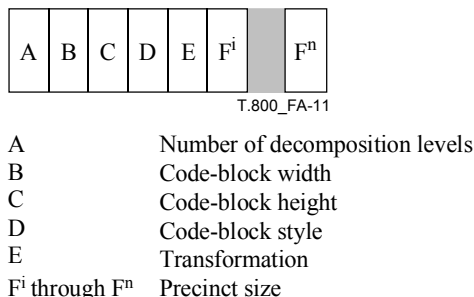


Figure A.11 – Coding style parameter diagram of the SPcoc parameters

Table A.22 – Coding style component parameter values

Parameter	Size (bits)	Values
COC	16	0xFF53
Lcoc	16	9 to 43
Ccoc	8 16	0 to 255; if Csiz < 257 0 to 16 383; Csiz ≥ 257
Scoc	8	Table A.23
SPcoc ⁱ	variable	Table A.15

Table A.23 – Coding style parameter values for the Scoc parameter

Value (bits)	Coding style
MSB LSB	
0000 0000	Entropy coder with maximum precinct values PP _x = PP _y = 15
0000 0001	Entropy coder with precinct values defined below
	All other values reserved

A.6.3 Region of interest (RGN)

Function: Signals the presence of an ROI in the codestream.

Usage: Main and first tile-part header of a given tile. If used in the main header, it refers to the ROI scaling value for one component in the whole image, valid for all tiles except those with an RGN marker segment.

When used in the tile-part header, the scaling value is valid only for one component in that tile. There may be at most one RGN marker segment for each component in either the main or tile-part headers. The RGN marker segment for a particular component which appears in a tile-part header overrides any marker for that component in the main header, for the tile in which it appears. If there are multiple tile-parts in a tile, then this marker segment shall be found only in the first tile-part header.

Length: Variable.

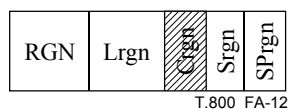


Figure A.12 – Region-of-interest syntax

- RGN:** Marker code. Table A.24 shows the size and values of the symbol and parameters for region-of-interest marker segment.
- Lrgn:** Length of marker segment in bytes (not including the marker).
- Crgn:** The index of the component to which this marker segment relates. The components are indexed 0, 1, 2, etc.
- Srgn:** ROI style for the current ROI. Table A.25 shows the value for the Srgn parameter.
- SPrgn:** Parameter for ROI style designated in Srgn.

Table A.24 – Region-of-interest parameter values

Parameter	Size (bits)	Values
RGN	16	0xFF5E
Lrgn	16	5 to 6
Crgn	8 16	0 to 255; if Csiz < 257 0 to 16 383; Csiz ≥ 257
Srgn	8	Table A.25
SPrgn	8	Table A.26

Table A.25 – Region-of-interest parameter values for the Srgn parameter

Values	ROI style (Srgn)
0	Implicit ROI (maximum shift)
	All other values reserved

Table A.26 – Region-of-interest values from SPrgn parameter (Srgn = 0)

Parameters (in order)	Size (bits)	Values	Meaning of SPrgn value
Implicit ROI shift	8	0 to 255	Binary shifting of ROI coefficients above the background

A.6.4 Quantization default (QCD)

Function: Describes the quantization default used for compressing all components not defined by a QCC marker segment. The parameter values can be overridden for an individual component by a QCC marker segment in either the main or tile-part header.

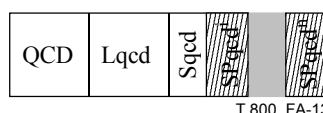
Usage: Main and first tile-part header of a given tile. Shall be one and only one in the main header. May be at most one for all tile-part headers of a tile. If there are multiple tile-parts for a tile, and this marker segment is present, it shall be found only in the first tile-part (TPsot = 0).

When used in the tile-part header it overrides the main QCD and the main QCC for the specific component. Thus, the order of precedence is the following:

Tile-part QCC > Tile-part QCD > Main QCC > Main QCD

where the "greater than" sign, >, means that the greater overrides the lessor marker segment.

Length: Variable depending on the number of quantized elements.

**Figure A.13 – Quantization default syntax**

QCD: Marker code. Table A.27 shows the size and values of the symbol and parameters for quantization default marker segment.

Lqcd: Length of marker segment in bytes (not including the marker). The value of this parameter is determined by the following equation:

$$Lqcd = \begin{cases} 4 + 3 \cdot \text{number\_decomposition\_levels} & \text{no\_quantization} \\ 5 & \text{scalar\_quantization\_derived} \\ 5 + 6 \cdot \text{number\_decomposition\_levels} & \text{scalar\_quantization\_expounded} \end{cases} \quad (\text{A-4})$$

where `number_decomposition_levels` is defined in the COD and COC marker segments, and `no_quantization`, `scalar_quantization_derived`, or `scalar_quantization_expounded` is signalled in the `Sqcd` parameter.

NOTE – The `Lqcd` can be used to determine how many quantization step sizes are present in the marker segment. However, there is not necessarily a correspondence with the number of sub-bands present because the sub-bands can be truncated with no requirement to correct this marker segment.

Sqcd: Quantization style for all components.

SPqcdⁱ: Quantization step size value for the *i*th sub-band in the defined order (see F.3.1). The number of parameters is the same as the number of sub-bands in the tile-component with the greatest number of decomposition levels.

Table A.27 – Quantization default parameter values

Parameter	Size (bits)	Values
QCD	16	0xFF5C
Lqcd	16	4 to 197
Sqcd	8	Table A.28
SPgcd ⁱ	variable	Table A.28

Table A.28 – Quantization default values for the Sqcd and Sqcc parameters

Value (bits) MSB LSB	Quantization style	SPqcd or SPqcc size (bits)	SPqcd or SPqcc usage
xxx0 0000	No quantization	8	Table A.29
xxx0 0001	Scalar derived (values signalled for N_{LL} sub-band only). Use Equation (E-5)	16	Table A.30
xxx0 0010	Scalar expounded (values signalled for each sub-band). There are as many step sizes signalled as there are sub-bands	16	Table A.30
000x xxxx to 111x xxxx	Number of guard bits: 0 to 7		
	All other values reserved		

Table A.29 – Reversible step size values for the SPqcd and SPqcc parameters (reversible transform only)

Value (bits) MSB LSB	Reversible step size values
0000 0xxx to 1111 1xxx	Exponent, ϵ_b , of the reversible dynamic range signalled for each sub-band (see Equation (E-5))
	All other values reserved

Table A.30 – Quantization values for the SPqcd and SPqcc parameters (irreversible transformation only)

Value (bits)	Quantization step size values
MSB LSB xxxx x000 0000 0000 to xxxx x111 1111 1111	Mantissa, μ_b , of the quantization step size value (see Equation (E-3))
0000 0xxx xxxx xxxx to 1111 1xxx xxxx xxxx	Exponent, ϵ_b , of the quantization step size value (see Equation (E-3))

A.6.5 Quantization component (QCC)

Function: Describes the quantization used for compressing a particular component.

Usage: Main and first tile-part header of a given tile. Optional in both the main and tile-part headers. No more than one per any given component may be present in either the main or tile-part headers. If there are multiple tile-parts in a tile, and this marker segment is present, it shall be found only in the first tile-part (TPsot = 0).

Optional in both the main and tile-part headers. When used in the main header, it overrides the main QCD marker segment for the specific component. When used in the tile-part header, it overrides the main QCD, main QCC, and tile QCD for the specific component. Thus, the order of precedence is the following:

Tile-part QCC > Tile-part QCD > Main QCC > Main QCD

where the "greater than" sign, >, means that the greater overrides the lessor marker segment.

Length: Variable depending on the number of quantized elements.

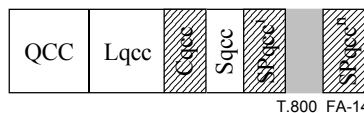


Figure A.14 – Quantization component syntax

QCC: Marker code. Table A.31 shows the size and values of the symbol and parameters for quantization component marker segment.

Lqcc: Length of marker segment in bytes (not including the marker). The value of this parameter is determined by the following equation:

$$Lqcc = \begin{cases} 5 + 3 \cdot \text{number\_decomposition\_levels} & \text{no\_quantization AND Csiz} < 257 \\ 6 & \text{scalar\_quantization\_derived AND Csiz} < 257 \\ 6 + 6 \cdot \text{number\_decomposition\_levels} & \text{scalar\_quantization\_expounded AND Csiz} < 257 \\ 6 + 3 \cdot \text{number\_decomposition\_levels} & \text{no\_quantization AND Csiz} \Rightarrow 257 \\ 7 & \text{scalar\_quantization\_derived AND Csiz} \Rightarrow 257 \\ 7 + 6 \cdot \text{number\_decomposition\_levels} & \text{scalar\_quantization\_expounded AND Csiz} \Rightarrow 257 \end{cases} \quad (\text{A-5})$$

where number_decomposition_levels is defined in the COD and COC marker segments, and no_quantization, scalar_quantization_derived, or scalar_quantization_expounded is signalled in the Sqcc parameter.

NOTE – The Lqcc can be used to determine how many step sizes are present in the marker segment. However, there is not necessarily a correspondence with the number of sub-bands present because the sub-bands can be truncated with no requirement to correct this marker segment.

Cqcc: The index of the component to which this marker segment relates. The components are indexed 0, 1, 2, etc. (either 8 or 16 bits depending on Csiz value).

Sqcc: Quantization style for this component.

SPqccⁱ: Quantization value for each sub-band in the defined order (see F.3.1). The number of parameters is the same as the number of sub-bands in the tile-component with the greatest number of decomposition levels.

Table A.31 – Quantization component parameter values

Parameter	Size (bits)	Values
QCC	16	0xFF5D
Lqcc	16	5 to 199
Cqcc	8 16	0 to 255; if Csiz < 257 0 to 16 383; Csiz ≥ 257
Sqcc	8	Table A.28
SPqcc ⁱ	variable	Table A.28

A.6.6 Progression order change (POC)

Function: Describes the bounds and progression order for any progression order other than specified in the COD marker segments in the codestream.

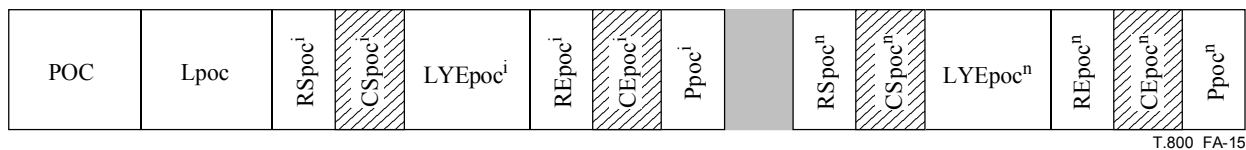
Usage: Main and tile-part headers. At most one POC marker segment may appear in any header. However, several progressions can be described with one POC marker segment. If a POC marker segment is used in the main header, it overrides the progression order in the main and tile COD marker segments. If a POC is used to describe the progression of a particular tile, a POC marker segment must appear in the first tile-part header of that tile. Thus, the progression order of a given tile is determined by the presence of the POC or the values of the COD in the following order of precedence:

Tile-part POC > Main POC > Tile-part COD > Main COD

where the "greater than" sign, >, means that the greater overrides the lessor marker segment.

In the case where a POC marker segment is used, the progression of every packet in the codestream (or for that tile of the codestream) shall be defined in one or more POC marker segments. Each progression order is described in only one POC marker segment and shall be described in any tile-part header before any packets of that progression are found.

Length: Variable depending on the number of different progressions.

**Figure A.15 – Progression order change tile syntax**

POC: Marker value. Table A.32 shows the size and values of the symbol and parameters for progression order change marker segment.

Lpoc: Length of marker segment in bytes (not including the marker). The value of this parameter is determined by the following equation:

$$Lpoc = \begin{cases} 2 + 7 \cdot \text{number\_progression\_order\_change} & \text{Csiz} < 257 \\ 2 + 9 \cdot \text{number\_progression\_order\_change} & \text{Csiz} \geq 257 \end{cases} \quad (\text{A-6})$$

where the number_progression_order_changes is encoder defined.

RSpocⁱ: Resolution level index (inclusive) for the start of a progression. One value for each progression change in this tile or tile-part. The number of progression changes can be derived from the length of the marker segment.

CS pocⁱ: Component index (inclusive) for the start of a progression. The components are indexed 0, 1, 2, etc. (either 8 or 16 bits depending on Csiz value). One value for each progression change in this tile or tile-part. The number of progression changes can be derived from the length of the marker segment.

- LYEpocⁱ:** Layer index (exclusive) for the end of a progression. The layer index always starts at zero for every progression. Packets that have already been included in the codestream are not included again. One value for each progression change in this tile or tile-part. The number of progression changes can be derived from the length of the marker segment.
- REpocⁱ:** Resolution Level index (exclusive) for the end of a progression. One value for each progression change in this tile or tile-part. The number of progression changes can be derived from the length of the marker segment.
- CEpocⁱ:** Component index (exclusive) for the end of a progression. The components are indexed 0, 1, 2, etc. (either 8 or 16 bits depending on Csiz value). One value for each progression change in this tile or tile-part. The number of progression changes can be derived from the length of the marker segment.
- Ppocⁱ:** Progression order. One value for each progression change in this tile or tile-part. The number of progression changes can be derived from the length of the marker segment.

Table A.32 – Progression order change, tile parameter values

Parameter	Size (bits)	Values
POC	16	0xFF5F
Lpoc	16	9 to 65 535
RSpoc ⁱ	8	0 to 32
CSpoc ⁱ	8 16	0 to 255; if Csiz < 257 0 to 16 383; Csiz ≥ 257
LYEpoc ⁱ	16	1 to 65 535
REpoc ⁱ	8	(RSpoc ⁱ + 1) to 33
CEpoc ⁱ	8 16	(CSpoc ⁱ + 1) to 255, 0; if Csiz < 257 (CSpoc ⁱ + 1) to 16 384, 0; Csiz ≥ 257 (0 is interpreted as 256)
Ppoc ⁱ	8	Table A.16

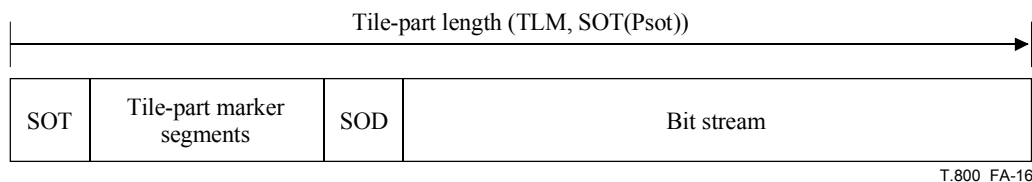
A.7 Pointer marker segments

Pointer marker segments either provide a length or pointer into the codestream. The TLM marker segment describes the length of the tile-parts. It has the same length information as the SOT marker segment. The PLM or PLT marker segment describes the length of the packets.

NOTE – Having the pointer marker segments all occur in the main header allows direct access into the bit-stream data. Having the pointer information in the tile-part headers removes the burden on the encoder of rewinding to store the information.

The TLM (Ptlm) or the SOT (Psot) parameters point from the beginning of the current tile-part's SOT marker segment to the end of the bit-stream data in that tile-part. Because tile-parts are required to be a multiple of 8 bits, these values are always a byte length. Figure A.16 shows the length of a tile-part.

The PLM or PLT marker segments are optional. The PLM marker segment is used in the main header and the PLT marker segments are used in tile-part headers. The PLM and PLT marker segments describe the lengths of each packet in the codestream.



T.800_FA-16

Figure A.16 – Tile-part lengths

A.7.1 Tile-part lengths (TLM)

Function: Describes the length of every tile-part in the codestream. Each tile-part's length is measured from the first byte of the SOT marker segment to the end of the bit-stream data of that tile-part. The value of each individual tile-part length in the TLM marker segment is the same as the value in the corresponding Psot in the SOT marker segment.

Usage: Main header. Optional use in the main header only. There may be multiple TLM marker segments in the main header.

Length: Variable depending on the number of tile-parts in the codestream.

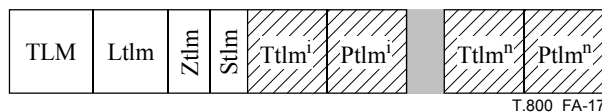


Figure A.17 – Tile part length syntax

TLM: Marker code. Table A.33 shows the size and values of the symbol and parameters for the tile-part length marker segment.

LtIm: Length of marker segment in bytes (not including the marker). The value of this parameter is determined by the following equation:

$$LtIm = \begin{cases} 4 + 2 \cdot \text{number\_of\_tile\_parts\_in\_marker\_segment} & ST = 0 \text{ AND } SP = 0 \\ 4 + 3 \cdot \text{number\_of\_tile\_parts\_in\_marker\_segment} & ST = 1 \text{ AND } SP = 0 \\ 4 + 4 \cdot \text{number\_of\_tile\_parts\_in\_marker\_segment} & ST = 2 \text{ AND } SP = 0 \\ 4 + 4 \cdot \text{number\_of\_tile\_parts\_in\_marker\_segment} & ST = 0 \text{ AND } SP = 1 \\ 4 + 5 \cdot \text{number\_of\_tile\_parts\_in\_marker\_segment} & ST = 1 \text{ AND } SP = 1 \\ 4 + 6 \cdot \text{number\_of\_tile\_parts\_in\_marker\_segment} & ST = 2 \text{ AND } SP = 1 \end{cases} \quad (A-7)$$

where *number_of_tile-parts_in_marker_segment* is the number of tile-part lengths that are denoted in this marker segment; *ST* and *SP* are signalled by *Stlm* parameter.

Ztlm: Index of this marker segment relative to all other TLM marker segments present in the current header. The sequence of (*Ttlmⁱ*, *Ptlmⁱ*) pairs from this marker segment is concatenated, in order of increasing *Ztlm*, with the sequences of pairs from other marker segments. The *j*th entry in the resulting list contains the tile index and tile-part length pair for the *j*th tile-part appearing in the codestream.

Stlm: Size of the *Ttlm* and *Ptlm* parameters.

Ttlmⁱ: Tile index of the *i*th tile-part. Either none or one value for every tile-part. The number of tile-parts in each tile can be derived from this marker segment (or the concatenated list of all such markers) or from a non-zero *TNstot* parameter, if present.

Ptlmⁱ: Length, in bytes, from the beginning of the SOT marker of the *i*th tile-part to the end of the bit stream data for that tile-part. One value for every tile-part.

Table A.33 – Tile-part length parameter values

Parameter	Size (bits)	Values
TLM	16	0xFF55
LtIm	16	6 to 65 535
Ztlm	8	0 to 255
Stlm	8	Table A.34
Ttlm ⁱ	0 if <i>ST</i> = 0 8 if <i>ST</i> = 1 16 if <i>ST</i> = 2	tiles in order 0 to 254 0 to 65 534
Ptlm ⁱ	16 if <i>SP</i> = 0 32 if <i>SP</i> = 1	14 to 65 535 14 to (2 ³² – 1)

Table A.34 – Size parameters for Stlm

Value (bits)		Parameter size
MSB	LSB	
xx00	xxxx	ST = 0; Ttlm parameter is 0 bits, only one tile-part per tile and the tiles are in index order without omission or repetition
xx01	xxxx	ST = 1; Ttlm parameter 8 bits
xx10	xxxx	ST = 2; Ttlm parameter 16 bits
x0xx	xxxx	SP = 0; Ptlm parameter 16 bits
x1xx	xxxx	SP = 1; Ptlm parameter 32 bits
		All other values reserved

A.7.2 Packet length, main header (PLM)

Function: A list of packet lengths in the tile-parts for every tile-part in order.

Usage: Main header. There may be multiple PLM marker segments. Both the PLM and PLT marker segments are optional and can be used together or separately.

Length: Variable depending on the number of tile-parts in the image and the number of packets in each tile-part.

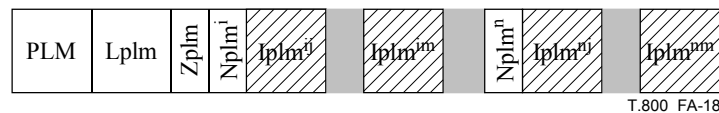


Figure A.18 – Packets length, main header syntax

PLM: Marker code. Table A.35 shows the size and values of the symbol and parameters for the packet length, main header marker segment.

Lplm: Length of marker segment in bytes (not including the marker).

Zplm: Index of this marker segment relative to all other PLM marker segments present in the current header. The sequence of (Nplmⁱ, Iplmⁱ) parameters from this marker segment is concatenated, in order of increasing Zplm, with the sequences of parameters from other marker segments. The kth entry in the resulting list contains the number of bytes and packet header pair for the kth tile-part appearing in the codestream.

Every marker segment in this series shall end with a completed packet header length. However, the series of Iplm parameters described by the Nplm does not have to be complete in a given marker segment. Therefore, it is possible that the next PLM marker segment will not have a Nplm parameter after Zplm, but the continuation of the Iplm series from the last PLM marker segment.

Nplmⁱ: Number of bytes of Iplm information for the ith tile-part in the order found in the codestream. One value for each tile-part. If a codestream contains one, or more, tile-parts exceeding the limitations of PLM markers, these markers shall not be used.

NOTE – This value is expressed with an 8-bit number limiting the number of Iplm bytes to 255 and the number of packets in a tile-part to 255, or less. This is not a restriction on the number of packets that can be in a tile-part. It is merely a limit on this marker segment's ability to describe the packets in a tile-part.

Iplm^{ij}: Length of the jth packet in the ith tile-part. If packet headers are stored with the packet, this length includes the packet header. If packet headers are stored in PPM or PPT, this length does not include the packet header length. One range of values for each tile-part. One value for each packet in the tile.

Table A.35 – Packets length, main header parameter values

Parameter	Size (bits)	Values
PLM	16	0xFF57
Lplm	16	4 to 65 535
Zplm	8	0 to 255
Nplm ⁱ	8	0 to 255
Iplm ^{ij}	variable	Table A.36

Table A.36 – Iplm, Iplt list of packet lengths

Parameters (in order)	Size (bits)	Values	Meaning of Iplm or Iplt values
Packet length	8 bits repeated as necessary	0xxx xxxx 1xxx xxxx x000 0000 to x111 1111	Last 7 bits of packet length, terminate number ^{a)} Continue reading ^{b)} 7 bits of packet length
^{a)} These are the last 7 bits that make up the packet length. ^{b)} These are not the last 7 bits that make up the packet length. Instead, these 7 bits are a portion of those that make up the packet length. The packet length has been broken into 7-bit segments which are sent in order from the most significant segment to the least significant segment. Furthermore, the bits in the most significant segment are right justified to the byte boundary. For example, a packet length of 128 is signalled as 1000 0001 0000 0000, while a length of 512 is signalled as 1000 0100 0000 0000.			

A.7.3 Packet length, tile-part header (PLT)

Function: A list of packet lengths in the tile-part.

Usage: Tile-part headers. There may be multiple PLT marker segments per tile. Both the PLM and PLT marker segments are optional and can be used together or separately. Shall appear in any tile-part header before the packets whose lengths are described herein.

Length: Variable depending on the number of packets in each tile-part.

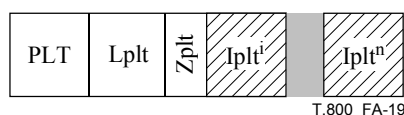


Figure A.19 – Packet length, tile-part header syntax

- PLT:** Marker code. Table A.37 shows the size and values of the symbol and parameters for the packet length, tile-part header marker segment.
- Lplt:** Length of marker segment in bytes (not including the marker).
- Zplt:** Index of this marker segment relative to all other PLT marker segments present in the current header. The sequence of (Ipltⁱ) parameters from this marker segment is concatenated, in order of increasing Zplt, with the sequences of parameters from other marker segments. Every marker segment in this series shall end with a completed packet header length.
- Ipltⁱ:** Length of the ith packet. If packet headers are stored with the packet, this length includes the packet header. If packet headers are stored in PPM or PPT, this length does not include the packet header lengths.

Table A.37 – Packet length, tile-part headers parameter values

Parameter	Size (bits)	Values
PLT	16	0xFF58
Lplt	16	4 to 65 535
Zplt	8	0 to 255
Iplt ⁱ	variable	Table A.36

A.7.4 Packed packet headers, main header (PPM)

Function: A collection of the packet headers from all tiles.

NOTE – This is useful so multiple reads are not required to decode headers.

Usage: Main header. May be used in the main header for all tile-parts unless a PPT marker segment is used in the tile-part header.

The packet headers shall be in only one of three places within the codestream. If the PPM marker segment is present, all the packet headers shall be found in the main header. In this case, the PPT marker segment and packets distributed in the bit stream of the tile-parts are disallowed.

If there is no PPM marker segment then the packet headers can be distributed either in PPT marker segments or distributed in the codestream as defined in B.10. The packet headers shall not be in both a PPT marker segment and the codestream for the same tile. If the packet headers are in PPT marker segments, they shall appear in a tile-part header before the corresponding packet data appears (i.e., in the same tile-part header or one with a lower TP_{sot} value). There may be multiple PPT marker segments in a tile-part header.

Length: Variable depending on the number of packets in each tile-part and the size of the packet headers.

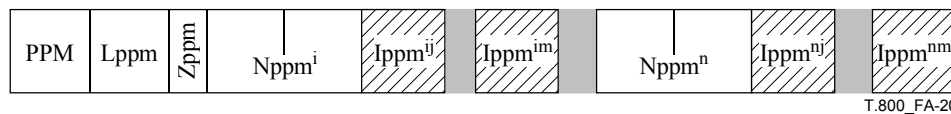


Figure A.20 – Packed packet headers, main header syntax

PPM: Marker code. Table A.38 shows the size and values of the symbol and parameters for the packed packet headers, main header marker segment.

Lppm: Length of marker segment in bytes, not including the marker.

Zppm: Index of this marker segment relative to all other PPM marker segments present in the main header. The sequence of (Nppmⁱ, Ippmⁱ) parameters from this marker segment is concatenated, in order of increasing Zppm, with the sequences of parameters from other marker segments. The kth entry in the resulting list contains the number of bytes and packet headers for the kth tile-part appearing in the codestream.

Every marker segment in this series shall end with a completed packet header. However, the series of Ippm parameters described by the Nppm does not have to be complete in a given marker segment. Therefore, it is possible that the next PPM marker segment will not have a Nppm parameter after Zppm, but the continuation of the Ippm series from the last PPM marker segment.

Nppmⁱ: Number of bytes of Ippm information for the ith tile-part in the order found in the codestream. One value for each tile-part (not tile).

Ippm^{ij}: Packet header for every packet in order in the tile-part. The contents are exactly the packet header which would have been distributed in the bit stream as described in B.10.